Catch the Wave of Linked Data with ONE Record

Looking beyond the hype for real solutions to real problems

Table of Contents

Welcome to the New Era of Data!

Air cargo transport is a complex industry. Its data exchange is mostly based on peer-to-peer information flow, meaning that each party sends the information to the next one in the supply chain. There is still a lot of paper and manual handling, both of which limit transparency and visibility of shipment and create error-prone environment.

Companies have their own software, applications, databases, spreadsheets and other documents that contain business data. Much of this data is **logically connected** as it relates to the same business components, but is **physically disconnected** which prevents search engines, databases and content management systems from linking up the separate bits and pieces of data.

It is a fact that data is an asset to any business. With continued globalization, the amount of data and its geographic distribution has reached a scale that has never been seen before in the software industry. Therefore, the success of a business relies heavily on its capacity to store, share, manipulate and report on that data.

ONE Record is a standard for data sharing that defines a common data model that is exchanged via a standardized and secured API.

It envisions an end-to-end digital logistics and transport supply chain where data is easily and transparently exchanged in a digital ecosystem of air cargo stakeholders, communities and data platforms.

The ONE Record Data Model specification provides the air cargo industry with a common language and a standard data structure for data exchange based on three concepts: **Semantic Web**, **Ontologies** and **Linked Data**.

> "ONE Record is part of the industry's answer to current global challenges: When implemented, it will improve efficiency, quality and speed. Receiving more data, earlier and in higher quality will enable our industry to be truly data driven and lower the hurdle for start-ups and innovations to thrive. By using big data, predictive analytics and artificial intelligence, we will be able to improve our operational efficiency and offer a more customer centric experience while providing full transparency for all stakeholders throughout the supply chain."
>
> Peter Gerber, CEO, Lufthansa Cargo.

This document explores a different way of looking at Semantic Web from the air cargo ecosystem perspective. It investigates why the usage of concepts as Linked Data and ontologies can be beneficial in a distributed end-to-end digital logistics and transport chain, such as the one that the ONE Record standard enables.

Further on, the document explains how to craft ontologies in a few simple steps. Once you jump in, you will see that it is not that hard. Tech-savvy readers will be delighted to discover technologies like JSON-LD and software development wizards will get some extra magic in the lines of code they write with Object Triple Mapping frameworks.

For some people, this document may be a first step in understanding some of the technical concepts behind ONE Record. The next step would be to get your hands dirty, install an ontology editor, a triple store library and start writing down some code.

Go ahead, admit it: This sounds like a lot of fun!

The Need for Digitalization

The digital air cargo environment today is fragmented, siloed with a mix of old and new systems and technologies. It might seem a paradox that air transport, which is supposed to be the fastest way to ship goods, is slowed down by manual processes and legacy ways of sharing information.

According to IATA, each air cargo shipment accounts for **30 pieces of paper** on average as it makes its way from shipper to consignee, via the freight forwarder, trucking company, terminal operator, airline, ground handler and customs authorities.

7800

tons of paper documents are generated annually by the air cargo industry. It's the equivalent of **80 Boeing 747** freighters filled only with paper.

Despite several initiatives to move towards purely electronic data transfers, such as e-AWB (Electronic Air Waybill), more than 30% of the shipment still needed the paper version of the transportation contract in April 2020.

Digitalization is a crucial transformation for bringing transparency and efficiency in current processes, and for reducing potential errors and delays, leading to overall improved customer service. Technologies like **APIs** and **ontologies** that enable next-level data exchange and allow to connect to different partners from the entire supply chain, using real-time data, are examples of tools that can bring air cargo to the new digital era.

Digitalization opens the way to new technologies like IoT (Internet of Things) and AI (Artificial Intelligence) which will ultimately lead to better customer experience, shipment control, improved processes and opportunities for innovation, enabling the creation of new value-added services for the customers.

