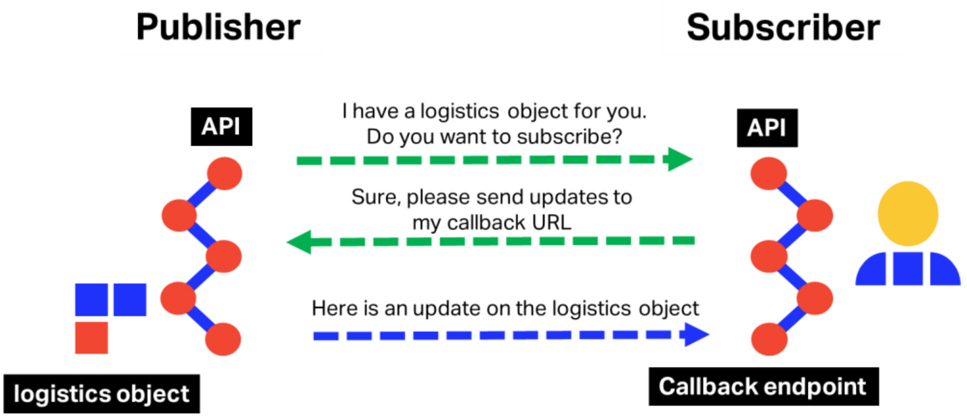
Automating data notifications in ONE Record

Publish & Subscribe in a distributed logistics network

In distributed applications, components of the system often need to provide information to other components as events occur. For example, companies need to be notified when new data becomes available, so they can act accordingly if required.

ONE Record proposes a Publish & Subscribe mechanism to allow for a distributed network of ONE Record compliant platforms. Companies can publish Linked Data that participants of the Internet of Logistics can read and subscribe to and subscribers get notified as soon as new data becomes available. This document describes the Publish & Subscribe model used in ONE Record as well as its basic requirements.


Publisher
Subscriber
API
API
I have a logistics object for you. Do you want to subscribe?
Sure, please send updates to my callback URL
Here is an update on the logistics object
logistics object
Callback endpoint

Pub/Sub high-level flow

The need to Publish & Subscribe (Pub/Sub)

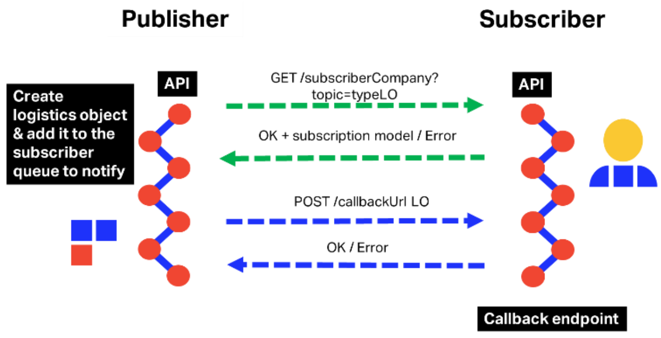
Enabling companies to share their data securely using common ontologies and allowing them to actively push updates to connected business partners is one of the main goals of the ONE Record standard. Authentication and authorization techniques are also included in ONE Record in order to guarantee secured access to the published data.

Each participant on the Internet of Logistics can act as a **source of data** (publisher) and/or a **consumer of data** (subscriber). A publisher must be able to inform all subscribers that data is available or that it has changed as soon as possible.

Pub/Sub implies that the publisher of a Logistics Object (LO), contacts the intended recipient (the subscriber) and proposes that they subscribe to a topic, that in ONE Record defines a type of LO (e.g. Booking, Shipment, etc.). The subscriber can subscribe by providing the publisher with a call-back URL where the publisher can send notifications about LOs of the given topic, such as LO creation or updates. When the subscriber receives a notification on the previously provided call-back URL, they can automatically take action if needed.

Pub/Sub flow in ONE Record

There are several approaches to Pub/Sub, but the recommended one in a ONE Record environment is the **push** approach. Push approach means that instead of asking the publisher for changes in a regular interval (pull), the subscriber is actively notified about any changes (push). This approach avoids the publisher to receive too many pull requests from possible subscribers.


