EOSC Future
eoscfuture.eu

EOSC-IF Order Management System: Interoperability Guideline

The EOSC Future project is co-funded by the European Union Horizon Programme call INFRAEOSC-03-2020, Grant Agreement number 101017536

Version 1.0
09 2023

EOSC-IF / Order Management System

Lead by **ACC Cyfronet AGH**
Authored by Jakub Sawicki (ACC Cyfronet AGH), Roksana Wilk (ACC Cyfronet AGH), Agnieszka Pułapa (ACC Cyfronet AGH), Paweł Gorczyca (ACC Cyfronet AGH), Katarzyna Lechowska-Winiarz (ACC Cyfronet AGH)
Reviewed by John Shepherdson (CESSDA)

Dissemination Level of the Document

Public

Abstract

The document describes the current architecture of the Marketplace Order Management System and provides guidelines for integration options for the providers and communities.

Version History

Version	Date	Authors/Contributors	Description
V0.1	08/09/2023	Jakub Sawicki (ACC Cyfronet AGH), Roksana Wilk (ACC Cyfronet AGH). Agnieszka Pułapa (ACC Cyfronet AGH), Paweł Gorczyca (ACC Cyfronet AGH)	Initial version, technical information
V0.2	11/09/2023	Agnieszka Pułapa (ACC Cyfronet AGH)	Review
V0.3	12/09/2023	Agnieszka Pułapa (ACC Cyfronet AGH), Katarzyna Lechowska-Winiarz (ACC Cyfronet AGH)	Definitions added, Minor changes
V0.4	13/09/2023	Agnieszka Pułapa (ACC Cyfronet AGH), Paweł Gorczyca (ACC Cyfronet AGH)	Responded to review comments
V1.0		Agnieszka Pułapa (ACC Cyfronet AGH), Paweł Gorczyca (ACC Cyfronet AGH), Katarzyna Lechowska-Winiarz (ACC Cyfronet AGH), Ron Dekker (TGB), Mike Chatzopoulos (ATHENA)	Final Version submitted to EC

Copyright Notice

Table of Contents

Table of Tables

Glossary

EOSC Future project Glossary is incorporated by reference: https://wiki.eoscfuture.eu/x/JQCK

List of Abbreviations

Acronym	Definition
OMS	Order Management System
API	Application Programming Interface
EOSC	European Open Science Cloud
JSON	JavaScript Object Notation
HTTP	Hypertext Transfer Protocol
SOMBO	(EOSC) Service Order Management Back-Office

1 Intended Audience

The intended audience for the Order Management Interoperability Guideline are providers and communities who would like to integrate with the Marketplace Order Management on different levels.

2 Description and Main Features

The API of the EOSC Marketplace enables integration in two areas: offering and ordering.

Offering

Exposes functions that are necessary to manage offerings, including their technical parameters and ordering configuration. Its OpenAPI Specification is available here.

Ordering

This is an API that enables integration with the ordering process. External Order Management Systems (OMSes) can use it to keep order processing on the provider or community side, while still providing users with a consistent order workflow and support.

The Offering API is accessible for data administrators regardless of the existence of the OMS. The Ordering API is available after registering an OMS in the marketplace.

Access model

OMS has relations with data administrators, which designate who can manage it. Data administrators can create a provider's group or resource dedicated OMSes. Data administrators can choose an OMS for an offer and the available options include: global OMSes, provider's OMS and a service dedicated OMS.

An OMS can have two roles in the context of project:

- mediator OMS: the default OMS
- provider OMS: every OMS, which is a primary OMS of a project's project item

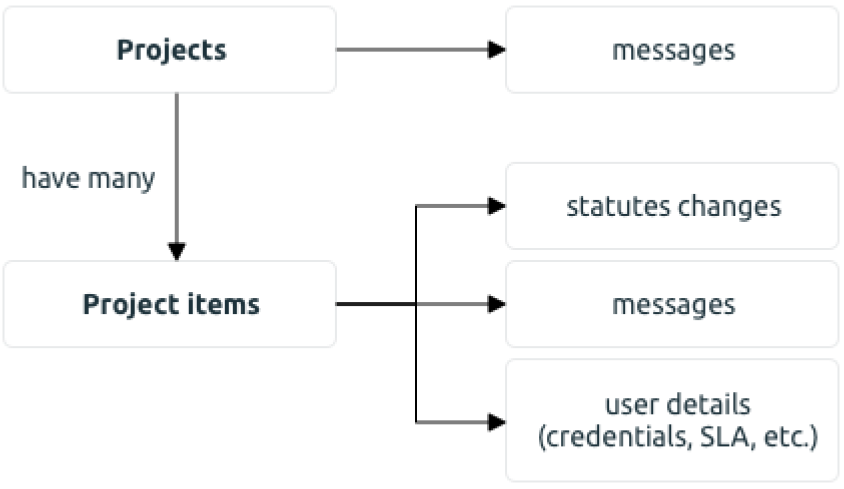
An OMS can have two roles in a context of the project item:

