How JSON Schema NDR and OpenAPI can enhance the interoperability and accessibility of UN/CEFACT standards

Lunch & Learn
8th May 2023
UN/CEFACT API NDR Project

- UN/CEFACT JSON NDR
- UN/CEFACT API RULES
- openAPI 3.x DESIGN RULES
- UN/CEFACT RDM
- UN/CEFACT PoC JSON SCHEMA
- ANY API DESIGN TOOL
- UN/CEFACT PoC API YML
- TEXTILE & LEATHER CERTIFICATE of ORIGIN
- BUSINESS REQUIREMENTS

- Project deliverables
- Proof of concept artefacts
- UN/CEFACT publications or artefacts
- Other

- To be produced based on NDR
- To be produced based on NDR and API rules
How JSON Schema NDR and OpenAPI can enhance the interoperability and accessibility of UN/CEFACT standards

Andreas Pelekies
Lead editor API-Tech-Spec project
andreas.pelekies@gefeg.com
Lunch & Learn
May 8th 2023
Just-in-time production needs just-in-time-data – but things happen!

3,500 cars a day, 10,000 parts per car
10,000 suppliers
Parts only for 2 hours of production
ETA is the most important parameter
~ 20 minutes EDI messages containing the ETA for each individual part are sent
One order per part
408
Request timeout

API design patterns

Joining meeting timeout.

Your connection has timed out and you cannot join the meeting. Verify your network connectivity and try again.
Many API design patterns out there

RESTful web API design

In this article
- What is REST?
- Organize the API design around resources
- Define API operations in terms of HTTP methods
- Conform to HTTP semantics
- Filter and paginate data
- Support partial responses for large binary resources
- Use HATEOAS to enable navigation to related resources
- Versioning a RESTful web API
- Open API Initiative

Patterns for integrating microservices

The following patterns are used to integrate microservices architecture.

- API gateway pattern
- Decouple messaging pattern
- Pub/sub pattern
Individual industries have individual requirements and individual implementation standards

3.6 Property names
Property names on objects **MUST** be in camelCase.

Property names containing arrays **SHOULD** be plural.

Property names **MUST NOT** include FK (Foreign Key) or PK (Primary Key), as this exposes database design.

Boolean properties **MUST** be prefixed by either `is` or `has`.

3.7 Enum values
Enum values **SHOULD** be declared using `UPPER_SNAKE_CASE`.

3.8 Arrays
Empty arrays **MUST NOT** be represented with null – but **MUST** be empty lists `[]`.

3.9 Date and Time properties
Date properties **MUST** be suffixed with "Date". A Date property only contains a date in YYYY-MM-DD format.

Source: DCSA API Design Principles 1.0, September 2020
UN/CEFACT Modelling Framework

**Methodology**
- Business requirement specification (BRS)
- Core Component Technical Specification (CCTS)
- Core Components Business Document Assembly (CCBDA)
- UN/CEFACT Modelling Methodology (UMM)
- Naming and Design Rules (NDR)

**Processes and Guidance**
- Trade facilitation recommendations

**Semantic Reference data models**
- UN/CEFACT Core Component Library (CCL)
- Supply Chain Reference Data Model (SCRDM)
- Multi Modal Transport (MMT) Reference Data Model
- United Nations Trade Data Elements Directory (UNTDED) Semantics
- United Nations Trade Data Elements Directory (UNTDED) Syntax Codes

**Syntax**
- UN/CEFACT XML
- UN/EDIFACT

**Modelling artefacts**
- Business artefacts
- Technical artefacts
From generic to concrete: Layers of profiles / subsets

Buy Ship Pay

Transport

Regulatory
- Im-/Export Declaration
- Customs Items
- Cargo Report

Consignment Item

Consignment details (Invoice Amount)

Your subset (e.g. eCMR)

Road consignment

Supply Chain
- Shipment Details
- Trade Item
- Trade Details (Invoice)

Trade Details (Invoice)

