\Orchestrating a brighter world

NEC Laboratories Europe, IoT Group

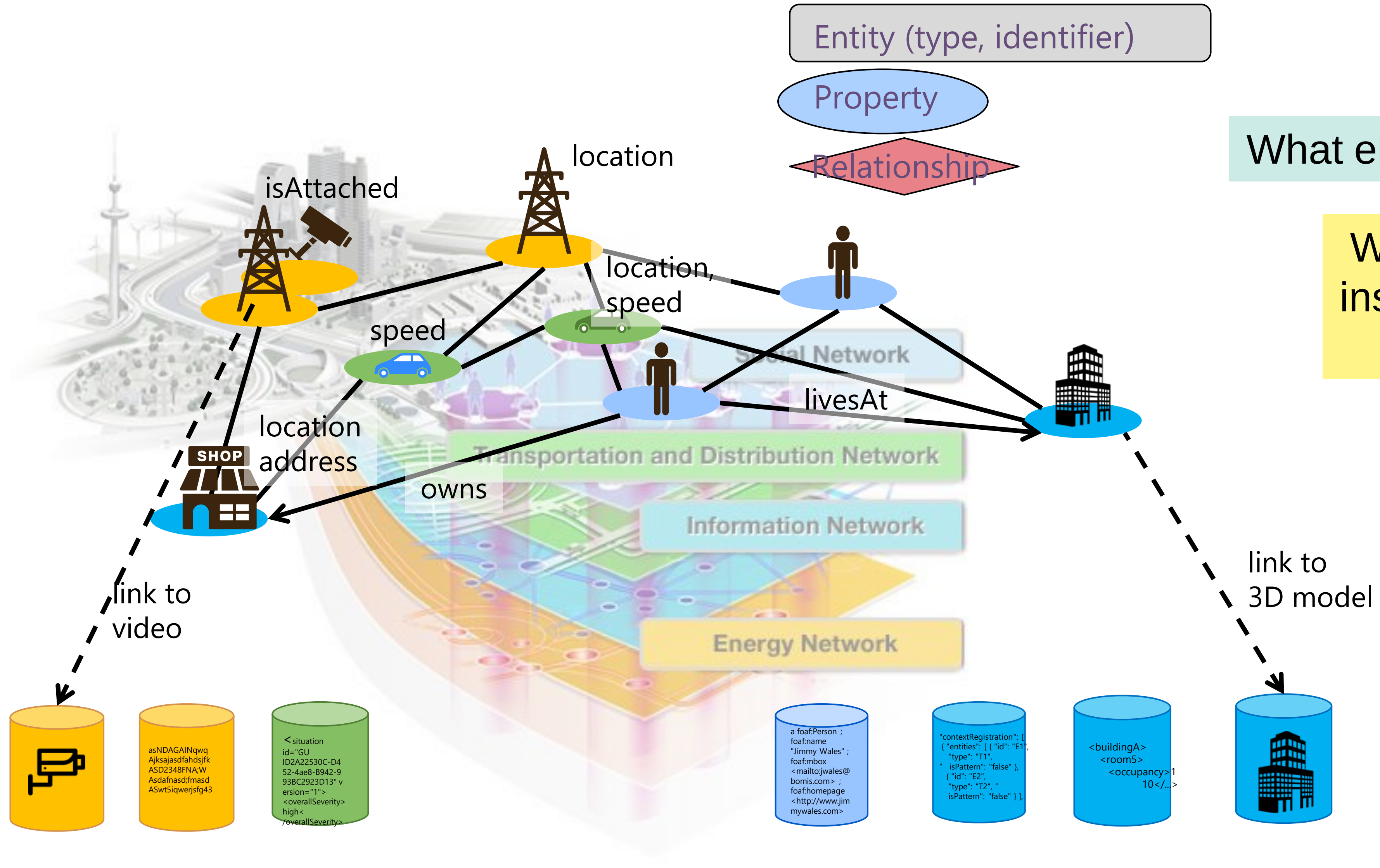
Creating OWL Ontologies based on Smart Data Models

21 January 2022, Martin Bauer

Table of Content

Smart Data Models

NGSI-LD Information Model



What entity types are there?