ONE Record Data Model: a Digital Twin of Air Cargo

ONE Record is driven by the need for digitalization of the air cargo industry to achieve operational excellence through optimized data sharing and lean business processes while eliminating the use of paper document.

The reasons for using a data model

As ONE Record intends to define a data sharing standard allowing the deployment of a network of plug and play platforms, it is essential to define a standard data model that can be used and understood by any stakeholder in the industry. The data model that has been designed together with a group of industry stakeholders aims to:

- Cover the end-to-end supply chain of goods transportation, including the possibility to integrate multi-modal transportation means.
- Find the optimal balance in simplicity, flexibility and robustness of the data model to ensure that all required data is captured and shared.
- Optimize the usage of modern technologies to facilitate and secure data exchange through APIs, state of the art security mechanisms, etc.
- Optimize data and minimize redundancy.

The pillars of ONE Record Data Model

Four core design principles have been defined to design the ONE Record Data Model.

Piece-centric

As defined by the IATA Recommended Practice 1689, a piece is “a uniquely identified physical single unit which may form all or a part of a shipment”

Physics-oriented: the digital twin concept

A digital twin is the "digital replica” of a physical entity, it can be applied to physical objects, processes, systems, etc. In short, anything that exists in the physical world, as opposed to the digital world, can have a digital twin

One single source of truth

The data stays at the source and clear data ownership ensures its integrity and accuracy, increasing trust in the use of data over paper

Data-driven

The model has been designed around data, not physical documents. All documents, especially the ones that are legally binding, can be easily created with all the required information and the appropriate level of security and trust it implies

Seeing how the data model applies to the air cargo industry

Air cargo is about physically moving goods. It includes physical assets such as the goods being transported or the means of transportation themselves. With this aspect in mind, the data model intends to reflect as much as possible the physical world using the digital twin concept. A digital twin is the "digital replica” of a physical entity and it can be applied to physical objects or physical operations.

Data objects that are essential to the air cargo are called **Logistic Objects**, or “LO”. The ONE Record Data Model focuses on defining these LO and how they interact with each other:

Physical assets are explicitly transcribed, for instance: pieces, shipments, ULDs or transportation means.

Paper documents such as the *Master Air Waybill* or the *House Air Waybill* are also represented by a dedicated *Waybill* object.

The Power of Ontologies

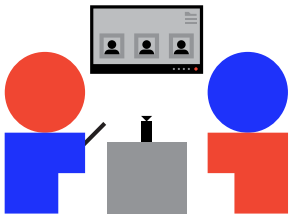
Ontologies are frameworks for representing knowledge about concepts across a domain and the relationships between them. The ONE Record standard takes full advantage of their ability to describe relationships and their interdependence in order to model high quality, linked and coherent data.

Before deep diving into ontologies, let’s take a step back and trace the root concepts behind ontologies: **Semantic Web**, **Resource Descriptive Framework (RDF)** and **Linked Data**.

Welcome to the New Era of Data with Semantic Web

Semantic Web is an extension of the Web that adds semantics to the current format of data representation enabling considerable gains in the information treatment. On the Web, you can post documents and photos about yourself, listen to podcasts, go shopping, and have video chats with friends. With Semantic Web, you also describe or codify the topic that you publish and with that

additional knowledge, remix it with data created by others and link together the most relevant of your information, establishing a world-wide standard of communication.



In a nutshell, Semantic Web was created to extend the Web and make data easy to reuse everywhere.

Realizing that XML is for documents, not data

When it was introduced in 1998, many thought that XML would solve the data-integration challenges that the world had at that point.

The problem is that it didn't fully solve that problem, as XML is a document mark-up language, not a data modelling language.

XML and its schema language, XSD, are not real data models, but document models. The difference is basically in how strong the model semantics are required to be. For example, a particular XML Schema nested tag could mean almost anything: parent-child, whole-part aggregate, unidirectional association. In contrast, Semantic Web languages are actual data models with strong, precise, mathematically grounded semantics.