- primary OMS: either OMS selected by a data administrator, or the default OMS
- secondary OMS: the default OMS

Every OMS with a role in project or project item has read access to all messages (*user_direct?* with obfuscated message).

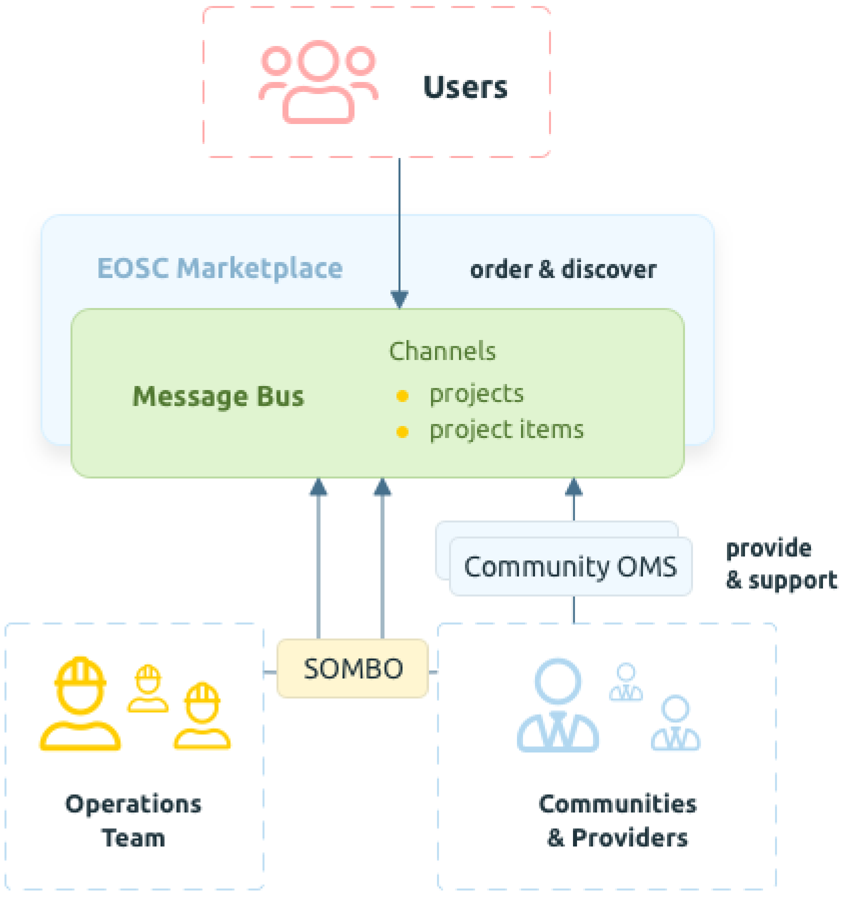
Data model

In the EOSC Marketplace, every user can create Projects to organize their actions and get guidance from the operations team. Within a Project, the user will create service orders and add open-access resources. Both are represented as a single resource type in the API: the Project Item. A Project Item's status and user details are mutable by an owning OMS. In the scope of both a Project and a Project Item, a conversation between parties is held as Messages associated with either a Project or a Project Item.



Data model relations

Below, we present an overview of the EOSC Marketplace APIs in the scope of interoperability patterns regarding order management. Users are presented with a consistent EOSC Marketplace user interface, while communities, providers, and operations teams can leverage the APIs to achieve a high interoperability level and provide value to the users in a way they see fit.



Communication flow in the EOSC Marketplace

Trigger

In order to receive asynchronous notifications about new and changed resources, an OMS needs to expose a HTTP trigger endpoint. HTTP (GET, POST or PUT) requests will be issued on this endpoint each time the OMS needs to be notified of any changes.

