TM Forum Open APIs

Conformance Certification

Company Name: GDi d.o.o.

TM Forum Open API Name: 
**TMF641 - Service Ordering Management**

TM Forum Open API Release Version: **4.1.0**

Report Date: **11/10/2023**
1. What Product or Solution does your API support?

GDi Ensemble for OSS Order Management System (OMS) supports the TMF 641 Service Ordering API Rest Specification, and it is a subject of the certification process. It is a central and crucial software component that acts as a mediator during the service order orchestration and fulfilment system between other GDi Ensemble suite software components:

- Workforce Management System
- Workflow Management System
- Asset Management System
- Scheduling System

Furthermore, it represents a central hub for data exchange between the GDi Ensemble for OSS software suite and other third-party systems that are part of the overall end-to-end service provisioning process:

- Customer relationship management system
- Billing systems
- Enterprises resource planning systems
- QoS – quality of service systems
- Ticketing (troubleshooting) systems
- Customer experience management systems
- Mediation systems
- Service and Resource Inventory systems

It relies on its third party-agnostic data model and metadata structure that holds the information regarding the customer’s services and resources during the fulfilment system. The interconnection with other third-party systems is implemented using the specialized line of adapters that act as a export-transform-load components that re-map the data from the OMS’s internal data structure to specialized data models recognizable and acceptable by the external systems.
2. Overview of Certified API

The subject of the certification process is TMF 641 Service Ordering API implemented within the GDi Ensemble for OSS Order Management System (OMS). The OMS component has fully implemented all necessary API methods with appropriate underlying data schemas and notification resource models defined by the TMF Forum Specification Service Ordering Management API User Guide v4.1.0.

All of the required API methods are covered and implemented within the OMS code base. The following picture demonstrates the swagger view of the implemented OMS Service Order Management API.

![Swagger View](image)

All of the required API methods are covered and implemented within the OMS code base.

Included OMS Service Order Management API methods:
- List and filtering of service orders
- Retrieve service orders
- Create service order
- Patch service order
- Delete service order

Using the official TMF API testing agent the OMS exposed and implemented APIs were successfully tested and the result report, for API conformance confirmation, was generated.

3. Architectural View

The Order Management System (OMS) within the GDi Ensemble for OSS is a sophisticated and agile solution tailored for the Communication, Media, and Telecommunication (CMT) domain. Its architecture is a testament to modern software design principles, emphasizing flexibility, scalability, and modularization. The OMS architecture follows a microservices-oriented approach, enabling it to efficiently handle the complexities of service order orchestration and fulfillment.

Microservices-Oriented Architecture

At the core of the OMS architecture is the microservices paradigm. This design philosophy decomposes the system into a set of small, independent services, each responsible for a specific business capability. This modular approach offers several advantages:

- **Scalability**: Microservices can be independently scaled based on demand. This ensures optimal resource utilization and the ability to handle varying workloads.

- **Flexibility**: Changes and updates can be made to individual microservices without impacting the entire system. This agility is crucial in adapting to evolving business requirements.
• **Isolation**: Microservices operate in isolation, reducing the risk of system-wide failures. This isolation also facilitates independent deployment and versioning.

Communication between the OMS microservices is realized by using the queue-based messaging system.

The following picture illustrates the overall architecture of the OMS system.

![Figure 2. Order Management System Architecture View](image)

**Key Components**

1. **North Bound System Adapter**

The North Bound System Adapter serves as the gateway for external systems (foremost the northbound business support systems BSS – such as customer relationship management system) to interact with the OMS. It handles the adaptation and translation of data between different systems and protocols. This component is crucial for ensuring seamless communication with external entities, playing a vital role
in interoperability. This component relies on configurations defined within the Configuration Exchange module to perform the mapping between the original input order and data structure demanded by the Order Capture component that is triggered via the TMF 641 Service Ordering API. To be able to include all necessary information for TMF 641 input request, the NBS Adapter component has to include the information regarding the already activated services present in the Service / Resource Inventory system for the targeted customer. While the NBS adapter component can include some implementation specifics of the source NB system, once triggered, the OMS system is internally performing service processing activities using its own data models and structures known as “contexts”. These data structures and metadata remain identical regardless of the type of external systems that OMS is connected with.

![Image of OMS Order Capture component](image)

**Figure 3. OMS Order Capture component receiving input order via TMF 641 Service Ordering REST API**

The part of the OMS architecture receiving the input order defined by TMF 641 guidelines is defined with the red rectangle area on the above picture.

### 2. Order Capture Component

The Order Capture Component is responsible for receiving incoming service orders, validating them, and preparing them for further processing. It acts as the entry point for orders initiated through the TMF 641 Service Ordering API. This component plays
a pivotal role in the accuracy and completeness of order information. This component stores the original input request within the OMS order database. The validation component has a set of configurable system and business rules that inspect the correctness and completeness of input request data and informs the calling system in case of some data mismatch, inconsistencies, etc. Once validated the OMS order is forwarded to the context creator component.

