Bridging the Internet of Logistics with the ONE Record API

How APIs can drive next-gen integration across the supply chain

Contents

04 Foreword

06 APIs at the Core of Digital Cargo Transformation

07 Overview of the ONE Record API Features

08 Handle Change Requests with PATCH

10 Share Data Downstream with Access Delegation

11 Automatic Data Notifications with Pub/Sub

14 Manage Access Control to API Operations

15 Discover Prior Versions of a Resource with Memento

17 What ONE Record API Can and Cannot Do

19 Takeaways & Next Steps to Take from Here

22 Enquiries and Feedback

Foreword

Every piece of software built today either uses an API or is an API. APIs – application programming interfaces – have become an essential element in today's digital ecosystem, and this fact is reflected in the API growth rate which is increasing at an explosive pace[1]. Most of modern consumers' day-to-day digital experiences – from purchasing goods online to searching for a video to watch in streaming – would not be possible without APIs working in the background and allowing machine-to-machine communication.

Every industry can benefit from APIs. Nevertheless, many industries are not yet taking full advantage of the power of APIs, as some of them are adopting APIs more slowly than others. One of the "slow adopters" is the air cargo industry, that has just slowly, but surely, entered the age of API.

"There have been many IT buzzwords over the years, but "API" is not one of them. Sharing data through modern technologies like APIs not only helps untangle the complex spider web of information exchange but it also sends a strong signal that you have a desire to make things easier for both customers and employees."

Jonathan Parkinson
Senior IT Advisor, Air Canada Cargo

[1] **https://blog.postman.com/api-growth-rate/**

Air cargo companies need to share and access different types of data such as bookings, documents, addresses etc. To facilitate this data sharing, IATA introduced ONE Record[2].

ONE Record is a standard for air cargo data sharing, aimed at creating a single record view of a shipment. It defines a common model for the data that is shared via standardized and secured web API. This global standard allows air cargo airlines, shippers, forwarders, ground handlers, and all the other entities from the supply chain to interact freely and in real-time, to transmit data including rich content, and to get instant notifications and acknowledgment. It aims at transforming the air logistics business to an extent that seems unconceivable in today's cargo world, which is based mostly on legacy systems and physical paper processes.

ONE Record is based on three pillars: Data Model, API and Security. These three pillars are interdependent and together they ensure that ONE Record compliant solutions provide all the needed semantics, interactions and security that cover air cargo processes.

This White Paper introduces the ONE Record API and arguments how Linked Data[3] based APIs can be beneficial by connecting disparate entities into a powerful network – the Internet of Logistics – that ultimately offers countless possibilities for digital transformation and continued innovation. It also presents the ONE Record API features and how they mirror digitally the existing interactions in the air cargo ecosystem.

[2] **https://www.iata.org/en/programs/cargo/e/one-record/**
[3] **https://en.wikipedia.org/wiki/Linked_data**

APIs at the Core of Digital Cargo Transformation

Air Cargo has been around for more than a century now. From the earlier eras, air cargo is based on old-fashioned, sequential, paper-based processes through which a large quantity of information is exchanged between actors of a complex supply chain. This practice engenders many limitations including poor data quality and lack of transparency.

Digitalization is currently the best method to improve these information processes and ensure that goods and information flows are optimized. The role of API is here – to effectively manage the supply chain for the cargo ecosystem, to minimize the amount of human interaction necessary and make the flow seamless.

Most companies have typically chosen APIs to be the essential technology to connect devices to on-premise servers and middleware, but also to connect to other partners. Nowadays, most businesses have switched to APIs, while a large majority of air cargo companies – forwarders, airlines, ground handlers – still use old electronic data exchange technologies to share data. Although old technologies and legacy systems served well during all these years and can still "do the job", they are too rigid to support new business needs in a dynamic and fast-changing digital environment.