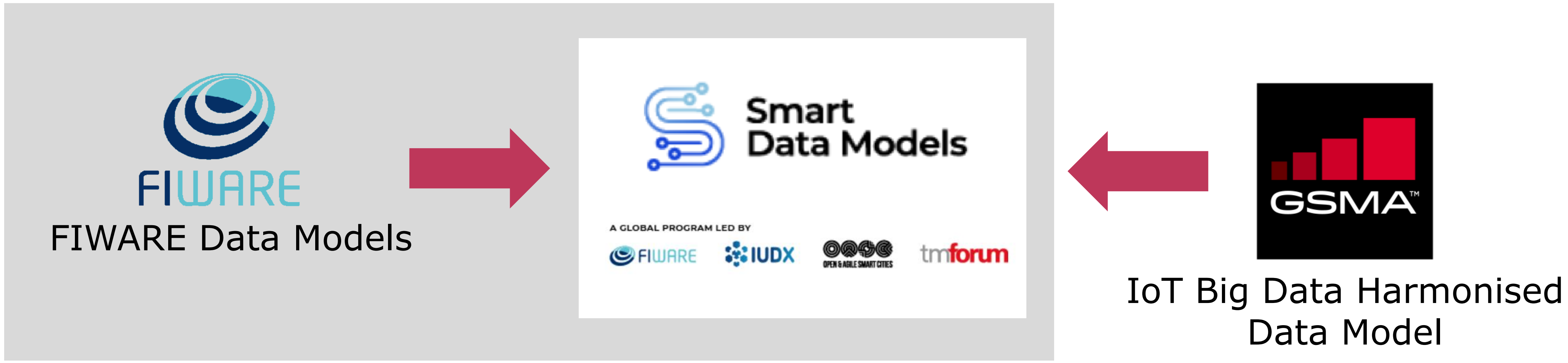
What properties can instances of a certain entity type have?

What relationships can instances of a certain entity type have?

→ Data Models

NGSI-LD Data Models

- NGSI-LD requires models defining entity types with their properties and relationships
- Existing NGSI-LD Data Models



→ FIWARE joined forces with tmforum, IUDX and OASC

→ **NEW: collaboration of FIWARE with OCF (Open Connectivity Forum) on Data Models (from their side: Device Models)**

Introduction

A GLOBAL PROGRAM LED BY FIWARE IUDX OPEN & AGILE SMART CITIES tmforum

What is Smart Data Models Program:

- Smart Data Models is a collaborative initiative to provide multisector agile standardized free and open-licensed data models based on <u>actual use cases and adopted open standards</u>

What is NOT Smart Data Models Program:

- Another ontology.
- Only working for NGSI standard (but it works of course)
- An standardization body but an agile standardization program

FIWARE

Alberto Abella is Presenting

Introduction

- **Specifications :**
  - 6 languages (EN, FR, GE, SP, JA, IT)
  - Automatically generated

- **Technical validation**:
  - Json schema
  - Single source of truth (include definitions)

