

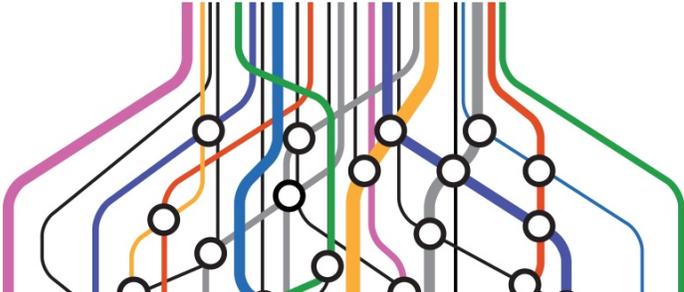
# Web APIs for the International Supply Chain

A primer for non-technical people

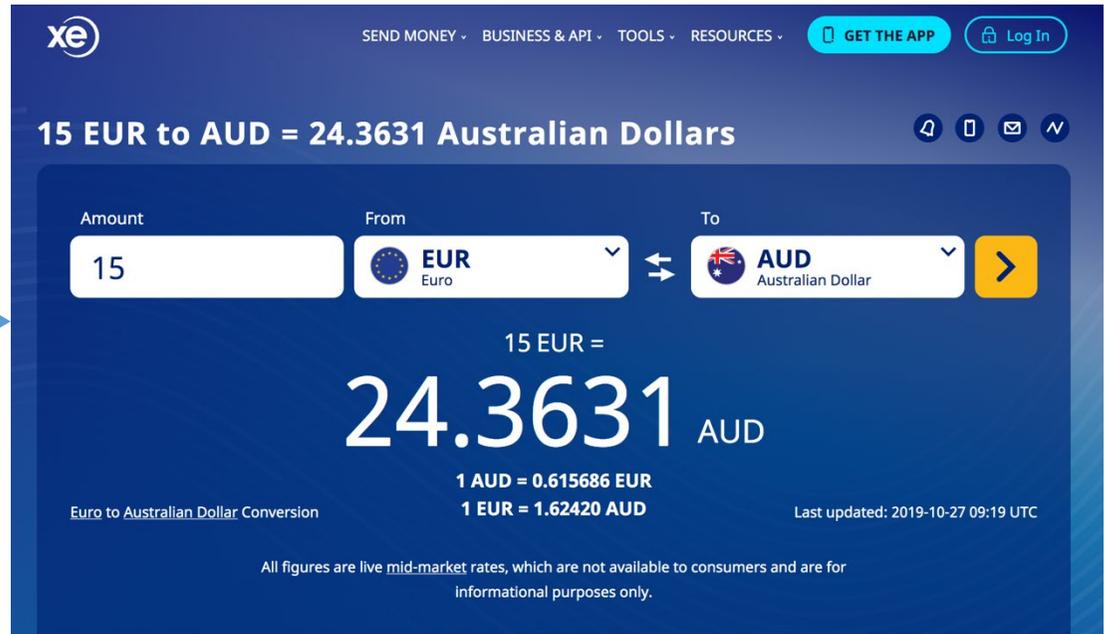
Steve Capell  
steve.capell@gmail.com



**UN / CEFAC**



# Here's a pretty website



The screenshot shows the XE website interface for currency conversion. At the top, there are navigation links: SEND MONEY, BUSINESS & API, TOOLS, RESOURCES, GET THE APP, and Log In. The main heading displays the conversion: 15 EUR to AUD = 24.3631 Australian Dollars. Below this, there is a form with three main sections: 'Amount' with a text input containing '15', 'From' with a dropdown menu set to 'EUR Euro', and 'To' with a dropdown menu set to 'AUD Australian Dollar'. A yellow arrow button is to the right of the 'To' dropdown. Below the form, the result is shown: 15 EUR = 24.3631 AUD. At the bottom, there are reciprocal rates: 1 AUD = 0.615686 EUR and 1 EUR = 1.62420 AUD. A timestamp indicates the rates were last updated on 2019-10-27 09:19 UTC. A disclaimer at the bottom states: 'All figures are live mid-market rates, which are not available to consumers and are for informational purposes only.'