A trigger call is issued with each change to resources (Projects, Project Items, Messages) that the OMS has access to.

After handling a trigger call, the adapter should send a request to the GET /events endpoint for updates.

Request and response format

The API expects requests in JSON format and returns JSON too. Its content-type is application/json.

Authorization

Authorization tokens are used to authorize both APIs. Such a token can be retrieved from the EOSC Marketplace API documentation and will be similar to: *ios_Bg6L1hsvDyvfYK_C*. In the case of the Ordering API, such a token will be issued upon OMS registration. To provide integration support for other software systems the token is long-lived, but can be revoked and regenerated. To do this for the OMS token, one needs to contact support (using : cyfronet-support@mailman.eosc-portal.eu). To authenticate your HTTP requests, set the X-User-Token request header.

For example: curl -H "X-User-Token: ios_Bg6L1hsvDyvfYK_C" [...]

3 Response to Community Need

Integration becomes necessary when users must seek permission or make payments for resource usage, or when resources require configuration to grant access. Resources that necessitate ordering can be linked with the EOSC Order Management System.

Orders can be administered in two ways:

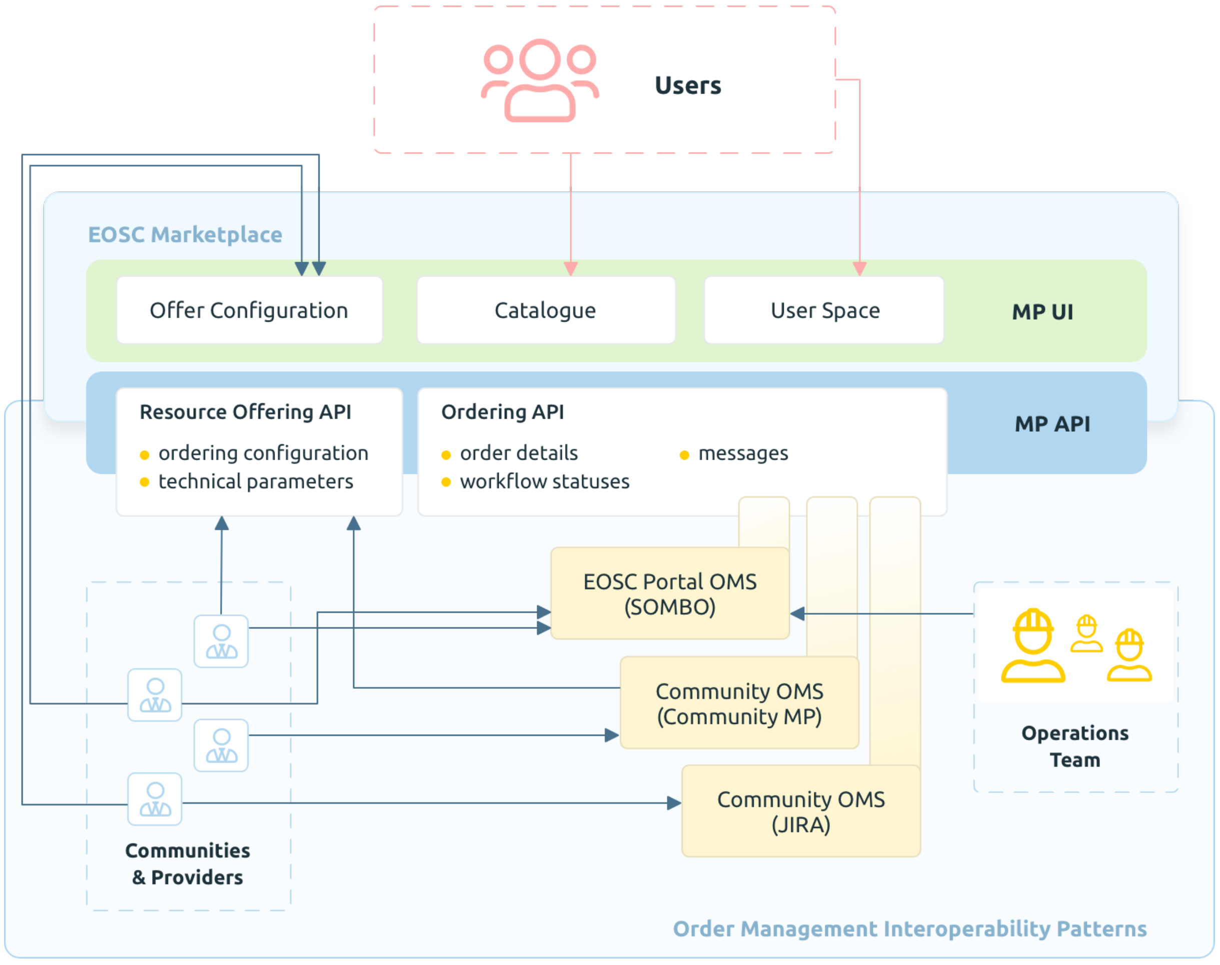
- Through the EOSC Order Management System.
- Through providers' own channels.