- **Examples**:
  - Required
  - Json, jsonld, geojson feature, DTDL, csv

Current St

Added all the c ternal context options

25/10/2021

Cross Sector, Smart C Smart

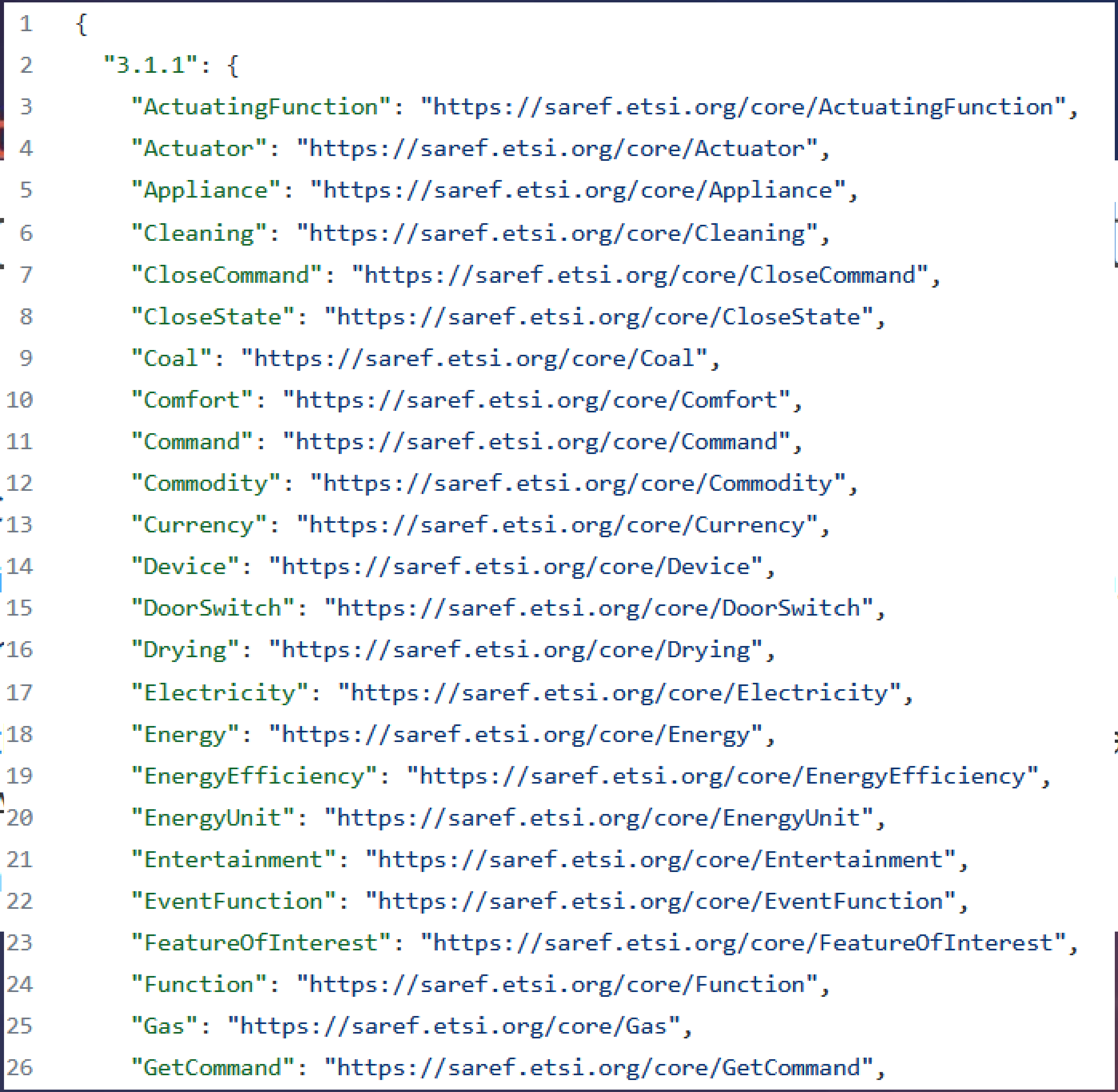
Manufacturing domai g domain,

SmartAeronautics, Sm

The new service for t available the

mapping of the 13 SA

Check them out in th



```
{
  "3.1.1": {
    "ActuatingFunction": "https://saref.etsi.org/core/ActuatingFunction",
    "Actuator": "https://saref.etsi.org/core/Actuator",
    "Appliance": "https://saref.etsi.org/core/Appliance",
    "Cleaning": "https://saref.etsi.org/core/Cleaning",
    "CloseCommand": "https://saref.etsi.org/core/CloseCommand",
    "CloseState": "https://saref.etsi.org/core/CloseState",
    "Coal": "https://saref.etsi.org/core/Coal",
    "Comfort": "https://saref.etsi.org/core/Comfort",
    "Command": "https://saref.etsi.org/core/Command",
    "Commodity": "https://saref.etsi.org/core/Commodity",
    "Currency": "https://saref.etsi.org/core/Currency",
    "Device": "https://saref.etsi.org/core/Device",
    "DoorSwitch": "https://saref.etsi.org/core/DoorSwitch",
    "Drying": "https://saref.etsi.org/core/Drying",
    "Electricity": "https://saref.etsi.org/core/Electricity",
    "Energy": "https://saref.etsi.org/core/Energy",
    "EnergyEfficiency": "https://saref.etsi.org/core/EnergyEfficiency",
    "EnergyUnit": "https://saref.etsi.org/core/EnergyUnit",
    "Entertainment": "https://saref.etsi.org/core/Entertainment",
    "EventFunction": "https://saref.etsi.org/core/EventFunction",
    "FeatureOfInterest": "https://saref.etsi.org/core/FeatureOfInterest",
    "Function": "https://saref.etsi.org/core/Function",
    "Gas": "https://saref.etsi.org/core/Gas",
    "GetCommand": "https://saref.etsi.org/core/GetCommand",
```

Current Status

- **Last Data Models:**

- Digital Innovation Hub:	Cross Sector
- Unmanned Aerial vehicles (drones):	Smart Aeronautics
- Open Connectivity:	Smart Sensoring
- EU Proof of Vaccination (COVID19):	SmartHealth
- Waste Water Plant:	Smart Water
- Streetlight Feeder:	Smart Cities