Trade Item

Shipment Details
NDR’s: From RDM to API

Reference Data Models

- Buy Ship Pay
- Transport
- Supply Chain
- Your subset

Compliance

Open API 3
JSON Schema

JSON-LD
UN/CEFACT API Standards

CCL / Reference Data Model

JSON Schema NDR TS
https://github.com/unecefact/spec-JSONschema

OpenAPI 3.x NDR TS
https://github.com/unecefact/spec-openAPI

JSON Schema publication

OpenAPI example Specifications

API Implementation

Input
Output
Specifications
Third party

"$schema" "type" "enum"

/paths responses events
UN/CEFACT JSON-LD Vocabulary based on the BSP RDM

Internal UN/CEFACT implementation
Ever tried buying furniture online?

... for a room like this one ...
The magic of schema.org and JSON-LD

Search for "swiss cheese fondue"

About 7,100,000 results (0.43 seconds)

Recipes:

- **Classic Swiss Cheese Fondue**
  - Food & Wine
  - Rating: 5.0 ★★★★★ (3,6K)
  - Time: 15 mins
  - Ingredients: Swiss cheese, white wine, lemon juice, garlic, kirsch

- **Authentic Swiss Cheese Fondue**
  - Earth, Food, and Fire
  - Rating: 4.5 ★★★★★ (30)
  - Time: 30 mins
  - Ingredients: White wine, crusty French bread, gruyere cheese, garlic, kirsch

- **Swiss Cheese Fondue (The Best)**
  - Ricardo Cuisine
  - Rating: 5.0 ★★★★★ (73)
  - Time: 25 mins
  - Ingredients: Ham, broccoli, white wine, green apple, cherry tomatoes
Interoperability by design between JSON Schema and JSON LD

- **CCL / Reference Data Model**
  - JSON Schema NDR TS
    - https://github.com/uncefact/spec-JSONschema
  - JSON-LD NDR TS (work-in-progress)
    - https://github.com/uncefact/spec-jsonld
  - OpenAPI 3.x NDR TS
    - https://github.com/uncefact/spec-openAPI

- **API Implementation**
  - JSON Schema publication
    - "$schema", "type", "enum"
  - JSON-LD Graph
    - @id, @context, @type
  - OpenAPI example Specifications
    - /paths, responses, events

- **Input**
- **Output**
- **Specifications**
- **Third party**
Buy Ship Pay Reference Data Model
Draft JSON-LD Vocabulary

https://service.unece.org/trade/unecefact/vocabulary/unecefact/
Challenge => Subset Reference Data Models
UN/CEFACT OpenAPI NDR 1.0
API Design Patterns for B2B communication
## 3 API NAMING AND DESIGN RULES

3.1 Conformance and Compliance ........................................................................... 13

3.2 Design Rules ........................................................................................................ 14

3.2.1 Media type for structured data exchange ....................................................... 14

3.2.2 Endpoints ......................................................................................................... 14

3.2.3 Discoverability ................................................................................................. 16

3.2.4 Date and Time .................................................................................................. 16

3.2.5 Using the UN/CEFACT semantics ................................................................... 16

3.2.6 Operations ....................................................................................................... 21
3.2.7 Pagination ........................................................................................................... 24
3.2.8 Filtering .................................................................................................................. 26
3.2.9 Sorting .................................................................................................................. 28
3.2.10 API Responses and error handling ..................................................................... 28
3.2.11 Error Response Payload ...................................................................................... 34
3.2.12 Design rule examples .......................................................................................... 36
UN/CEFACT OpenAPI NDR TS 1.0

4.2 API VERSIONING ........................................................................................................... 37
  4.2.1 Versioning Scheme ................................................................................................. 37
  4.2.2 URI Versioning ....................................................................................................... 38
  4.2.3 Providing version information ............................................................................. 38
  4.2.4 Robustness ........................................................................................................... 40
  4.2.5 Deprecation and End of Life Policy .................................................................... 41