- Types [www.xe.com](http://www.xe.com)
- Enters amount
- Chooses From/to currency
- Hits go

Pretty familiar experience. It's obvious what it does.

# Here's a similar thing with just the data



```

{
  success: true,
  timestamp: 1572056346,
  base: "EUR",
  date: "2019-10-26",
  - rates: {
    JPY: 120.406304,
    AUD: 1.624184
  }
}
  
```

Types:

[http://data.fixer.io/api/latest?access\\_key=89aae4bb97079f6e159730ad3f000538&base=EUR&symbols=JPY,AUD](http://data.fixer.io/api/latest?access_key=89aae4bb97079f6e159730ad3f000538&base=EUR&symbols=JPY,AUD)

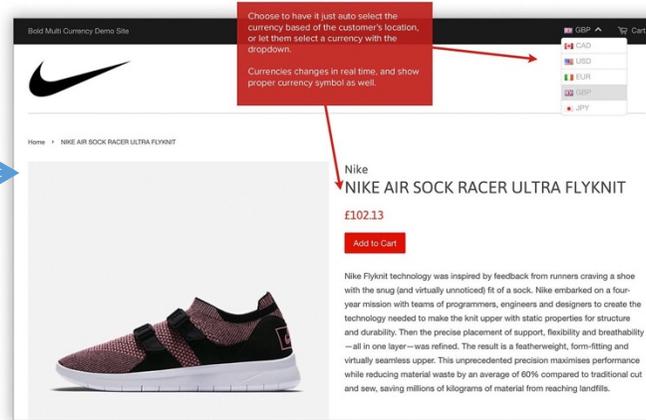
It's doing the same job. It's not as pretty for humans but it's **excellent for machines**.

This is a web API. API means "Application Programming Interface". Web means the protocol is "just use the web". More on that later.

# That's really cool because you can do this



Types  
[www.shoes.com](http://www.shoes.com)



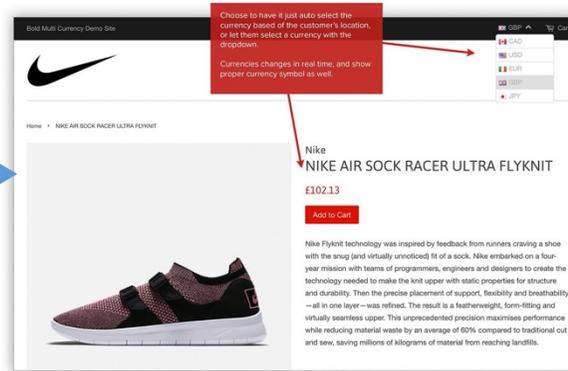
```
{  
  success: true,  
  timestamp: 1572056346,  
  base: "EUR",  
  date: "2019-10-26",  
  - rates: {  
    JPY: 120.406304,  
    AUD: 1.624184  
  }  
}
```

The online shop application doesn't have to maintain a database of exchange rates and update them daily. It just calls a web API to do that.

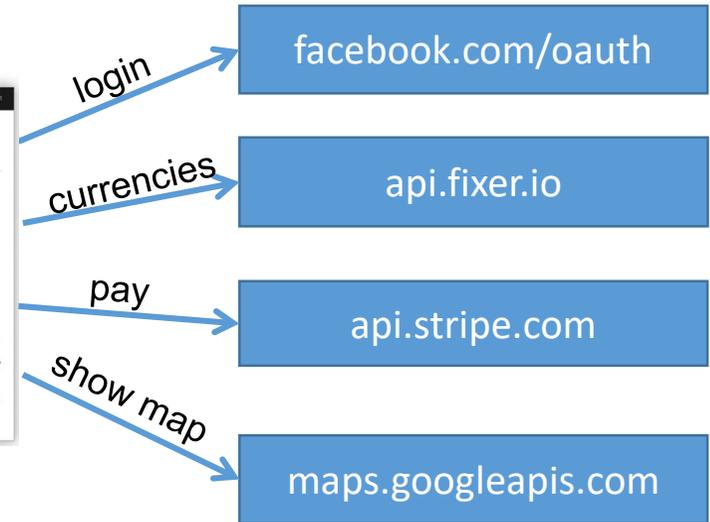
# In fact, most websites do this



Types  
[www.shoes.com](http://www.shoes.com)



## APIs



APIs are the building blocks of every modern website.