Differential factors

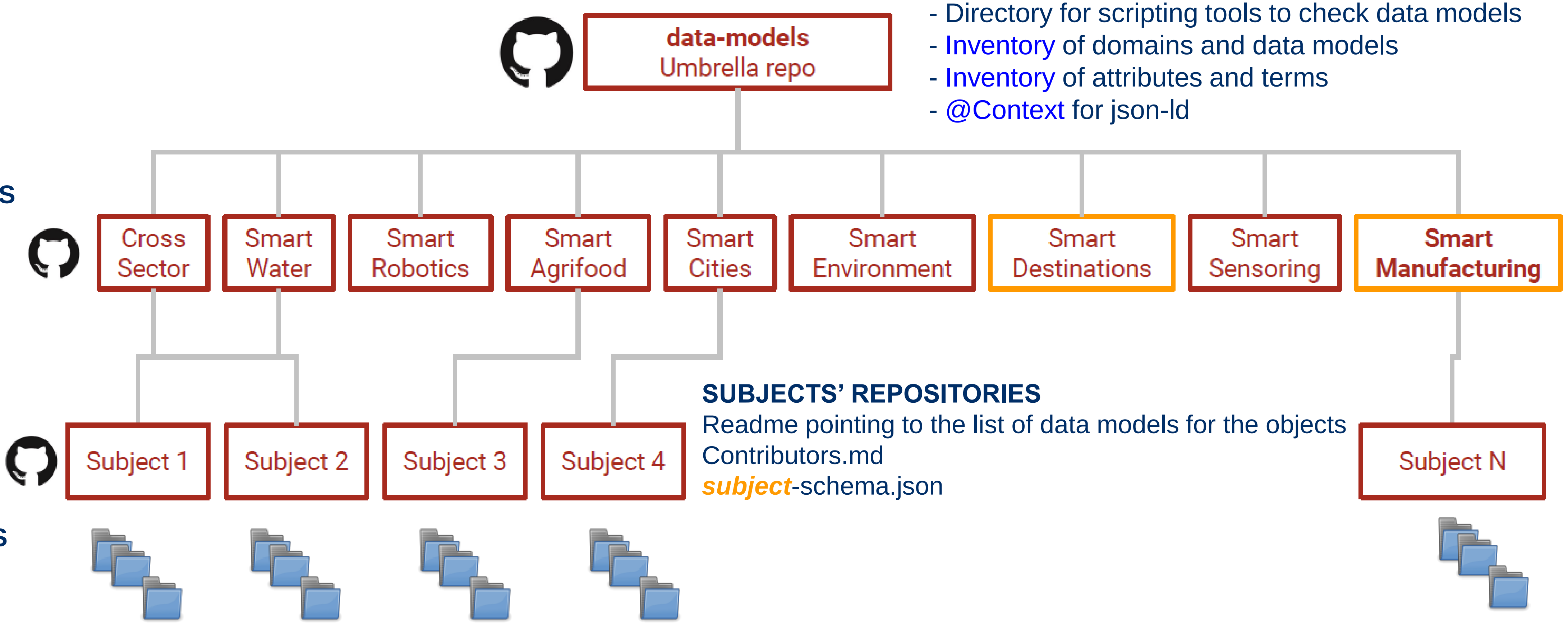
- **Open** licensed: Free use, modification and sharing
- **Collaborative**: Data models provides by actual users,
- **Not reinventing** the wheel: Adopting existing standards OCF, GTFS, GBFS, CIM Energy, OPC UA, etc.
- **Automated**: Publishing and translations (6): Once ready publication takes 15 minutes, tests can be automated
- Mapping **existing vocabularies**: Schema.org, SAREF, GSMA, ETSI can be mapped for the linked data context

Alberto Abella is Presenting

Smart Data Models Initiative structure

DATA-MODELS
- Guidelines for coding new data models
- Template for new data models and examples
- Directory for scripting tools to check data models
- Inventory of domains and data models
- Inventory of attributes and terms
- @Context for json-ld

DOMAINS REPOSITORIES
Readme pointing to the list of subjects
General info or shared resources



SUBJECTS' REPOSITORIES
Readme pointing to the list of data models for the objects
Contributors.md
subject-schema.json

DATA MODELS
README.md
/doc/spec.md
/examples
schema.json
Current Adopters

List of Smart Data Models: https://smartdatamodels.org/index.php/list-of-data-models-3/

Source: FIWARE Foundation, [Smart Data Models Initiative]

Process for Creating NGSI-LD Compatible Data Model

- NGSI-LD Data Models have to be compatible with the NGSI-LD Information (Meta) Model.
- NGSI-LD expects the specification of terms mapped to URIs for
  - Entity types
  - Property names
  - Relationship names

which are defined in the @context used in NGSI-LD requests

- While this is the minimum requirement, it makes sense to more explicitly define which properties and relationships an entity with a certain entity type can have.
- The Smart Data Model define this in form of a JSON schema (which can be used by applications to verify NGSI-LD data)
- Alternatively, the Data Model can also be defined as an Ontology, enabling the definition of type hierarchies, ranges for relationships and further conditions.