Let's take a simple example of an XML tag:

```
<employee><name>Joe</name></employee>
```

Unfortunately, parsers cannot tell that the tag above means that there is a class of things called *Employee* and an instance of it called *Joe*.

On the other hand, let's take an RDF (Semantic Web language that will be presented later in this document) triple example:

```
<#employee><#name><#Joe>.
```

Any RDF parser can understand that the above triple means that there is a class of data called *Employee* and there is an instance of one *Employee* called *Joe*.

Recognizing the importance of ontologies

An ontology defines a common vocabulary for different stakeholders who need to share information in a domain. It includes machine-interpretable definitions of basic concepts within a domain and the relations among them.

Let's take a simple example: An airline may want to integrate cargo data coming from other airlines. The data can be imported into a common RDF model by using, for example, converters to the airlines databases. However, one database may use the term "cargo", whereas the other may use the term "freight". To make the integration complete, an extra definition should be added to the RDF data, describing the fact that the relationship described as "freight" is the same as "cargo". This extra piece of information is, in fact, a simple ontology.

Some of the reasons why someone would want to develop an ontology are:

- To share a common understanding of the structure of data among stakeholders.
- To use an accurate description of knowledge, unlike natural language in which words can have different semantic meanings depending on their context.
- To enable the reuse of domain knowledge. An example would be an ontology created to describe the domain of a bookstore, which once shared would allow other bookstores to build new catalogues without having to redo an analysis of their domain.
- To avoid misinterpretations and to contribute to greater reliability of knowledge representation.
- To enable automated reasoning about data.

Using the Resource Descriptive Framework (RDF)

The Resource Description Framework (RDF) is the base language for writing ontologies in Semantic Web and is used for describing data, metadata and other data languages. Any data model or data language that uses RDF is a part of Semantic Web.

RDF is based on the concept that every data item should have a unique Web identifier (a URI = Uniform Resource Identifier). Also, every data item can be linked to every other item, enabling an interconnected set of data that may be distributed at a global scale across the Internet.

RDF provides a simple model for defining data, but it does not specify advanced semantics for relationships and data models. For that purpose, there are more advanced languages like OWL (Web Ontology Language), an extension of RDF to supply a rich set of semantics for building complex data models and vocabularies.

Discovering Linked Data

The RDF data standard is built on the principles of Linked Data. The term Linked Data refers to a set of best practices for publishing structured data on the Web. Tim Berners-Lee introduced these principles in the design issue note "Linked Data".

The principles of Linked Data

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names
3. When someone looks up a URI, provide useful information through RDF links
4. Include links to other URIs, so that they can discover more things

ONE Record ontology to rule them all

IATA's ONE Record standard uses a web architecture based on a secure lightweight API and distributed data, allowing all the partners from the supply chain to share and access data actively.

The concepts of ontologies and Linked Data match the needs of the next era of transport and logistics perfectly, as the supply chain participants seek to digitalize their processes, improve the use and re-use of their information and maximize transparency and visibility.

The ONE Record model specification based on Linked Data principles provides the air cargo industry with a standard structure for data exchange using common ontologies that facilitates data integration with existing and new services.

From Physical Freight to Machine Readable Data

Semantic Web data formats were designed to provide a way to describe and define data by using more data. The web of Linked Data allows multiple and independent publishers to describe data models and data concepts in such a way that they can be linked, described, and queried as if they were part of a single database.

Different tools can be used in order to model physical objects into an RDF compliant vocabulary. In the following sections, we will showcase the methodology used by the ONE Record Data Model and technical experts for creating the ONE Record ontology.

How to easily create ontologies

Let's say that you understand Semantic Web, RDF and OWL and you feel that these concepts are the best thing since you have discovered how to use hashtags. Even after following technical examples and tutorials, you realized that you would never be able to create and maintain RDF (Resource Description Framework) nor OWL (Web Ontology Language) manually in your favorite text editor by copying and pasting model description from a spreadsheet.