# And there's thousands of cool APIs

## Search ProgrammableWeb

shipping

SEARCH

One exact Category match : [Shipping](#)

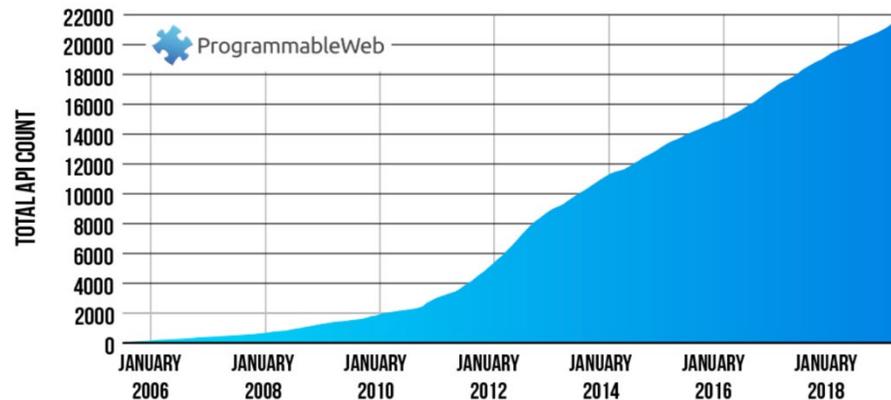
APIs (393)	SDKs (226)	Articles (557)	Libraries (19)	Sample Source Code (113)	Frameworks (2)	Mashups (54)
------------	------------	----------------	----------------	--------------------------	----------------	--------------

### APIS (393)

		Mashups	Followers
<a href="#">Shipping Gear</a>	Shipping Gear is an API that aims to help users with eCommerce shipping...	1	6
<a href="#">ParcelBright Shipping</a>	The ParcelBright Shipping API allows you to integrate shipping...	0	6
<a href="#">eCoComa Shipping</a>	The eCoComa Shipping API allows users to retrieve real-time shipping...	0	6
<a href="#">UPS Shipping</a>	The UPS Shipping API provides UPS shipping functionalities to be...	13	314
<a href="#">AuctionInc Shipping</a>	... e-commerce and auction tools, including a full-featured shipping rate...	0	11

[VIEW ALL 393 APIS](#)

GROWTH IN WEB APIS SINCE 2005



And many many more that aren't listed here.

# And here's a nice API from closer to home



Tracker site



Types  
[www.tracker.com](http://www.tracker.com)

Tracking results

MSKU0135001

From **Gaoming** To **Copenhagen** Container number: **MSKU0135001**

Container details MSKU0135001	Container type size 40ft Dry Container	Estimated arrival date 05 Dec 2019	Last location Discharge - Nansha New Port, Gu 25 Oct 2019
----------------------------------	---	---------------------------------------	---

All dates and times are given as reasonable estimates only and subject to change without prior notice.