Smart Data Model Schema Example

- The smart data model separately describes each entity type
  - **Entity type** (e.g. *building*) with the list of its
  - **properties** and a description – in case of
  - **enumerations** (e.g. *category*) with the possible values
  - and the **required property** (i.e. all entity instances MUST have at least these properties)
  - Finally, there is a **textual description**

Entity: Building

Open License

Global description: **Information on a given Building**

List of properties

•address: The mailing address.
•alternateName: An alternative name for this item
•areaServed: The geographic area where a service or offered item is provided
•category: Category of the building. Enum:'apartments, bakehouse, barn, bridge, bungalow, bunker, cathedral, cabin, carport, chapel, church, civic, commercial, conservatory, construction, cowshed, detached, digester, dormitory, farm, farm_auxiliary, garage, garages, garbage_shed, grandstand, …

Required properties

- address
- category
- id
- type

This entity contains a harmonised description of a Building. This entity is associated with the vertical segments of smart homes, smart cities, industry and related IoT applications. This data model has been partially developed in cooperation with mobile operators and the GSMA, compared to GSMA data model following changes are introduced the reference to BuildingType is removed, since BuildingType compared to category attribute does not introduce significant information. category attribute is required. openingHours is introduced following schema.org data model to allow fine-grained on building opening times. GSMA supported this as free text in the notes attribute (removed as well). refSubscriptionService is not supported, since SubscriptionService model is not supported currently

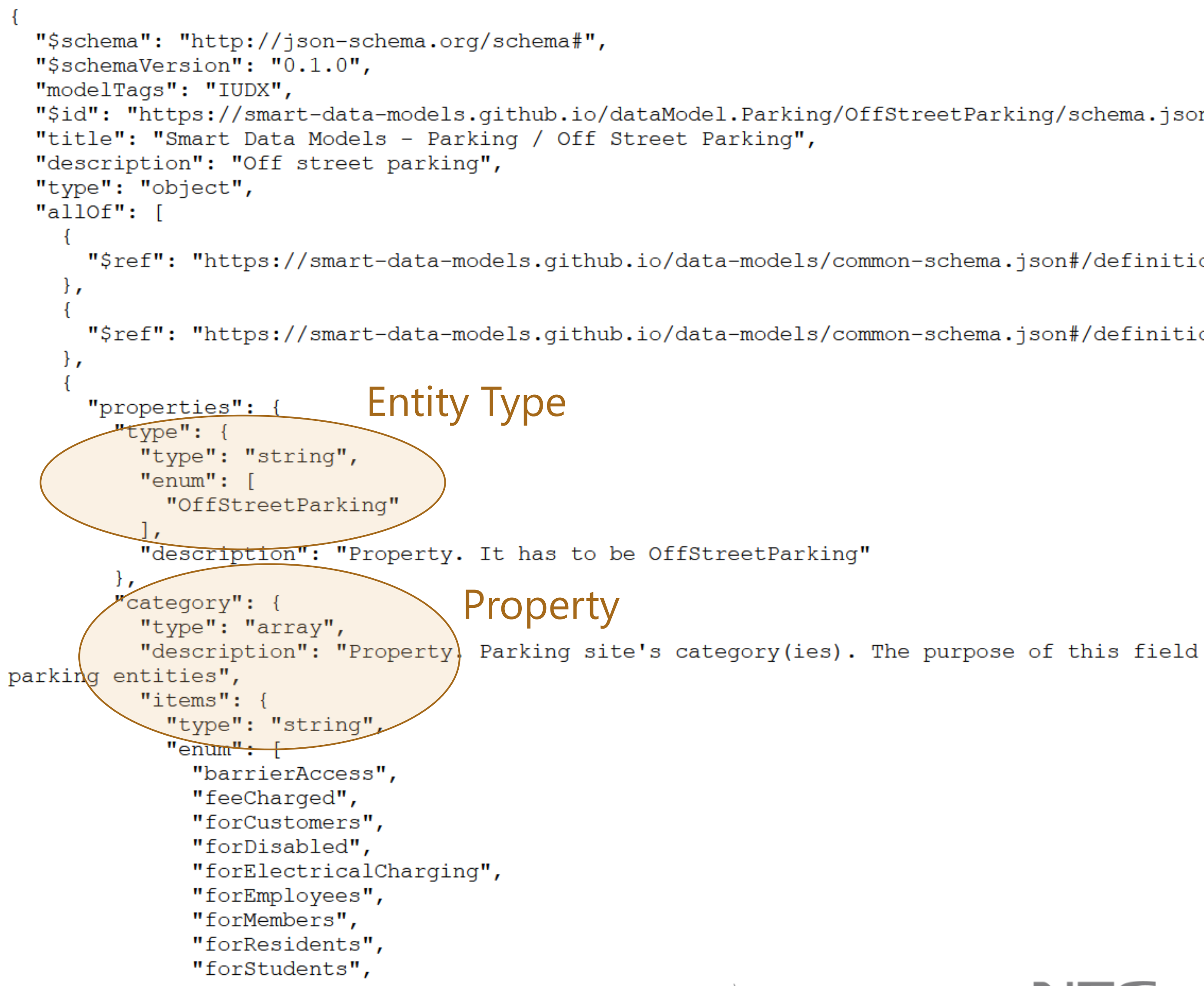
Smart Data Models as “Backbone Ontology”?