In contrast to legacy systems and messaging standards, APIs are easy to develop and maintain, they are based on technology standards that are programming language agnostic and enable connected communication. With the rising demand for transparency and scalability, these API characteristics are not only necessary, they are also crucial in providing a way to address the growing needs of the air cargo industry focused on interoperability in a connected and collaborative ecosystem. Unfortunately, only few air cargo stakeholders have been using APIs for data sharing with their partners to date.

ONE Record

In order to pave the way for more efficient, transparent and tech-savvy air cargo business, IATA introduced the ONE Record standard, specified by making use of technologies such as ontologies and APIs and meant to bring air cargo into the digital era.

One of the main aims of ONE Record is to define connectivity between air cargo players through APIs, that ultimately serve as a bridge connecting all the supply chain entities into the Internet of Logistics.

Getting the Gist of APIs

API stands for Application Programming Interface and describes a set of requirements that govern how certain software components, layers or applications should communicate to another.

In practice, an API is a set of programming instructions and standards for accessing a web-based software application or tool. Nowadays, modern APIs can connect in real-time with near zero delays between request and response.

It is essential to mention that an API is a **software-to-software interface**, not a user interface. With APIs, applications talk to each other without any user knowledge or intervention. For example, when you purchase a book online and enter your credit card information, the online bookstore uses an API to send your credit card information to a remote application that verifies whether your information is correct. Once the payment is confirmed, the remote application sends a response back to the bookstore website saying that it is OK to validate your purchase.

The user can only see a single interface – the online bookstore – but behind the scenes, many systems are communicating to each other using APIs.

Overview of the ONE Record API Features

Entities of the cargo supply chain need to share and access different types of data such as bookings, ULD identifiers, documents, addresses, etc. In ONE Record, these are referred to as **Logistics Objects**. As ONE Record uses the Linked Data concept for its specifications, each Logistics Object is unique and identified through its own URI[4] (Uniform Resource Identifier). Therefore, in order to retrieve the data related to a Logistics Object through the ONE Record API, the client only needs the URI of that particular object.

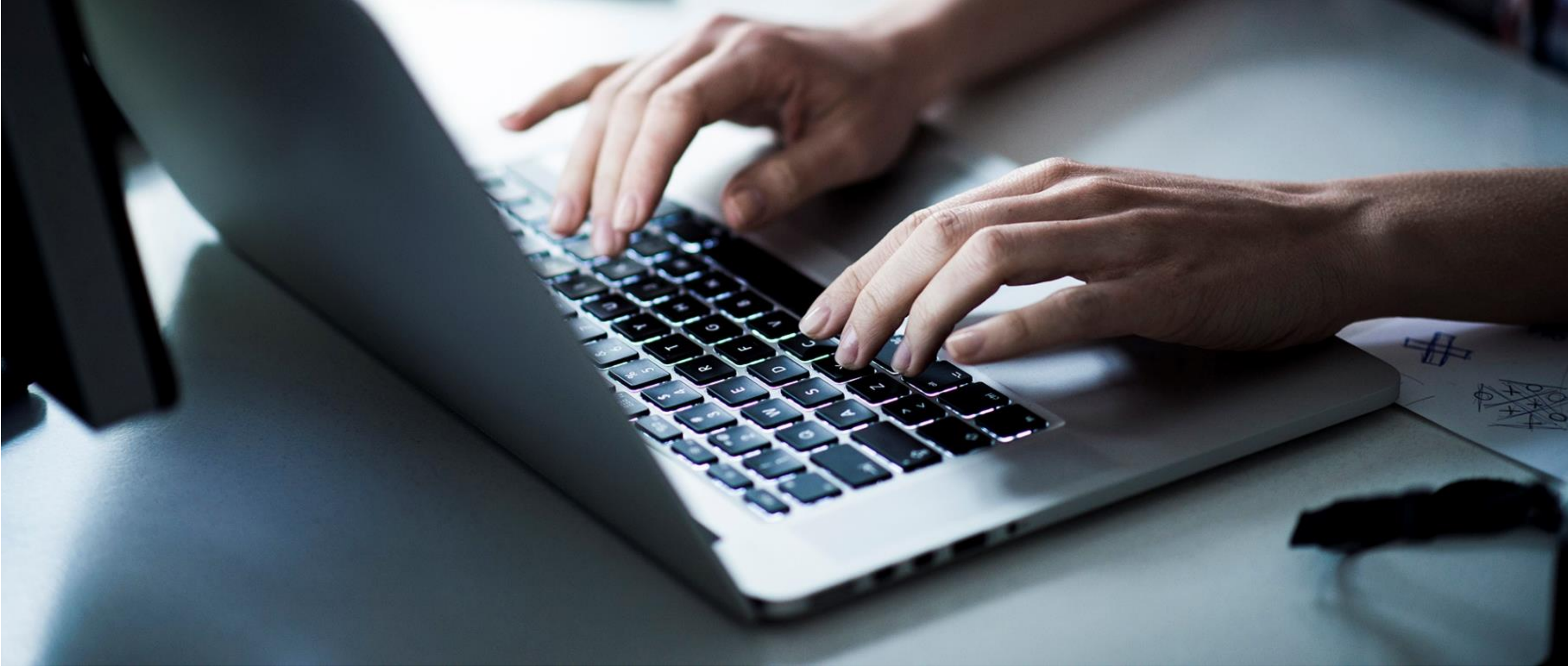
There are three HTTP operations to interact with a Logistics Object that is hosted on a ONE Record server in the Internet of Logistics: POST, GET and PATCH. The format supported by these three operations is JSON-LD[5].