4.3 HYPERMEDIA ............................................................................................................... 44
  4.3.1 Hypermedia - Linked Data .................................................................................... 44
  4.3.2 HATEOAS ............................................................................................................. 44
  4.3.3 Hypermedia Compliant API .................................................................................. 45

API SECURITY ..................................................................................................................... 47

EVENT DRIVEN DATA EXCHANGE .................................................................................. 49

6.1 CALLBACKS .................................................................................................................. 50
6.2 WEBHOOKS .................................................................................................................. 50
UN/CEFACT OpenAPI NDR TS 1.0

Template and examples on

https://github.com/uncefact/spec-openAPI
UN/CEFACT JSON Schema draft 2020-12 publication

This directory contains JSON schema artefacts. They are fully based on the JSON schema draft 2020-12. This means that they fully comply with the requirements of OpenAPI 3.1.x.

In the library directory, the UN/CEFACT reference data models are provided in the library variant. This means that all contextualisations of the base classes can be reproduced. Thus, in addition to the basic data types, it also contains the business messages derived from the respective master messages. These can be further contextualised during implementation if required.

https://github.com/uncefact/spec-JSONschema
API Pilot Implementation in Automotive Inbound Supply-Chain

Jörg Walther
Odette International Ltd.
jwalther@odette.org
May 2023
In 2020 VDA and Odette started to explore the potential of Rest API in the Automotive Supply Chain

Aims:
- To get familiar with the (for us new) technology
- To decide, which standards exist and should be taken into account
- To identify potential Use-Cases
Our “API Toolbox”

<table>
<thead>
<tr>
<th>Description</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>UN/CEFACT DATA MODEL</td>
<td>There is a uniform framework for the specification of API regardless of the specific application area (logistics, finance, master data, etc.). The VDA/Odette recommendation uses international standards as far as possible.</td>
</tr>
<tr>
<td>UN/CEFACT API DESIGN RULES</td>
<td></td>
</tr>
<tr>
<td>OPEN-API V3.x SPECIFICATION</td>
<td></td>
</tr>
</tbody>
</table>

The required data structures are to be used and accepted internationally, therefore artefacts from the UN/CEFACT Reference Data Model are used. Here, too, the drafts (WiP) of the UN/CEFACT experts were used as the basis for the recommendation. Future adjustments are possible. From this, program codes can be generated automatically in various programming languages for implementation.
Since these documents were not ready, we actively supported their development and finalisation.
Identification of the Business Need (use-case)

UN/EDIFACT Scenario does not reach all partners and does not cover all processes

- **DESADV with parts and shipment information**
- **IFTSTA** Usually with reference to MoT
- **Transport status** often with reference to truck only (no shipment reference)
- **DESADV** often without MoT info
- Whether DESADV is correct can only be seen at goods reception.
From RDM to openAPI Spec (initial draft)

**Path Elements**
- **C ConsignmentItem**
- **C Consignment**
- **C TransportEquipment**
- **C TransportMeans**
- **C TransportMovement**
- **C TransportCapacityReservation**
- **C TransportEvent**
  - **A ID**
  - **A TypeCode**
  - **A ReportedConditionTypeCode**
  - **A OccurrenceDateTime**
  - **C OccurrenceLocation**
  - **C SpecifiedParty**
  - **C ReferencedTransportMovement**
  - **C ReferencedConsignment**

**paths**
- **/**
  - **extensions**
  - **/consignments**
    - **/consignments/(ident)**
    - **/consignment-items**
    - **/consignment-items/(ident)**
  - **/transport-events**
    - **/transport-events/(ident)**
  - **/transport-capacity-reservations**
    - **/transport-capacity-reservations/(ident)**
  - **/transport-movements**
    - **/transport-movements/(ident)**
  - **/transport-means**
    - **/transport-means/(ident)**
  - **/transport-equipments**
    - **/transport-equipments/(ident)**
  - **/exchanged-messages**
    - **/exchanged-messages/(ident)**
Implementation as MVP
Our Proof-of-Concept
**Business Information Orchestration (our DP*)**