- Huge number of data models:
  - 717 official data models
  - 53 subjects
  - 12 domains
  - 15,400 terms defined
- Strong Community:
  - FIWARE, IUDX, OASC, tmforum + OCF, GSMA
  - 111 active contributors from 71 organizations
- Based on NGSI(-LD) Information Model
- Curated and homogenized
  - Well, mostly → devil is in the detail, see later

General Modelling Issues (JSON vs. OWL)

JSON Schema to OWL Ontology – Why can it be done in this case?

- **A JSON Schema** defines the **structure of a JSON document**, i.e. what elements it must (required) or may (optional) have.
- An **ontology** describes the **semantic model** of the information, i.e. what concepts are represented.
- In this case, we know that the **JSON schema describes an NGSI-LD document**, i.e. which Entities and Attributes (Properties or Relationships) must or may be contained
- As we know that it **follows the NGSI-LD information model** and where the respective information can be found, **we can extract the semantic information**.



```
{
  "$schema": "http://json-schema.org/schema#",
  "$schemaVersion": "0.1.0",
  "modelTags": "IUDX",
  "$id": "https://smart-data-models.github.io/dataModel.Parking/OffStreetParking/schema.json
  "title": "Smart Data Models - Parking / Off Street Parking",
  "description": "Off street parking",
  "type": "object",
  "allOf": [
    {
      "$ref": "https://smart-data-models.github.io/data-models/common-schema.json#/definitio
    },
    {
      "$ref": "https://smart-data-models.github.io/data-models/common-schema.json#/definitio
    },
    {
      "properties": {
        "type": {
          "type": "string",
          "enum": [
            "OffStreetParking"
          ],
          "description": "Property. It has to be OffStreetParking"
        },
        "category": {
          "type": "array",
          "description": "Property. Parking site's category(ies). The purpose of this field
parking entities",
          "items": {
            "type": "string",
            "enum": [
              "barrierAccess",
              "feeCharged",
              "forCustomers",
              "forDisabled",
              "forElectricalCharging",
              "forEmployees",
              "forMembers",
              "forResidents",
              "forStudents",
```

Differences in Modelling Assumptions: Arrays

- **Core Issue: JSON has the concept of Array, OWL (RDF Family) has not**
- Smart Data Models make use of Arrays a lot, also in connection with Enumerations
- Closest concept to Arrays in OWL: Lists

- **BUT: JSON-LD uses JSON Arrays???**
- Yes, to represent multiple instances
- Example: hasFriend has a single value in case there is one, the value is represented as a JSON array of values, if there are multiple
- However: You can specify that a value is a @List (rdf:List), in this case the value is always represented as a JSON Array in JSON-LD

→ **Array type is mapped to rdf:List (due to complexity without further restricting the elements)**

Differences in Modelling Assumptions: Enumerations (1)

- Smart Data Models make use of a lot of enumerations, in combination with string values, but also with Arrays
- OWL allows restricting data properties to a set of values, but the representation is somewhat more complex
- For string values, this has been implemented (enumeration values are also provided in comment)
- For arrays, I decided it was too complex → enumeration values are only explicitly provided in comment, so information is not completely lost