Ontology

/änˈtäləjē/ /ɑnˈtɑlədʒi/

NOUN

1. The branch of metaphysics dealing with the nature of being.

2. A set of concepts and categories in a subject area or domain that shows their properties and the relations between them.

Such a process would be long and error-prone, and it would take hours and hours of copy-pasting for big vocabularies.

Fortunately, there are different "non-manual editing" ways to create RDF data and OWL ontology models.

Ontology Tools

There are many tools for developing and maintaining ontologies. All good ontology tools allow for the definition of classes, class hierarchy's variables, variable-value restrictions, and the relationships between classes and the properties of these relationships.

Protégé

Developed by the Stanford Center for Biomedical Informatics Research at the Stanford University School of Medicine, Protégé tool is one of the oldest and most widely deployed ontology modelling tools. It was originally conceived as a frame-based modelling tool for rich ontologies following the Open Knowledge Base Connectivity protocol. Later iterations of Protégé have expanded to include a plug-in that is now widely used for OWL and RDF modelling.

The ONE Record Data Model experts currently use the Protégé tool for creating the ONE Record ontology. Protégé is widely used in the academic community and it is free and open-source. Its fully-featured support for OWL and RDF also makes it popular among commercial enterprises.

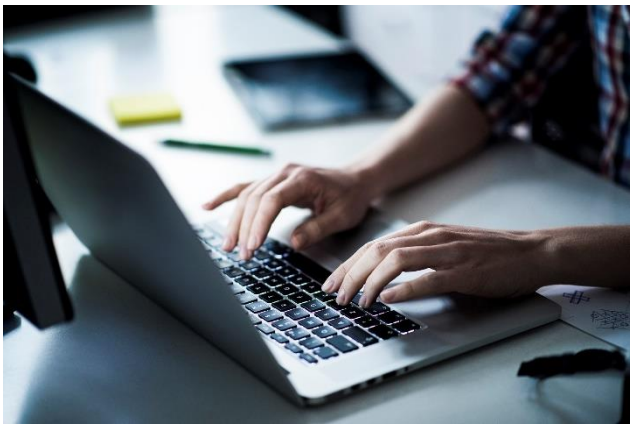
Crafting your first ontology

There is no "correct" way for developing ontologies. We can state that ontology development is following an agile process as it is an incremental and iterative process: we first defined the rough objects, then we defined the relations

between them and finally we refined the relations by adding further details such as cardinality.

Currently, the ONE Record ontology does not include business process definitions, and these will be defined in the next iterations of the ONE Record ontology development.

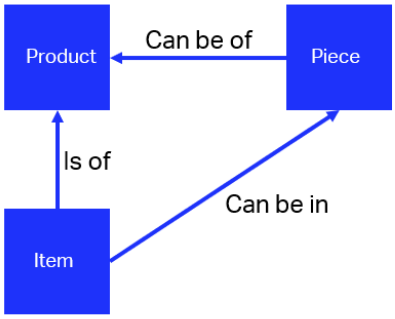
In the following example, we present the high-level steps in the creation of an ontology. You can find several step-by-step guides with screenshots of the user interface for each of these steps on the Protégé website, for example.



Step 1 – Define the terms of the ontology and the relations between them

As mentioned before, the concepts in the ONE Record ontology are digital twins of the physical entities in the cargo industry. In short, anything that exists in the physical world, as opposed to the digital world, can have a digital twin. These are nouns (objects) or verbs (relationships) in sentences that describe the ONE Record domain.

In the following simple diagram, the objects are **Product**, **Item** and **Piece**, digital twins of the physical objects with the same name from the freight ecosystem.



Fragment of ONE Record Data Model

Step 2 – Define the classes and the class hierarchy

The second phase of the ontology definition with Protégé is the creation of the classes. In our example, the classes would be **Item**, **Product** and **Piece**. As every Item is a Product, Item would be defined at a lower level than Product, meaning that Item is a subclass of Product (defined by "*Is of*" relation).