Consequently, providers with their order management systems aren't obligated to integrate with the EOSC Order Management System. Nevertheless, it is advisable to do so, as it offers greater convenience to users who can then place orders directly from the EOSC Marketplace.

By fully integrating with the EOSC Order Management System, one can avoid the need to develop an independent order management service (if you don't already have one). The resource's maturity will be enhanced. The end-user experience will be enriched:

1. Users will receive notifications regarding the status of their orders via the EOSC Portal.
2. Managing all orders will become more straightforward for end users, as they can do so using a consistent interface without leaving the EOSC Portal.

4 High-level Service Architecture



Architecture of the ordering process

Integrators (providers and communities) can choose from a range of integration levels, adding flexibility. In a minimal setup, a provider can use the Marketplace UI to configure their offerings and use SOMBO for order management, avoiding any implementation and integration costs. On the other hand, one can choose to use an in-house system that fully integrates with EOSC Marketplace APIs both for offering and order management. Existing community systems can be extended this way, exposing the communities to EOSC users while maintaining the same set of tools for operations. Users can check project items statuses and exchange messages with the operations team and service providers through the Marketplace UI. Communities and providers should provide an OMS endpoint to push a trigger for updating data. This allows project item information to be kept up to date. The Offering API enables the management of offers, which are objects of customers' orders. The Ordering API keeps all the information about orders regarding OMS connected to the provider or a community.

5 Definitions

EOSC Order Management is a set of services, processes and guidelines to enable providers to integrate ordering and procurement of their EOSC Resources. Users, on the other hand, can then enjoy a more integrated user journey from resource discovery to procurement, within a coherent EOSC Portal. It provides users who order EOSC Resources with a more uniform way through the ordering process. When a user aggregates resource orders (project items) within a project, it also enables inter-provider communications, to enhance composability of resources and increase outreach. Both users and providers also gain support from the EOSC Portal Operations Team in this way. The EOSC Marketplace is the central part that facilitates the ordering processing and provides various ways to interface with it.

6 Licensing Information

The Offering and Ordering APIs are compatible with OpenAPI under the Apache-2.0 license.