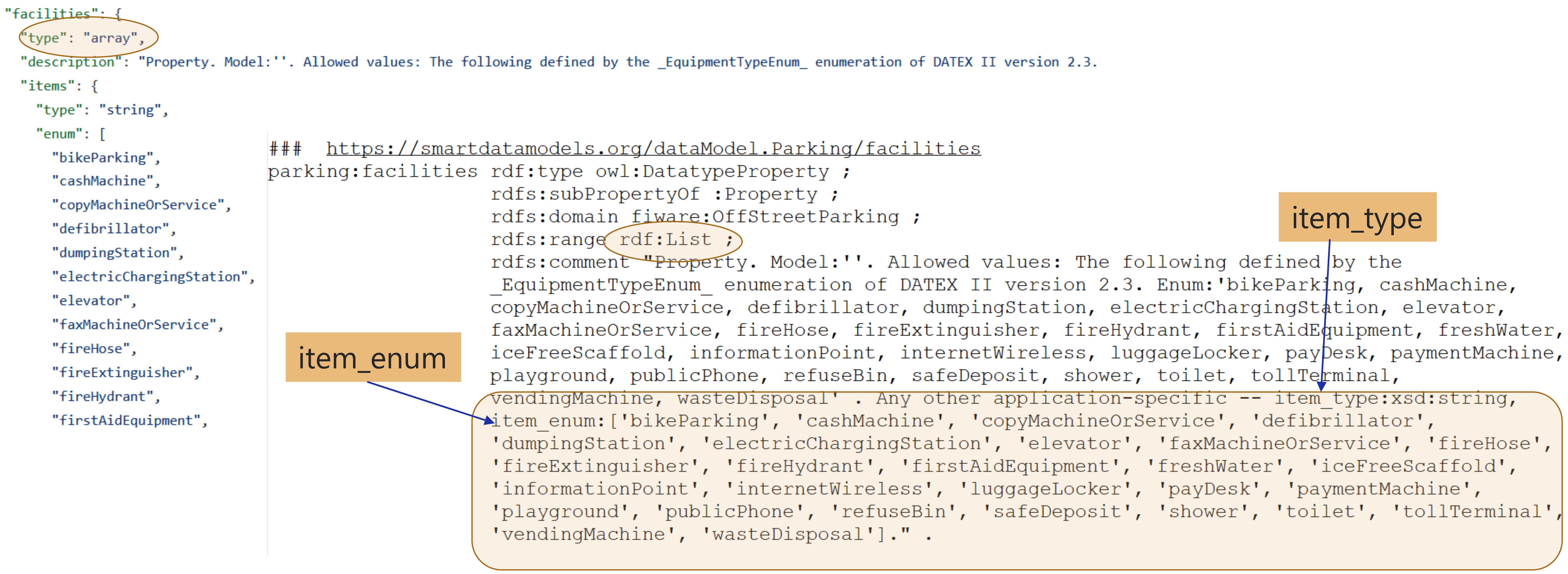
Differences in Modelling Assumptions: Enumerations (2)

- Enumeration for String Values

```
"reservationType": {
    "type": "string",
    "description": "Property. Conditions for reservation. Applications _SHOULD_ inspect the value of this property at parent's level
    "enum": [
        "mandatory",
        "notAvailable",
        "optional",
        "partly"
    ]
},
```

```
###  https://smartdatamodels.org/dataModel.Parking/reservationType
parking:reservationType rdf:type owl:DatatypeProperty ;
                        rdfs:subPropertyOf :Property ;
                        rdfs:domain [ rdf:type owl:Class ;
                                      owl:unionOf ( fiware:OffStreetParking
                                                    fiware:ParkingGroup
                                                  )
                                    ] ;
                        rdfs:range xsd:string ,
                                   [ rdf:type rdfs:Datatype ;
                                     owl:oneOf [ rdf:type rdf:List ;
                                                 rdf:first "mandatory"^^xsd:string ;
                                                 rdf:rest [ rdf:type rdf:List ;
                                                            rdf:first "notAvailable"^^xsd:string ;
                                                            rdf:rest [ rdf:type rdf:List ;
                                                                       rdf:first "optional"^^xsd:string ;
                                                                       rdf:rest [ rdf:type rdf:List ;
                                                                                  rdf:first
                                                                                  "partly"^^xsd:string ;
                                                                                  rdf:rest rdf:nil
                                                                                ]
                                                                     ]
                                                          ]
                                               ]
                                   ]  ;
```

