# Discovering PATCH in ONE Record

## The Linked Data way of updating Logistics Objects

**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.**

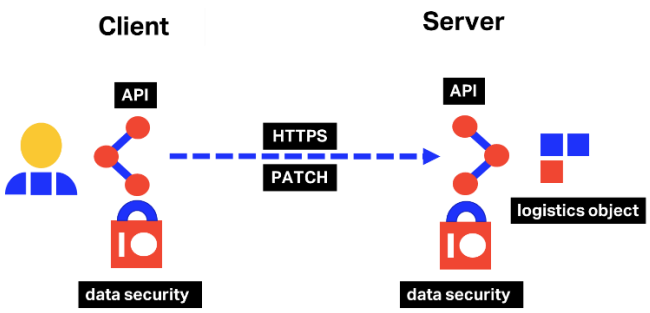
**This document presents an overview of how the HTTP PATCH operation can be implemented in a Linked Data environment, based on practical considerations from feature-driven development of the ONE Record standard.**

## 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 instead of HTTP PUT. 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.




PATCH in ONE Record

## Linked Data PATCH Format

As explained previously, Linked Data describes a method of publishing structured data on the Web so that it can be interlinked and become more relevant.

There are few available examples of implementations showcasing how to develop HTTP PATCH from an RDF or JSON-LD perspective and that take in account RDF particularities such as blank nodes and `rdf:List` manipulations.

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, so PATCH should be easier to implement as there is less functionality required. However, JSON-LD is more complex, as it is not plain-old JSON.

W3C has developed the Linked Data PATCH Format, 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 triples in a graph representing the resource.

The ONE Record API & Security group of experts has based its PATCH proposal on the W3C Linked Data PATCH Format and on the JSON-LD PATCH specifications.

## 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.

The following example contains a list of operations to apply on a Logistics Object. This is what ONE Record PATCH format looks like: it has an operation that can be `add`" or "`del`", a predicate and an object. These concepts are quite familiar to those who work with the Semantic Web.

```
{
   "operations":[
      {
         "op":"del",
         "p":"https://onerecord.iata.org/AirWaybill#totalPieceAndULDCount",
         "o":{
            "value":"10",
            "datatype":"https://www.w3.org/2001/XMLSchema#decimal"
         }
      },
      {
         "op":"add",
         "p":"https://onerecord.iata.org/AirWaybill#totalPieceAndULDCount",
         "o":{
            "value":"11",
            "datatype":"https://www.w3.org/2001/XMLSchema#decimal"
         }
      },
      {
         "op":"add",
         "p":" https://onerecord.iata.org/Airwaybill#date",
         "o":{
            "value":"2019-08-18",
            "datatype":"http://www.w3.org/2001/XMLSchema#date"
         }
      }
   ]
}
```

Fragment of PATCH request

- Operation objects must have exactly one "`op`" (operation) member. This value indicates which operation is to be performed. The value must be one of "`add`" or "`del`"; all other values result in an error.
- Operation objects must have exactly one "`p`", predicate member. The value of this member must be an Internationalized Resource Identifier (IRI) pointing to the path of the element to be changed.
- Operations objects must have exactly one "`o`", object member. The value of this member must an object. This object must contain two members, a "`value`" and a "`datatype`".

### 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.

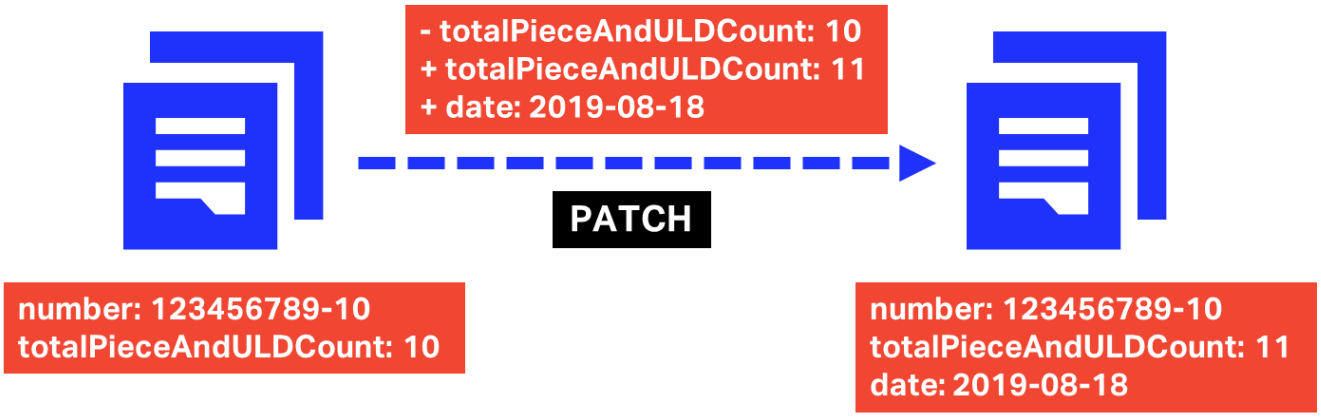
The example below describes the change to be made – delete the **totalPieceAndULDCount** of value 10 and add value 11 instead. Also, a new field – **date** – is added.

## 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 cargo ecosystem.

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

PATCH request in unsuccessful. No partial updates are accepted.




PATCH example

8. 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.
9. If a change request is rejected, the revision number of the Logistics Object is not incremented.

## Conclusion

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.

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