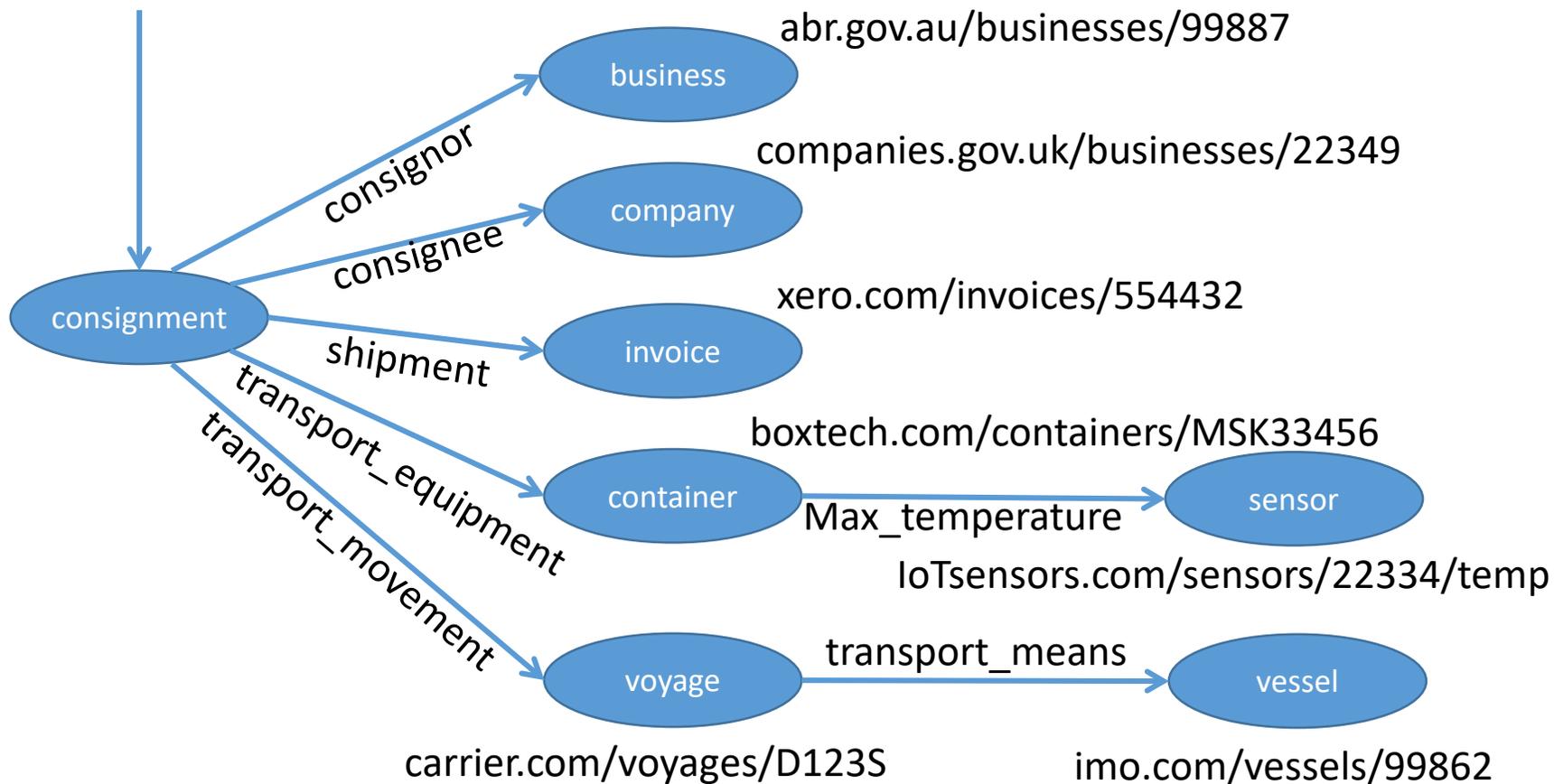
```
{
  "uploaderaccountname": "TOUAX",
  "bic_code": "GLDU",
  "prefix": "GLD",
  "equipment_identifier": "U",
  "serial_number": "533426",
  "check_digit": "0",
  "group_st": "20DC",
  "detail_st": null,
  "tare_kg": 2230,
  "tare_lbs": 4920,
  "max_payload_kg": 30480,
  "max_payload_lbs": 67200,
  "max_gross_mass_kg": 28250,
  "max_gross_mass_lbs": 62280,
  "cubic_capacity_cbm": 33,
  "cubic_capacity_cuft": 1170,
  "stacking_kg": 216000,
  "stacking_lbs": 476190.
}
```

As a carrier, or a port authority, or a regulator – just tell me the container number and I'll go to the source of truth to get details.