Creating data with POST

When creating a new Logistics Object, the client must perform an HTTP POST request. The data for the Logistics Object to be created should be included in the request body and provided that the client is authenticated and authorized, the server will accept the request and create a new Logistics Object. The creation will be generally effectuated by the owner of the data, who in most cases owns or at least controls the ONE Record server.

[4] https://en.wikipedia.org/wiki/Uniform_Resource_Identifier

[5] https://json-ld.org/



Reading data with GET

To read the content of a Logistics Object, the client needs to perform an HTTP GET request. The accessed server will check that the client is authenticated and authorized before returning the data in the JSON-LD format.

Updating data with PATCH

To make a change to data of a Logistics Object, the client needs to send a change request via the HTTP PATCH method. In ONE Record, the PATCH request must specify the type of operation to apply to the data element (add, change, or delete) and the path to the data element to be changed. The owner of the Logistics Object will then decide whether they will accept the change request or not. All PATCH operations are persisted in an audit trail so that all parties can trace back the history and changes that concern a specific Logistics Object.

More features of the ONE Record API

In addition to the HTTP operations on the Logistics Objects, the ONE Record API specifies further functions such as:

- Access delegation;
- Automatic data updates through Publisher & Subscriber;
- Access control;
- Snapshots of data at a certain moment in time with Memento.

These features will be explained more in detail in the following sections.

Handle Change Requests with PATCH

As mentioned before, in the air cargo ecosystem, companies need to share, access and update multiple types of data such as bookings and contacts, data referred to as Logistics Objects. ONE Record compliant APIs support three ways to interact with a Logistics Object via POST, GET and PATCH HTTP methods.

This section presents an overview of how the HTTP PATCH operation can be implemented in a Linked Data environment, based on practical considerations from feature-driven development of the ONE Record standard.

Why use HTTP PATCH Method?

PATCH is a concept derived from the common understanding of the word "patch". A patch is basically a description of differences between two states of a resource.

The PATCH method is a request method supported by the HTTP protocol for making partial changes to an existing resource.

In ONE Record, an authorized client can only request partial updates of a Logistics Object and not request to replace the entire data, hence the choice of using HTTP PATCH[6] instead of HTTP PUT[7]. When updating a single field of a Logistics Object, sending the complete object via HTTP PUT request might also be heavy and utilizes a lot of unnecessary bandwidth. Therefore, in the ONE Record scenarios, the semantics of HTTP PATCH make a lot more sense.