3. Context Creator and Configuration Exchange

The Context Creator manages the contextual information associated with service orders. It relies on context and configuration exchange systems to maintain a coherent understanding of the order throughout its lifecycle. This component is instrumental in orchestrating complex service order workflows by ensuring that the necessary context is available to downstream processes. Decomposition subcomponent is called in order to further process the input Customer Facing Services (CFSs) into hierarchical CFS-RFS service tree defined by the configuration stored in the external Service Catalog system. The CFS-RFS catalog specification and hierarchical metamodel is also stored in the OMS’s own Service Inventory configuration files defined within the Configuration Manager. Decomposition module is responsible to extract the incoming CFS information and enrich with appropriate resource data and their characteristics relevant to successful workflow process initialization and service activation. Additionally, the decomposition sub-component can, using the OMS configuration via Configuration Exchange service, automatically calculate the necessary field task and material normative needed for the specific installation process triggered by the input order. The Context Creator component, after the decomposition phase, creates the OMS Order Context and OMS Inventory context entities and stores in the OMS Context database using the OMS Context Exchange microservice module as a mediator. The calculated CFS-RFS tree data structure is passed to the external Service Inventory module where the actual CFS/RFS objects are instantiated. The Context Creator component is also responsible, based on the targeted Inventory state of customer service and input order (contexts), for creating a set of decision variables that will be passed to the workflow process instance to guide and direct process execution flow within the process decision elements (gateways, if-then-else components). The input information is used to compute in advance the value of process-driving variables that
will be passed to the process so that the workflow engine itself does not need to include any additional implementation (scripts) to calculate the decision variables on-the-fly. It is important to state that the Context creator can cover only the initial part of the automatic process execution flow that is not obstructed by user tasks (manual) requiring some user-defined input to forward the process execution. Once the manual task occurs in the process execution flow, the workflow engine itself has to implement the logic for execution of decision-based elements. Once the process-driving variables are calculated within this module, the Process Starter module is triggered.

4. Process Starter

The Process Starter component initiates and coordinates the various processes involved in order fulfillment. It acts as an orchestrator, directing the flow of the order through the different microservices. This component ensures that each step in the fulfillment process is executed in the correct sequence, guaranteeing the overall integrity of the service order. The OMS system has possibilities to trigger and execute service fulfilment processes based on different service provisioning approaches:

- Trigger processes based on order type, one process per order type. Example: one integrated process for service activation, another process for deactivation, etc.
- RFS-based process triggering. Depending on the type and number of RFSs calculated in the decomposition phase, the process starter engine can start multiple (one-per RFS) processes within the workflow system. The implementation of each RFS process flow has to take into consideration the provisioning sequence of other RFSs in order to perform a synchronized service activation / deactivation. The synchronization process is implemented using an event-based approach to communicate between multiple RFS process instances during execution.
- Single process instantiation – in some implementation the logic of activation / deactivation of specific services is handled by the workflow-designed process itself. Using the workflow decision elements and gateways, as well as service tasks for integration with OMS configurations and context data, the single process can include all necessary business logic to decide the appropriate process branch that will be executed for the specific input order.
Once the process instance is created within the workflow engine system, the OMS flow ends, while the OMS component acts as a mediator between the multiple external systems including the GDi Ensemble for OSS systems such as workflow management, workforce management, scheduling systems, etc. The integration is fulfilled via the BPM Exchange module that serves and transforms the data stored within the OMS configuration and context managers. All the information regarding the current state of the order as well as current state of customer services and resources are backpropagated from external workflow and workforce systems to the central OMS context manager module that holds the “ground-truth” information within the context data structure.

5. BPM Exchange Module

Collaboration with Business Process Management (BPM) is facilitated through the BPM Exchange API. This component allows seamless integration with BPM systems, enabling the OMS to leverage BPM capabilities for complex process management. The API ensures a synchronized and optimized workflow by exchanging information with the BPM system. OMS is a workflow and workforce-agnostic system that can serve various GDi Ensemble or other third-party solutions with service order provisioning related information during fulfilment process. It relies on the static context structures that hold the current information regarding the state of process execution, data related to customer services and resources, etc. The external service-fulfilment-systems are dependent on the information stored in the OMS context that is provided via the BPM Exchange module. This module is implemented also in an adapter-like way including some specifics of the targeted workforce and workflow management systems and their APIs in order to provide them with the context-related information. Using the configurations provided by the OMS Configuration Manager, it is possible to create a specific mapping between the source OMS context data attributes and destination data structure defined by the specific third-party integration API. Additionally, the OMS system using the same BPM Exchange Module, can be subscribed to any external third-party system event that can be used internally for further processing (creating alarms, raising notifications, triggering some action, etc.).
OMS Architectural View Conclusion

The OMS architecture for the GDi Ensemble for OSS embodies a state-of-the-art design that aligns with industry best practices. Its microservices-oriented approach, coupled with the specialized functionality of each component, positions it as a powerful and adaptable solution for orchestrating service orders in the fast-paced CMT domain. The integration of standardized APIs, such as the TMF 641 Service Ordering API, ensures interoperability and compatibility, making the OMS a cornerstone in the modern landscape of operational support systems.

4. Test Results

Click here to see the test results: GDi-TMF641 API-HTML Results