Step 3 – Define the properties of classes

Properties define the internal structure of concepts, as classes alone cannot provide enough information about a domain.

In our example, we would create a *containedItem* property that would be of **domain** Piece and **range** Item, the RDF translation of the fact that a Piece contains an Item.

All subclasses of a class inherit the properties of that class. For example, all the properties of the class Product will be inherited by Item, a subclass of Product.

Step 4 – Define further information about the properties

Properties can have different facets describing the value type, allowed values, the cardinality and other features of the values the properties can take.

The classes to which a property is attached or classes that the property describes are called the domain of the property. In our example, *containedItem* is of domain Piece.

Range define allowed classes for a property. In our example, *containedItem* can only be of range Item.

In our example, the property *containedItem* also has a cardinality of minimum 1, meaning that a Piece contains at least one Item.

Step 5 – Generate Turtle representation of the ontology

After all the classes and their properties are defined in the Protégé environment, a Turtle representation of the ontology can be exported.

Turtle is a syntax and file format (.ttl) for expressing RDF data model that is machine-readable. Software developers can use this file to implement applications consuming ONE Record compliant models.

Example of a simple generated Turtle representation for the description of *containedItem* property:

```
piece:containedItem rdf:type owl:ObjectProperty ;
                    rdfs:domain :Piece ;
                    rdfs:range :Item ;
                    rdfs:comment "Details of contained item(s) "@en ;
                    rdfs:label "piece:containedItem"@en .
```

Turtle syntax for *containedItem* property

The cardinality of *containedItem* is generated at the Piece level class description, as follows:

```
:Piece rdf:type owl:Class ;
       rdfs:subClassOf [ rdf:type owl:Restriction ;
                         owl:onProperty piece:containedItem ;
                         owl:minCardinality "1"^^xsd:nonNegativeInteger
                       ] ,
```

Turtle syntax for cardinality

In this section, we have described a short ontology-development methodology that is used by ONE Record Data Model and technical experts via Protégé tool. However, after following the suggestions, the most important things to remember are the following: **there is no single correct ontology for a domain and there is no single correct way to create it**.

From a Document-based Web to a Web of Meaningful Interlinked Data with JSON-LD

When we retrieve some data from the Web, we are unlikely to have accessed all the relevant information about that resource. Additional data or content may be available from both the original source, as well as other third-party sources on the Web. The Semantic Web, through the Linked Data initiative, connects resources on the Web and allows applications to find more information by repeatedly following links to discover those additional sources.

Getting to know JSON

JSON (JavaScript Object Notation) is an open standard and lightweight file format for storing and transporting data that is easy for machines to parse and generate.

JSON is defined as a collection of name/value pairs, as in the example below:

```
{
   "firstName": "Jane",
   "lastName": "Doe",
   "contact": {
     "emailAddress": "email@example.com",
     "phoneNumber": "4123456476553"
   },
   "jobTitle": "Developer"
}
```

Example 1 - JSON

Considering that the JSON format is text only, it can conveniently be sent to and from a server and used as a data format by all the programming languages.

The magic of Linked Data

Linked Data is a way to structure and exchange information by using links and allowing the creation of a network of machine-readable data across the Web. It enables an application to start at one piece of Linked Data and follow embedded links to other pieces of Linked Data that are hosted on different servers across the Web (called "Follow-Your-Nose effect").

Linked Data enables a **decentralized architecture** as it is built on URIs that lead directly to the source of the data, even if the data is found on a different server. Consequently, as the data is decentralized, it usually remains at source, which means that it is copied less, and the cost of hosting it is reduced.

JSON + Linked Data = JSON-LD

JSON-LD (JSON for Linked Data) is a method of encoding Linked Data using JSON. It is easy for humans to read and write it and it is compatible with various programming environments, REST Web services, and unstructured databases. JSON-LD it was built in such a way that it requires minimum effort from developers to transform their existing JSON.