What is the Linked Data PATCH Format?

As explained in a dedicated white aper, Linked Data describes a method of publishing structured data on the Web so that it can be interlinked and consequently become more contextual and relevant.

W3C has specified a Linked Data PATCH Format[8], a format for describing changes to apply to Linked Data. It defines a list of operations to be performed against a Linked Data resource, namely the addition or removal of RDF[9] triples in a graph representing the resource.

RDF is very simple in its structure and does not require more than add and delete. Also, RDF triples are much simpler than either JSON or XML formats, so PATCH should be easier to implement as there is less functionality required. However, JSON-LD – the data format used for Linked Data by the ONE Record API – is more complex[10][11] than plain-old JSON[12].

How to use PATCH in ONE Record?

In ONE Record API, the PATCH request represents an array of objects. Each object represents a single operation to be applied to the target Logistics Object.

The evaluation of a PATCH request occurs as a single event. Operations are sorted and processed as groups of **delete** and then **add** operations until all the operations are applied, or the entire PATCH fails.

PATCH Operations

add: Add has a simple function, it always adds new sets of statements. If a pre-existing statement exists with similar or the same characteristics, it must not be overwritten. To overwrite, a delete and an add operation must be performed.

del: Del always removes sets of statements.

A few considerations related to PATCH in ONE Record

There are a few points to consider when implementing PATCH in a ONE Record API. These points are adaptations to the air cargo ecosystem.

- Only the publisher can change the Logistics Object, where the publisher is the party that creates the Logistics Object on the ONE Record server.
- A business partner can request a change on the Logistics Object.
- The publisher makes the Logistics Object changes based on previously defined business rules.
- An **audit trail** (history) of all the changes is stored and can be retrieved at any moment from a dedicated endpoint in the API.
- Logistics Objects have a **revision number**, which is an integer to be incremented after every applied change.
- When retrieving a Logistics Object, the latest version should always be returned.

[6] https://tools.ietf.org/html/rfc5789
[7] https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.6
[8] https://www.w3.org/TR/ldpatch/
[9] https://www.w3.org/RDF/
[10] https://www.youtube.com/watch?app=desktop&v=hdT6SH0QzZ0
[11] https://github.com/digibib/ls.ext/wiki/JSON-LD-PATCH
[12] https://www.json.org/json-en.html

- As mentioned before, PATCH operations are sorted and processed as groups of delete and then add operations until the operations are applied, or the entire PATCH fails. No partial updates are accepted.
- As a best practice, a GET Logistics Object should be performed before requesting a PATCH in order to make sure that the change is made towards the latest version of the object.
- If a change request is rejected, the revision number of the Logistics Object is not incremented.

Update Logistics Objects is a crucial functionality of the ONE Record API. The specification for PATCH used in ONE Record is simple and only "does what it needs to do". This approach enables linking entities in an innovative way, and it follows HTTP PATCH definition.

Share Data Downstream with Access Delegation

Data sharing between companies generally requires a form of access restriction in order to make sure that the right company accesses the right data.

One unique feature of transport and logistics chains is their sequential nature. Cargo is moved from the shipper to a forwarder to a ground handling agent to an airline, etc. Each party along this chain has data that needs to be shared downstream and status or clarifications back upstream. The challenge is that in most cases, these parties along the transport chain don't always know each other but do need to access each other's data.

For example, an import customs agent may need information from the originating shipper between whom there is no relationship. Traditionally, each party copies data to the next party in the chain so this issue doesn't exist. With data sharing however, data needs to stay at its source.

In ONE Record, the typical scenario is that the company that creates the data notifies their partner and provides them access details such as the URI of the Logistics Object. However, that second company may need to share the same object with another company downstream.