# Lets do some “Imagineering”

Here’s a consignment resource

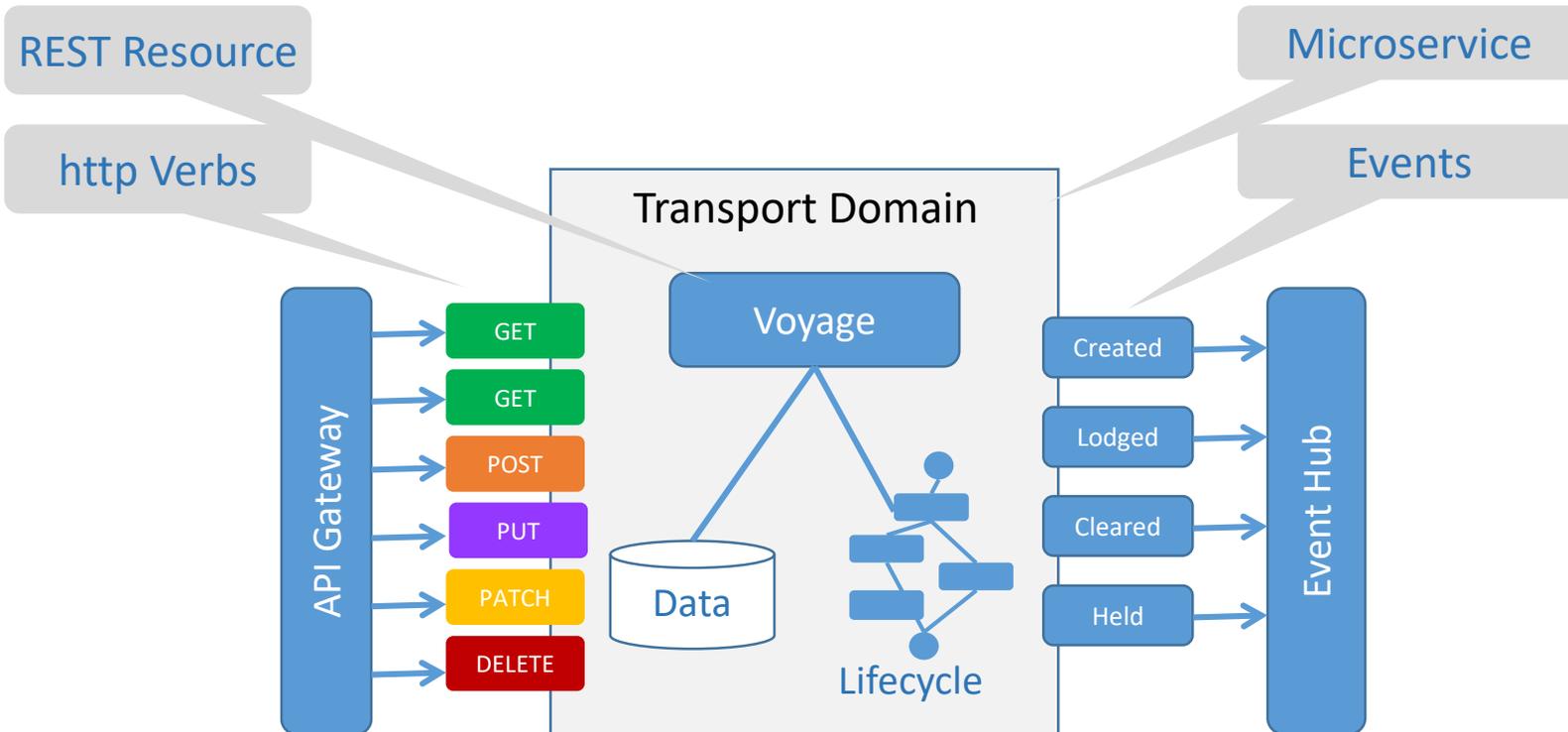
<https://api.3plcompany.com/consignments/12345>



Its really a set of links to sources of truth. A paradigm shift from document exchange to linked data discovery. This is a different business model, not just a different technology