JSON by itself is often meaningless outside its creator's context. Its meaning can only be interpreted from documentation or technical specifications shared by its creator. JSON-LD encodes contextualized meaning to JSON, by adding a vocabulary to define its attributes and allowing search engines and applications to directly understand what a particular entity is about.

You can find a simple example of JSON-LD in the following picture:

```
{
  "@context": "https://onerecord.iata.org",
  "@id": "https://somewebsite/janeDoe",
  "@type": "Person",
   "firstName": "Jane",
   "lastName": "Doe",
   "contact": {
     "@id": "https://somewebsite/janeDoe/contact",
     "@type": "Contact",
     "emailAddress": "email@example.com",
     "phoneNumber": "4123456476553"
   },
   "jobTitle": "Developer"
}
```

Example 2 - JSON-LD

The differences between Example 1 and Example 2 are in the following elements:

- @context element, which means: "Hey Browser, the vocabulary I am referencing can be found at onerecord.iata.org";
- @id element, which identifies the node at which the data about Jane Doe can be found. This means "Hey Browser, the information about Jane Doe can be found at somewebsite/janeDoe";
- @type element, which means "Hey Browser, I am using the Person type, which can be found at onerecord.iata.org/Person". When typing this sort of URL in the browser, the full documentation and technical specifications of the Person type should be returned.

ONE Record standard is built on the principles of Linked Data and its API specifications fully support JSON-LD. You can find the latest ONE Record ontology and ONE Record compliant JSON-LD files on IATA ONE Record Github page.

Bridging Semantic Web and Object-Oriented Programming with Object Triple Mapping

The use of Semantic Web and ontologies in creating smart applications has drastically increased during the recent years because of its high potential in allowing machines to reason with provided semantics.

Providing powerful tools that can improve the development of applications based on ontologies can accelerate the adoption of Semantic Web. For this purpose, many tools such as ontology editors, ontology reasoners, triple store frameworks and object triple mapping systems have been developed.

This section introduces the concept of object triple mapping, which essentially consists of a bridge between RDF/OWL and object-oriented programming.

Describing the world with object-oriented programming

Object-oriented programming (OOP) is a paradigm for software development, based on the concept of objects, that can contain data, in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods).

OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. OOP is well-suited for programs that are large, complex and need constant updates or maintenance. The most popular OOP languages include Java, Python and C++.

Fundamental characteristics of OOP

1. Inheritance – Parent to child relationships;
2. Polymorphism – Overloading and overriding class members;
3. Encapsulation – Hiding data behind operations;
4. Abstraction – Reveal mechanisms that are only relevant for the use of other objects.

In the same way as in the creation of ontologies, in OOP a developer needs first to identify all the objects that they want to manipulate as well as the relationships between them (a process known as data modelling).

How Semantic Web and OOP match

Semantic Web shares a few major characteristics with OOP:

- Semantic Web has classes for defining data concepts.
- Semantic Web has instances for actual data values.
- Inheritance among classes is supported.
- Strongly typed data types are supported (string, integer, etc.).
- Whole/part relationships are supported. Whole/part relationships are when one class represents the whole object and other classes represent parts. The whole acts as a container for the parts.

Aside from these features, Semantic Web is more rigorous and formal, by specifying extra information such as the allowed relations between classes and their instances.

How OOP and Semantic Web differ

There are several discrepancies between OOP and RDF/OWL, as defined by the Software Engineering Task Force (SETF):

- Classes in OOP are regarded not as sets to which instances belong, but as types for instances.
- Each instance in OOP belongs to one class as its type's instance.
- Instances in OOP cannot change their type at runtime.
- The list of compiled classes in OOP must be fully known at runtime and it cannot change after that.
- There is no reasoner in OOP that can be used for consistency checking at runtime.
- Properties in OOP are defined locally at the class level and not as stand-alone entities.
- Instances in OOP cannot have arbitrary values for any property without the definition in its class, and no domain constraint.

Challenges for technical experts working with Semantic Web

The most challenging concern for a software developer who is trying to model RDF or OWL is to unlearn the OOP acquired practices.