As an example, an airline that received a booking from a forwarder might need to share that data with their ground handler as well. As a first step, the forwarder created a Logistics Object for the booking and sent its URI to the airline. When the airline performs an HTTP GET request on that Logistics Object URI, the forwarder will usually grant access and return the data to the airline, but only to that airline and no one else. In order to share the booking URI to the ground handler, the airline can use **Access Delegation**.

In the access delegation protocol, a client application asks resource owner for permission to give a third-party access to a protected resource on their behalf. In the previous example, the airline informs the data owner, i.e. the forwarder, that they want the ground handler to access the booking data. If the forwarder approves that request, then the airline will provide the booking URI to the ground handler that will be now able to retrieve the booking data via a HTTP GET request on the forwarder's ONE Record API.

How to revoke access?

In the same way, if the airline from this example wishes to revoke access to a certain Logistics Object from their ground handler, they can make a revocation request to the forwarder and from then on, the ground handler partner would no longer have access to the data.

Understanding the concept of Trust Chains

The concept of companies requesting a delegation of access to their partners can also be used by these partners themselves, who now became third parties. In the example above, the ground handler can request that the forwarder gives access to their interline partner. The forwarder will grant the access on the basis that they trust the airline who has trusted their ground handler who trusts their interline partner.

The chains of trust are based on business partnerships and trust in the transport chain. They ensure that the company who has shared a Logistics Object on a server, always knows who may access this and at any time, it can revoke all or part of the chain of trust.

Automatic Data Notifications with Pub/Sub

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 section describes the Publish & Subscribe model used in ONE Record as well as its basic requirements.

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 contacts the intended recipient (the subscriber) and proposes that they subscribe to a topic, that in ONE Record defines a type of Logistics Object (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 Logistics Objects of the given topic, such as object 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.

Here are the high-level steps in a ONE Record Pub/Sub scenario:

Step 1 – Publish a Logistics Object

The publish action occurs when a Logistics Object is created or updated on a ONE Record Server. At this stage the Logistics Object is accessible to retrieve via the ONE Record 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 Logistics Object. To achieve this, the company publishing the Logistics Object must check with each of the companies it wants to give access to, whether they subscribe to these Logistics Object types (topics). If they do, they provide the details of the endpoint where the Logistics Object 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 Logistics Object URI 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 Logistics Object over a certain time.

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

Pub/Sub basic operations

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

Subscribe: An interested entity (subscriber) sends subscription information containing a call-back 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 URI of the newly created/updated a Logistics Object 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 a Logistics Object, all the partners which are subscribed to other data that is linked to this object receive the update.

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 by its 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.

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's proposed Publish & Subscribe mechanism aims to tackle the distributed approach of sharing updates of Linked Data.

Manage Access Control to API Operations

The main goal of access control is to minimize the risk of unauthorized operations on resources. Access control is a fundamental component of security compliance that ensures security technology and access control policies are in place to protect the data.

In ONE Record, access to resources can be handled by using Access Control Lists (ACLs) stored in the backend systems of the ONE Record Servers and defined using the Web Access Control standard from W3C.

What is Web Access Control (WAC)?

According to W3C[13] WebAccessControl specifications, "Web Access Control is a decentralized system for allowing different users and groups various forms of access to resources where users and groups are identified by HTTP URIs". It enforces access control based on the **Access Control List (ACL)** RDF resource associated with the requested resource. ACL allows access to agents (users, groups, etc.) to perform various kinds of operations (read, write, control, etc) on the respective resource.

How can I define to whom I give data access?

In ONE Record, access to resources can be specified by using Access Control Lists (ACLs) associated to specific Logistics Objects. Each Logistics Object resource possesses a related ACL containing a set of **Authorization** statements that describe:

- **who** has access to that resource;
- **what types** (or **modes**) **of access** they have.

Each Authorization is a single rule for access, such as "*entities one and two may write to Logistics Object logisticObjectReference*", described with a set of RDF properties.