Publisher
Subscriber
API
API
Create logistics object & add it to the subscriber queue to notify
GET /subscriberCompany?topic=typeLO
OK + subscription model / Error
POST /callbackUrl LO
OK / Error
Callback endpoint

Pub/Sub API operations

Step 1 – Publish an LO

The publish action occurs when an LO is created or updated on a ONE Record Server. At this stage the LO is accessible via the Server API to authorized companies.

Step 2 – Retrieve Subscription information from companies to give access to

The second step is retrieving the subscription information from the companies you want to give access to this LO. To achieve this, the company publishing the LO must check with each of the companies it wants to give access to, whether they subscribe to these LO types (topics). If they do, they provide the details of the endpoint where the LO updates should be pushed to.

The prerequisite to this is that the companies must know each other through a previous exchanged Company Identifier so that the machines can ask this question during operation. These Company Identifiers may also be retrieved from common or local directories.

Step 3 – Notify the subscribers

Once the subscription information is received, the publisher will push the LO URL to the intended subscribers using the details provided. If the subscriber was not available at the time, then the publisher would need to queue and retry to push the LO over a certain time.

Note: In Pub/Sub, publishing parties need to store a list of all the parties subscribed to their LO topics in their backend systems.

Pub/Sub basic operations

Publish: The publisher creates a LO or publishes changes to an existing one.

Subscribe: An interested entity (subscriber) sends subscription information containing a callback URL to the owner of the data.

Notify: The publisher looks up all subscriptions and notifies interested subscribers of a topic by sending them the URL of the newly created/updated LO of that topic. The subscriber needs to be successfully authenticated and authorized in order to read published data.

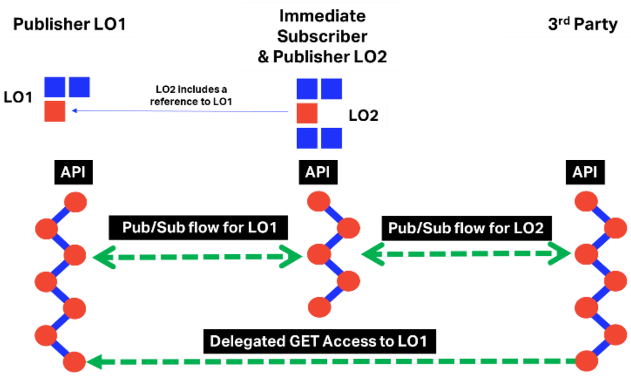
Pub/Sub topics and guaranteed delivery queue

Data is exchanged between applications using a notion of **topics** and **delivery queues**. While in transit, data is kept in message queues that ensure integrity and availability of the system. Should a subscribing application go down, messages are safely retained by the publisher until the recipient is ready to read them again.

For each subscriber and each topic, a **message queue** is maintained automatically by the publisher to keep data in, until the subscriber confirms it has received a certain notification.

Pub/Sub in multi-party scenario

By definition, Linked Data is interlinked with other data, and the ownership of the different levels of data can be distributed throughout the Internet of Logistics. One of the challenges encountered while building a Pub/Sub solution for ONE Record is to make sure that when there is an update on an LO, all the partners which are subscribed to other data that are linked to this object receive the update.


Publisher LO1
Immediate Subscriber & Publisher LO2
3rd Party
LO1
LO2 includes a reference to LO1
LO2
API
API
API
Pub/Sub flow for LO1
Pub/Sub flow for LO2
Delegated GET Access to LO1

Pub/Sub for multi-party

As a solution, in ONE Record Pub/Sub only exists between the publisher and its immediate subscribers. In the multi-party scenario, third parties would get delegated GET access from the publisher as initially requested byits immediate subscriber. Each party in chain starting from the immediate subscribers is then responsible for updating the next party in the chain, through a Pub/Sub relationship that they will have with their downstream partners.

You can find a detailed use case in the ONE Record API & Security specifications on GitHub.

Conclusion

By enabling companies to share their data securely using common ontologies and allowing them to actively push updates to connected business partners, they are able to improve data exchange in complex supply chains.

ONE Record standard proposes a mechanism based on Publish & Subscribe to tackle the distributed approach of sharing updates of Linked Data.

More info at https://www.iata.org/one-record/.