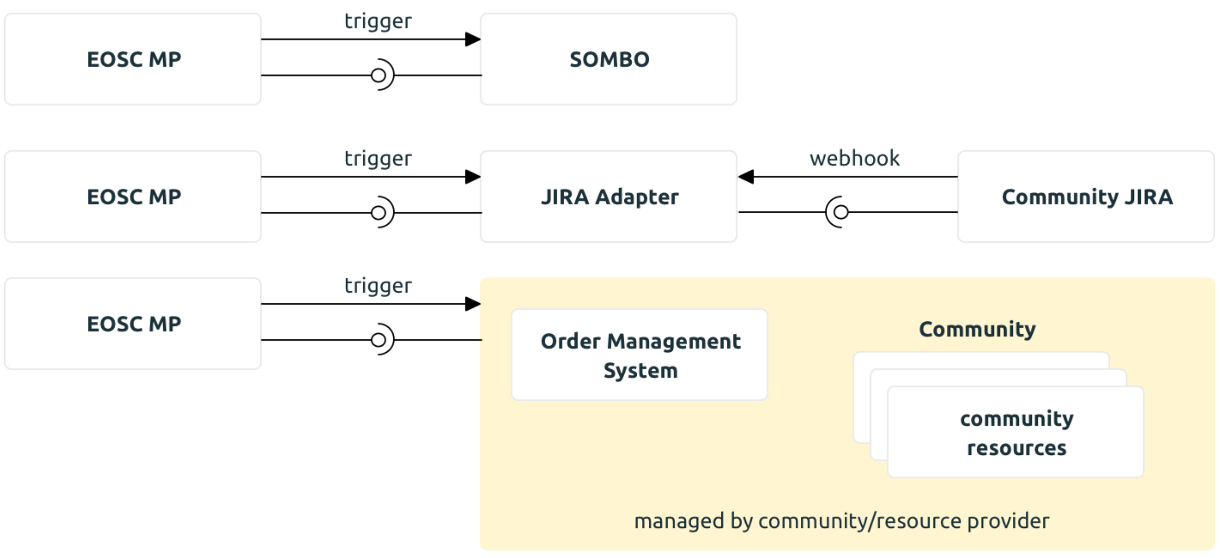
7 Related Standards

Table 9-1: Related Standards

Title	Short Description	relatedStandardIdentifier
OpenAPI v3.0.1	Both the Offering and Ordering APIs are compatible with the OpenAPI standard	https://github.com/OAI/OpenAPI-Specification/blob/3.0.1/versions/3.0.1.md

8 Integration Options

The Ordering API is meant to support integration of external OMSes.
The interoperability layer is based on the Ordering API with asynchronous triggers that can be accessed by interested parties.Integration Options are presented on the picture.



Integration Options

9 Interoperability Guidelines

Integration steps

In order to integrate your ordering process with the EOSC Marketplace, you have to follow a few steps:

- Obtain an OMS registration in the test infrastructure,
- Connect your instance of an OMS adapter (you can use the reference Jira implementation for this).
- Switch to the production environment.

1. OMS Adapter (General)

The OMS adapter functions as a way to generalize communication with ticket management systems. The reference implementation supports integration with JIRA. But any integrator can develop its own compatibility layer with their current ticketing system.

The EOSC Marketplace has a very simple way of communicating with the Adapter (trigger). Everything else is done proactively by the OMS adapter itself. Usually the ticket management system with which integration occurs informs the OMS adapter about some changes, then the OMS adapter calls the Marketplace API. When any change occurs in the Marketplace (new Project / Project Item is created, Message is posted) the Marketplace triggers the OMS adapter (every registered OMS adapter has a corresponding trigger configuration) and the adapter then makes any necessary calls to the ticketing system to enact that change.

It is in the scope of the OMS adapter to store mappings between Marketplace's project item IDs and ticketing system IDs.

2. Reference Adapter (JIRA)

The provided reference implementation contains a Marketplace API client which is responsible for communication with the Ordering API. It also contains a thin translation layer which translates Project Item parameters to fields in JIRA. The core philosophy enshrined in the system is that it should not, under any circumstances, lose information about ordering changes. To fulfill this purpose all actions are scheduled to the Distributed Task queue, this way they can be retried if network errors occur, or in case they fail they can be reviewed manually by the operator.

Access model

The following applies to the Ordering API.

1. OMS roles

An OMS can have two roles in the context of a Project Item:

- primary: if selected by the EOSC Resource provider to handle that offering orders,
- secondary: the default OMS.

In case the primary OMS wasn't selected, then the default OMS is considered both as primary and secondary OMS.

2. An OMS can have two roles in the context of a Project:
   - mediator: the default OMS,
   - provider: every OMS, which is a Primary OMS of any Project Item of the Project.

- Read access

If an OMS has any role in a Project or Project Item, it can read it. The Project Item's field *user_secrets* will have obfuscated values (keys will be visible).

Every OMS with a role in a Project or Project Item has read access to all associated messages. However, the content of messages with *user_direct* scope will be obfuscated. Messages with internal scope will not be disclosed to users.

- Write access

A Project cannot be changed via the API. A Project Item can be changed by its primary OMS.

Messages have fields *scope* and *role*, which determine their write access. The scope can have values *public*, *internal* or *user_direct*. The role can have the following values: *user*, *provider* or *mediator*.

The write permissions for different combinations of scope and role are as follows. They also differ slightly for Project and Project Item messages.

scope \ role	**user**	**provider**	**mediator**
public	false	is provider OMS?	is mediator OMS?
internal	false	is provider OMS?	is mediator OMS?
user_direct	false	false	is mediator OMS?

Permissions for specific roles in the message flow