ONE Record recommends the use of the ACL ontology[14] in order to express the Authorizations. As the ACL is specific to each ONE Record Server and it is not a mandatory requirement to make it available to external entities, any other kind of data model/ontology can be used instead.

Given an URI for an individual Logistics Object, a ONE Record Client can discover the location of its corresponding ACL by performing a GET request and parsing the **rel="acl" Link** header.

Access Control in ONE Record

ACL Ontology from W3C could be used

Each ONE Record Server decides if it shares its ACL externally

The link to the ACL should be returned in the Link header when performing GET Logistics Object

[13] https://www.w3.org/wiki/WebAccessControl

[14] https://www.w3.org/ns/auth/acl

Note: ONE Record Clients must not assume that the location of an ACL resource can be derived from a Logistics Object URI.

What types of Authorizations can be defined?

ONE Record recommends the definition of three types of Authorization:

1. **Single Authorization –** when a single company identifier from the Internet of Logistics has access to the Logistics Object;
2. **Group Authorization –** when a group of company identifiers has access to the Logistics Object. The ONE Record Server can define internally groups of access such as Airlines, Ground Handlers, Customs, etc.
3. **"Public" Authorization –** when every authenticated company identifier accessing the Logistics Object URI can retrieve the data.

What modes of access ONE Record recommends?

ONE Record specifies three modes of access on LOs:

READ / **GET**
Read the contents (including querying it)

WRITE / **POST** and **PATCH**
Write contents or modify part of it

CONTROL / **GET**, **POST** and **PATCH**
Read and Write

Discover Prior Versions of a Resource with Memento

In ONE Record environment, linked data resources are updated in real time and generally only their latest state is available for retrieval. In consequence, supply chain stakeholders often need to snapshot a current state of data, for example for a Shipment Logistics Object, and they need to know which version of data was used for that snapshot. Every time a snapshot is triggered successfully, a new version entry is created by the versioning service on the ONE Record Server.

There are few existing studies and proposed solutions for versioning linked data resources. This section introduces the approach used by the ONE Record standard based on the Memento Protocol.

What is Memento Protocol?

Memento is defined in the IETF RFC 7089[15] as an implementation of the time dimension of content negotiation[16], as defined by Tim Berners Lee in 1996. Memento Protocol aims to bring time-based access to Web resources using HTTP capabilities.

Essentially, Memento is an attempt to permit users to view any Web resource as it looked like on a given date in the past.

ONE Record uses Memento Protocol concepts for accessing versions of stored linked data resources.

[15] https://tools.ietf.org/html/rfc7089

[16] https://en.wikipedia.org/wiki/Content_negotiation

What are the components of Memento Protocol?

Memento Protocol defines four types[17] of Web resources:

Original Resource

An Original Resource is a Web resource for which we want to find a prior version.

In ONE Record, the Original Resource is a linked data resource – a Logistics Object – published on a ONE Record Server in the Internet of Logistics. For any given Original Resource several Mementos may exist, each one reflecting a frozen prior state (snapshot) of the Original Resource.

Memento

A Memento is a Web resource that represents a prior version of the Original Resource, i.e. that encapsulates what the Original Resource was like at some time in the past.

In ONE Record, a Memento contains a snapshot of the data at a certain moment in time.

TimeGate

The TimeGate is a resource providing access to prior states of the Original Resource using datetime negotiation. It "decides" on the basis of a given datetime, which Memento best matches what the Original Resource was like around that given datetime.

When negotiating with the TimeGate, the ONE Record Client uses an **Accept-Datetime** header to express the desired datetime of a prior/archived version of the Original Resource it wishes to retrieve. The TimeGate responds with a **HTTP Link header** containing the location of a matching Memento of the resource closest to the datetime sent by the client in the request header.

The response will also include a **Memento-Datetime** header informing the ONE Record Client about when the snapshot of the resource was taken. If the resource did not yet exist at that time, the server will respond with the **appropriate HTTP 404 NOT FOUND** status code.