# And lets clear up some terminology

REST = “Representational State Transfer”. A mouthful – but it just means a best-practice style for building high quality web APIs



POST [https://api.transport.border.gov.au/v1/voyages/{new\\_voyage\\_data}](https://api.transport.border.gov.au/v1/voyages/{new_voyage_data})

PATCH [https://api.transport.border.gov.au/v1/voyages/V1234S/{updated\\_voyage\\_data}](https://api.transport.border.gov.au/v1/voyages/V1234S/{updated_voyage_data})

GET <https://api.transport.border.gov.au/v1/voyages?Port=AUBNE>

# But what about security?

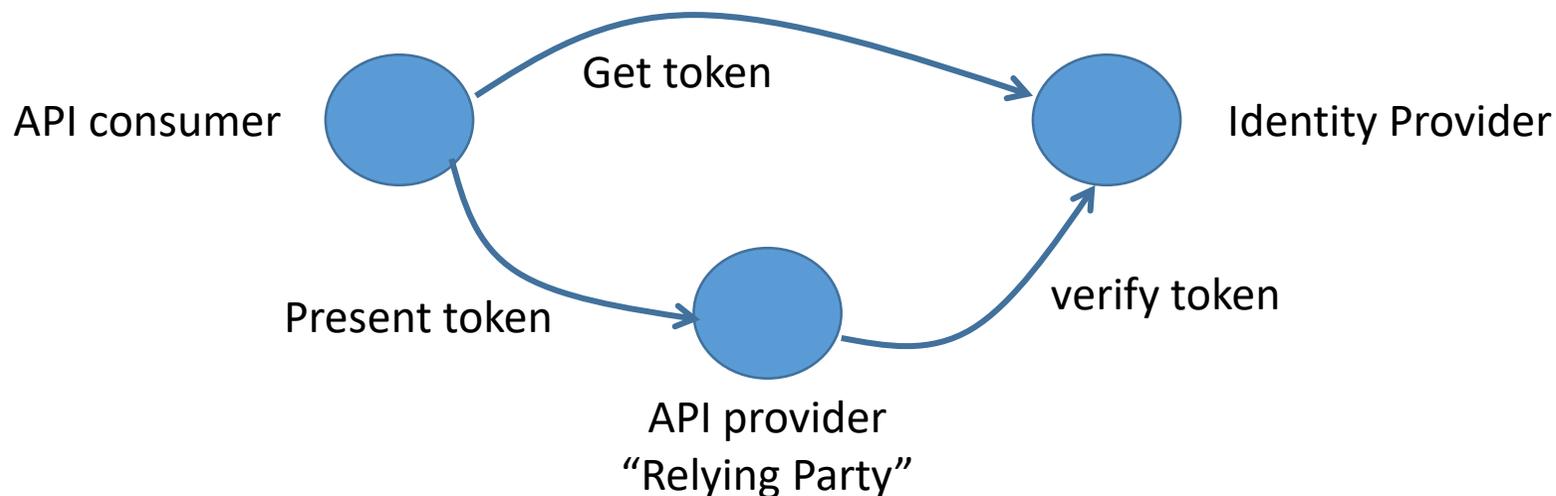
As an API consumer that is following links to a distributed web of resources, how do I identify myself to each one and how does the API provider decide whether to grant access?



It won't scale well if every provider needs to register, verify identity, and issue tokens to every consumer.

# Federated Identity using OIDC

Fortunately the web has already solved this problem by separating the job of verifying identity from the job of providing a service. The standard protocol is called “Open ID Connect” (OIDC) but you all probably know it as “sign-in with FaceBook / Google / etc”. It works just as well for web APIs that return data as for websites that return pages.