For programmers with OOP background it is quite easy to model OWL classes. Still, they need to unlearn the idea that classes are just static datatype for objects at runtime and start considering classes as dynamic sets of instances that may change membership at any time during runtime.

Database experts are generally more comfortable with the notion of OWL classes as sets. However, they still need to resist the appeal of normalizing the data model using primary and foreign keys and focus on modeling objects that accurately represent the information model.

Example from "JOPA: Accessing Ontologies in an Object-oriented way" paper

"(…), an ontology specifies that each Person has a name. Due to the open world assumption, the ontology is consistent even if a particular Person does not have recorded his/her name. However, a genealogical application accessing the ontology needs the name to be known, which causes the application to crash whenever it receives (consistent, but application–incompatible) data from an ontological source, specifying a Person without a name."

Introducing Object Triple Mapping

Object triple mapping (OTM) was designed for helping to develop ontology-based applications through the object-oriented programming paradigm.

OTMs map RDF triples into data objects, enabling developers to handle RDF triples as objects in object-oriented applications. In the same manner, developers of relational databases have been using object-relational mapping (ORM) systems (e.g. DQL or Hibernate) to map relational databases into objects.

OTM with JOPA

There are several libraries for multiple programming languages that have been introduced for mapping RDF model to the object model.

One that was particularly tested by the ONE Record team is JOPA (Java OWL Persistence API). The code is available on Github.

JOPA is a domain-specific library that expresses the domain model in the form of object classes, and their properties (attributes or relationships). JOPA automatically maps the notions from the object-oriented realm to the semantic data model and vice versa.

JOPA compiles integrity constraints into a set of Java annotations that keep their semantics at the object level. This allows easy validation of restrictions such as cardinality, as well as domains and ranges of annotations, data or object properties.

The key outcome from using OTM

Nowadays, a huge amount of existing applications is coded with object-oriented programming languages such as Java or Python.

A key point for accelerating the adoption of Semantic Web is the creation and usage of tools such as object triple mapping frameworks that translate RDF/OWL to object-oriented models. This kind of tools is crucial as they save hours of development and maintenance and translate ontologies in languages that are more familiar to developers.

There are many OTM libraries. However, these libraries differ a lot in their capabilities and there is still much work to do in this area.

Takeaways & Next Steps to Take from Here

Say by now, you're convinced that Semantic Web and ontologies are a game-changer for digitalization of logistics. But where do you go from here? This document is only a first step for bettering your understanding of new formats and architectures. Here are some ideas of what you could do next to deepen your knowledge.

Seven Next Steps to Take from Here

1. Read more about IATA ONE Record standard on IATA website
2. Read up on RDF and OWL Modelling
3. Read RDF and OWL Specifications
4. Check out IATA ONE Record Github repositories
5. Install an ontology editor and play with IATA ONE Record ontology or write your own ontology
6. Be brave: Participate in a ONE Record Hackathon
7. Get to the next level & join a ONE Record pilot

The main takeaway of using ontologies and Linked Data is that structuring your data makes it easier for other applications and search engines to discover and understand it. What the cargo ecosystem needs – and what the Semantic Web can provide – is a universal solution for linking disassociated freight content into a larger cohesive Internet of Logistics.

ONE Record promises a new digital era for logistics and transport supply chain, and Linked Data is one of its key enablers as the industry will fully benefit by the efficiency and transparency it provides in sharing data.

"ONE Record uses the existing possibilities of the Internet and guarantees fast and uncomplicated access to the data via URIs. Beyond the technical aspects, this approach changes the way we handle air cargo data. Instead of focusing on transferring data from one company to another, the focus will shift to what we can do with that data."

Henk Mulder, Head of Digital Cargo, IATA.

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)

International
Air Transport
Association
Route de l'Aéroport
33, P.O. Box 416 1215,
Geneva 15 Airport,
Switzerland.
Tel: +41 (0) 22 770 2525