1. Ship-from reports: shipment or THU is ready for pick-up
2. Ship-from reports: goods-issue (hand-over to carrier) completed
3. Forwarder or carrier reports: pick-up has been completed
4. Forwarder or carrier reports: transport or shipment has arrived at the destination of the transport leg
5. LSP reports: unloading of a shipment from a means of transport completed
6. Cross Dock reports: loading onto a means of transport completed
7. Forwarder or carrier reports: goods have been handed over to final recipient (proof of delivery)
8. Ship-to reports: LSP reports: unloading of a shipment from a means of transport completed
9. Ship-to reports: goods receiving completed

**Technical realisation:**
- POST /transportEvents
Expected Results / Benefits

Technically:
• One access point to provide and retrieve data for many different partners (EDI mainly Point-to-Point);
• Data capture and provision with mobile devices;
• New partners can be integrated more easily.

Business-wise:
• The actual (latest) status of shipments is always retrievable due to link between MoT and shipment;
• Shorter reaction times in cases of disturbances;
• In the end, a full history of the transport can be retrieved and analysed.

Technical realisation:
• GET /consignments/{id}
• GET /transportEvents/?ConsignmentID=XYZ123
Lessons Learned (so far)

• Regarding the business process:
  • Keep it simple: require only the necessary information;
  • Take, what you get;
  • Consider imperfect processes – make sure, the system still works.

• Regarding the data model:
  • We probably need a standardised way of defining new relations between ABIE (outside the document centric philosophy)
    e.g. we have Consignment – has – Events but we need Event – references – Consignments
    (in our context API are often event-driven).

• Regarding the solution:
  • Put the intelligence in the back-end.
E-Business Standard API

Implementation of development

Andreas Pelekies
Lead editor API-Tech-Spec project
andreas.pelekies@gefeg.com
Using a Business Requirement Specification

**BRS**
- Defines the processes and requirements

**JSON Schema**
- Contextualise RDM

**OpenAPI**
- Use JSON-Schema
- Create services for process implementation
Re-Use the wisdom of decades

- Import or re-use your own or your industries
  - XSD specification
  - JSON schema
  - JSON-LD
- Compliant to the UN/CEFACT standards
- Tick-off what you do not need
From Model to Specification including harmonised definitions an code lists

```json
openapi: 3.0.3
info:
  title: UN/CEFACT eCMR Demo using OpenAPI 3.0.3
description: |
version: 1.0.0
servers:
paths:
  /:
  /eCMR/dangerous-goods/{RegistrationPlateNumber}:
    get:
      tags:
        description: Gets a list of dangerous goods on a transport means.
        operationId: getDangerousGoodsInfo
        parameters:
        responses:
          200:
            description: OK
            headers:
            Link:
              description: |
              Link header providing information on pagination.

Examples:

Link:
<https://api.unece.org/demo?cursor=AZFJ98M>; rel="current"
```

DocumentCode:
  description: Code specifying the name of a document such as 352 for Proforma invoice, 380 for Commercial invoice.
type: object
properties:
  content:
    description: |
    Applicable codes:
      * '730' - Road consignment note
denum:
      - '730' # Road consignment note
type: string
listAgencyID:
  enum:
Use JSON Schema to create API Specs with the tool of your choice
For further information:

• All UNECE and UN/CEFACT Recommendations, codes, standards and publications are available for free on our website at: [www.unece.org/cefact/](http://www.unece.org/cefact/)

• The (draft) JSON Schema artefacts can be found at [https://github.com/uncefact/spec-JSONschema](https://github.com/uncefact/spec-JSONschema)

• The (draft) OpenAPI artefacts can be found at [https://github.com/uncefact/spec-openAPI](https://github.com/uncefact/spec-openAPI)

• All experts are welcome to join the standards development work free of charge
We look forward to answering your questions!

Lunch & Learn
May 8th 2023