TimeMap

A TimeMap is a resource that lists links to all of the stored states for a resource along with their timestamps: The Original Resource itself, its TimeGate, as well as its Mementos alongside with their timestamps. In a nutshell, TimeMaps are exposed by systems that host prior versions of Original Resources and allow for batch discovery of Mementos. Original Resources, TimeGates and Mementos can make TimeMaps discoverable by providing an HTTP Link header with a relationship type of "timemap".

Why use Memento Protocol?

Versioning of Logistics Objects is important for ONE Record as it legally binds data at a given moment in time.

An important advantage in using Memento Protocol in the ONE Record environment is that it closely mirrors linked data principles by using HTTP and content-negotiation for version retrieval. It is a well-documented solution which offers excellent discoverability through clear references and it can be easily integrated with existing linked datasets. Because Memento Protocol specifies different categories of resources, ONE Record can provide a distributed architecture by enabling prior version access for original resources hosted on different servers served by separate hosts.

[17] **https://mementoweb.org/guide/quick-intro/**

What ONE Record API **Can** and **Cannot** Do

As explained previously in this document, APIs are essential to digital transformation, and most of our digital experiences today are powered by APIs. Therefore, the expectation is often that once a company starts implementing and using APIs, the digital transformation is simply going to happen. Of course, that's not necessarily the case, and the purpose of this section is to explain some of the useful things where APIs and specifically the ONE Record API can play an essential role, and where they are necessary but sometimes not sufficient.

Improve process efficiency

Switching from paper based or legacy fragmented processes to ONE Record can drive cost reductions by improving or removing existing inefficient or manual processes. Nevertheless, it is up to each company to integrate the ONE Record API with existing legacy systems and to map existing data to ONE Record data models.

Single source of truth

In ONE Record API, a single Logistics Object can be identified with one URL for all clients. This enables to share various objects URLs between different parties and makes the access very simple to the data to the authorized business partners. The single source of truth ensures that companies are operating based on standardized, relevant data across the supply chain. Without a single source of truth, data exists in siloes and each company becomes a sort of black box.

A challenge derived from the existence of the single source of truth is that the company that creates and owns the data must ensure that it is always available when a partner tries to retrieve that data.

Data on-demand

Logistics Objects being available for retrieval via a unique URL from the ONE Record API allows clients to access the data they need when they need it. The only constraints are to ensure that data is available, that the client has the needed authorization to retrieve that data and that it can translate the model to its systems.

ONE Record API defines the data model to be exchanged, the API endpoints to implement and the security needs, however it is the responsibility of the ONE Record Server to make sure that the data is available for its partners at any time and that it defines a correct access control.

Real-time notifications

ONE Record specifies real-time notifications channel via the Pub/Sub mechanism. Although simple to implement, the Pub/Sub mechanism can raise problems of maintainability and scalability and the parties involved need to make sure that they are always available to send/receive notifications.

Plug-and-Play data integration

ONE Record API is built on the principle of "Plug-and-Play". "Plug-and-Play" makes possible to integrate services and to aggregate data from multiple parties. Interoperability is becoming more and more important, because supply chain organizations deal with dozens of vendors, dozens of business partners and dozens of systems in different geographic locations. It is crucial to make sure that all those systems can work together in order to unlock the potential for the data and to enable collaboration across the entire supply chain.

In an interconnected Internet of Logistics, being able to connect via a standardized API to different partners gives the ability to enrich your data with additional sources and therefore unlock numerous innovation opportunities.

Integrate to the Internet of Things

Nowadays, we hear more and more that the Internet of Things (IoT) is the future as the usage of devices equipped with sensors increases day-by-day in all domains. In air cargo, one of the capabilities that IoT can enable is the full end-to-end tracking of shipments. By making ONE Record API interoperable, secure, scalable and discoverable, we can pave the way towards improving many IoT challenges, such interoperability and creating secure and persistent communication between cloud services and the devices and sensors.