## Tokens carry “claims”

Google knows this user as  
 steve.capell@gmail.com

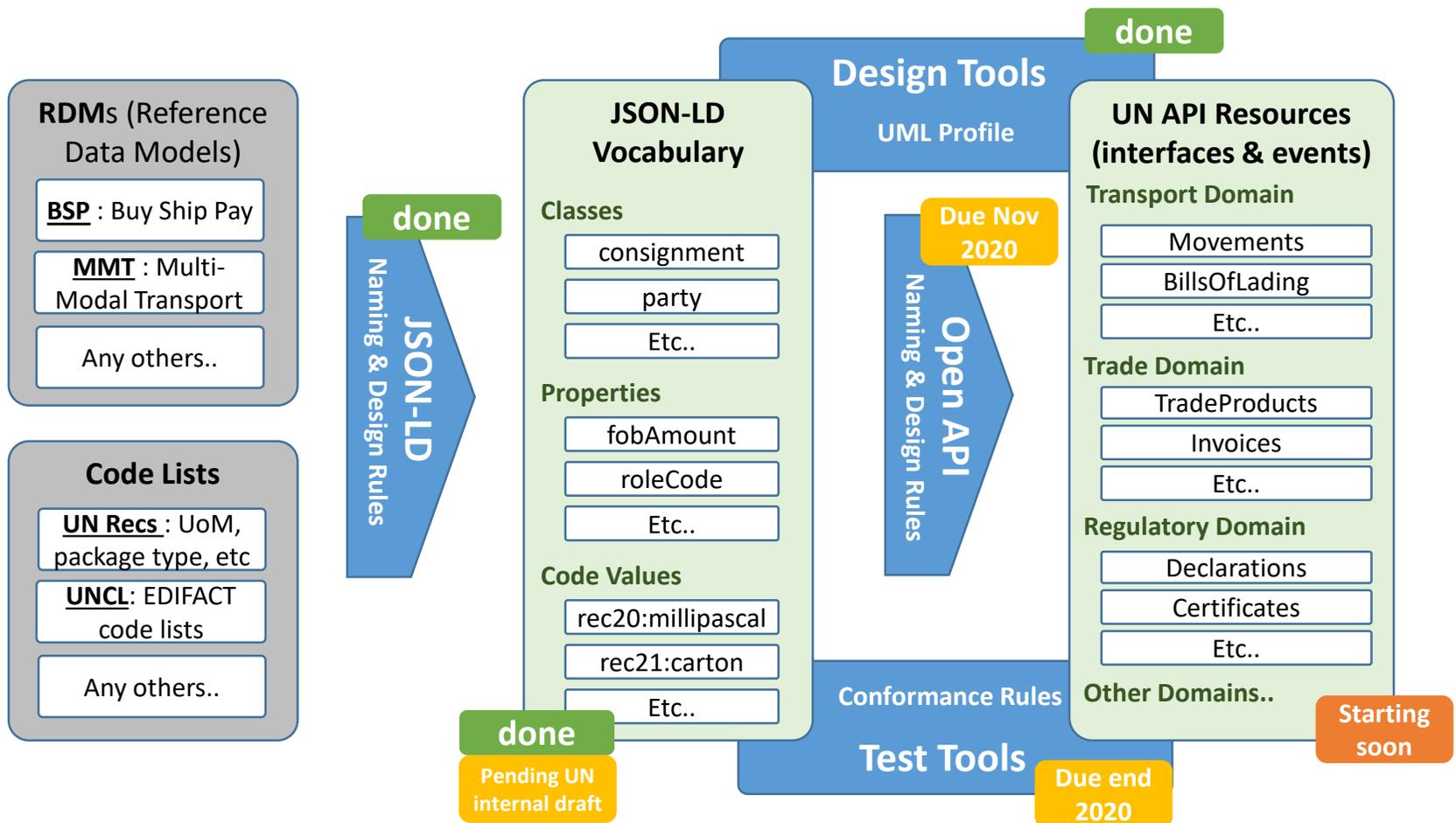
Some with weak integrity

ATO knows this party as  
 ABN 123456

Some with strong integrity

# But what about semantics?

- how to make consistent sense of the data in all these APIs?
- That's why UN/CEFACT is running the RDM2API project!





# Thanks for listening

Feel free to contact me

- [Steve.capell@gmail.com](mailto:Steve.capell@gmail.com)
- [edi3.org](http://edi3.org)