Data: A competitive advantage

Most of the companies from the supply chain have assets like clients, partners, suppliers, etc. All these assets translate into a company's single most precious benefit: data. When used cleverly, data is a competitive advantage in the market. But to play that advantage, data needs to be reliable and accessible in real time.

ONE Record API enables a company to securely expose data, but also can enable better analytics, by giving better visibility of the data from the shipper all the way to the receiver of a shipment.

In a nutshell

APIs are necessary but not sufficient for many important challenges and opportunities in today's increasingly digital organizations. The most important takeaway of this section is that ONE Record takes a holistic view: the API is part of a proposed solution for air cargo digitalization. In order to take full advantage of its capabilities, it should be built together with the ONE Record Data Model and Security specifications and be further integrated to other parties and services.

Use the ONE Record API internally

- Build an ONE Record API inside your company, integrate it with existing systems and open up innovation, agility and flexibility in the existing processes

Connect with partner ONE Record APIs

- Open secure collaboration and innovation with partners from the Internet of Logistics and expand your company's value proposition

Make your API public and attract innovation

- Build and inspire a broad ecosystem of developers able to create new experiences and apps that can make big breakthroughs related to digital cargo

How to use the ONE Record API

Takeaways & Next Steps to Take from Here

APIs have become a common bridge across silos of disconnected data in a way that no other technology could. APIs are not just a buzzword: they are foundational computing interfaces upon which infinite applications and systems can be built. The more the internet expands, the more the world of APIs expands.

APIs are one of the three pillars of the ONE Record standard, which is the core of air cargo digital transformation proposed by IATA. ONE Record leverages APIs and other technologies like ontologies and Linked Data to drive efficiency and the emergence of new services in the cargo environment. The ONE Record API is key to improving supply chain processes and enhancing all the involved parties experience. It provides a channel that allows retrieval of the data when it is wanted by the authorized partners, in a format that is machine-readable.

How to start leveraging the ONE Record API now?

1. Read more about IATA ONE Record standard on IATA website
2. Remember your objectives: define your use cases and needs
3. Implement a ONE Record API or get in touch with an IT provider that delivers ONE Record services
4. Embrace the legacy by integrating the ONE Record API and the existing systems
5. Implement a security layer
6. Find complementary partners: connect to other companies from the Internet of Logistics
7. Start exchanging data

Combined with legacy systems and business assets such as data, the ONE Record API can give the industry the agility needed to incorporate innovations into the ecosystem with a reduced time to market. Companies could shift their focus from fragmented paper and legacy systems-based processes to launching new concepts, incorporating day-to-day customer digital experience into the cargo business. Providing data such as Logistics Objects updates in real time further improves the efficiency and speed of business operations.

In order to step in the new digital era, companies do not need to reinvent the wheel. They can take advantage of existing ONE Record APIs and services and move towards modular, services-based API architecture to create an environment where APIs are Plug-and-Play across internal and external systems – the Internet of Logistics. It takes resources and time to make this switch, but the long-term benefits are definitely worth it.

The best part is that, once the ONE Record APIs are in place, any company can connect to each other and start exchanging data in a format that they both understand. The ONE Record API ensures compatibility with the existing processes through its features such as Pub/Sub, access control, delegation and versioning of data.

"ONE Record brings a welcome shift in the way the Air Cargo industry is aligning itself to leverage the ubiquity of the internet. The transformative capabilities of data sharing with ontologies and interactive cargo with IoT integration makes it an immensely powerful platform for the industry to address exciting new markets in e-commerce, pharma, perishables, live animal shipping to name a few. ONE Record is "always on" like the internet and will open new ways of doing business compared to traditional systems that rely solely on messaging."

Pramod Rao
CEO, Nexshore Technologies

Enquiries and Feedback

For enquiries and feedback about this white paper you can contact IATA at onerecord@iata.org.

Acknowledgments

This white paper is the result of a series of research and development efforts by IATA in collaboration with supply chain partners.

Author

Andra Blaj (IATA)