Differences in Modelling Assumptions: Enumerations (3)



```
"facilities": {
  "type": "array",
  "description": "Property. Model:''. Allowed values: The following defined by the _EquipmentTypeEnum_ enumeration of DATEX II version 2.3.
  "items": {
    "type": "string",
    "enum": [
      "bikeParking",
      "cashMachine",
      "copyMachineOrService",
      "defibrillator",
      "dumpingStation",
      "electricChargingStation",
      "elevator",
      "faxMachineOrService",
      "fireHose",
      "fireExtinguisher",
      "fireHydrant",
      "firstAidEquipment",
```

```
###  https://smartdatamodels.org/dataModel.Parking/facilities
parking:facilities rdf:type owl:DatatypeProperty ;
                   rdfs:subPropertyOf :Property ;
                   rdfs:domain fiware:OffStreetParking ;
                   rdfs:range rdf:List ;
                   rdfs:comment "Property. Model:''. Allowed values: The following defined by the
                   _EquipmentTypeEnum_ enumeration of DATEX II version 2.3. Enum:'bikeParking, cashMachine,
                   copyMachineOrService, defibrillator, dumpingStation, electricChargingStation, elevator,
                   faxMachineOrService, fireHose, fireExtinguisher, fireHydrant, firstAidEquipment, freshWater,
                   iceFreeScaffold, informationPoint, internetWireless, luggageLocker, payDesk, paymentMachine,
                   playground, publicPhone, refuseBin, safeDeposit, shower, toilet, tollTerminal,
                   vendingMachine, wasteDisposal' . Any other application-specific -- item_type:xsd:string,
                   item_enum:['bikeParking', 'cashMachine', 'copyMachineOrService', 'defibrillator',
                   'dumpingStation', 'electricChargingStation', 'elevator', 'faxMachineOrService', 'fireHose',
                   'fireExtinguisher', 'fireHydrant', 'firstAidEquipment', 'freshWater', 'iceFreeScaffold',
                   'informationPoint', 'internetWireless', 'luggageLocker', 'payDesk', 'paymentMachine',
                   'playground', 'publicPhone', 'refuseBin', 'safeDeposit', 'shower', 'toilet', 'tollTerminal',
                   'vendingMachine', 'wasteDisposal']." .
```

Yes, allowed values are often already provided in the original description, BUT not always, and if it is provided, it is not always complete

Differences in Modelling Assumptions: Enumerations (4)

Approach, if we ever want to do a proper mapping of array enumerations: create own list type for each such array and then restrict data type of elements of this new list type → see example from stackoverflow: https://stackoverflow.com/questions/19474738/can-i-specify-a-range-for-rdflist-members/19480798#19480798

```
@prefix :      <http://stackoverflow.com/a/19480798/1281433/code#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

:rest   a             owl:ObjectProperty ;
        rdfs:domain  :List ;
        rdfs:range   :List .

:List   a       owl:Class .

:nil    a       owl:NamedIndividual , :ElementList , :List .

:ElementList  a          owl:Class ;
        rdfs:subClassOf  [ a                     owl:Class ;
                           owl:intersectionOf  ( :List [ a                  owl:Restriction ;
                                                         owl:allValuesFrom  :Element ;
                                                         owl:onProperty     :first
                                                       ] [ a                  owl:Restriction ;
                                                           owl:allValuesFrom  :ElementList ;
                                                           owl:onProperty     :rest
                                                         ] )
                         ] .

:Element  a     owl:Class .

:first  a             owl:ObjectProperty ;
        rdfs:domain  :List .

<http://stackoverflow.com/a/19480798/1281433/code>
        a       owl:Ontology .

[ a                      owl:Axiom ;
  rdfs:comment           "It's probably a good idea to specify that nil is an ElementList.  This could also
be inferred, though, if there is a nil-terminated List that is known to be an ElementList." ;
  owl:annotatedProperty  rdf:type ;
  owl:annotatedSource    :nil ;
  owl:annotatedTarget    :ElementList
] .
```

Unique Identifiers

- JSON Schemas are applied to NGSI documents (both NGSIv2 and NGSI-LD)
- Short terms are used in the JSON Schemas for defining the structure
- In NGSI-LD, JSON-LD is the basis, i.e. @context with mapping of short terms to URIs is needed in addition!

- Original approach: Smart Data Models provide a single *@context* for all datamodels together (still available) – many terms missing!!!
- New: *@context* for each Smart Data Model is provided
  - BUT: only attribute names are provided (i.e. no Entity Types – why?)

→ *@contexts* are used when creating OWL ontologies – typically old complete @context and then specific Smart Data Model @context (overwriting the original)

Relationships

- NGSI-LD has introduced explicit Relationships in addition to Properties
- Relationships logically relate to other Entities
- Entities are identified by their URI
- The Smart Data Models have different ways of specifying Relationships, often implicit
  - Prefix “ref” is used
  - Datatype EntityIdentifierType is referenced
  - Type definition is used inline
  - “Relationship” is specified in the description
- Whenever, Relationship can clearly be identified, it is modelled as such with range: Entity
- Else: Data Property of type URI

```
"EntityIdentifierType": {
  "anyOf": [
    {
      "type": "string",
      "minLength": 1,
      "maxLength": 256,
      "pattern": "^[\\w\\-\\.\\{\\}\\$\\+\\*\\[\\]`|~^@!,:\\\\]+$",
      "description": "Property. Identifier format of any NGSI entity"
    },
    {
      "type": "string",
      "format": "uri",
      "description": "Property. Identifier format of any NGSI entity"
    }
  ],
  "description": "Property. Unique identifier of the entity"
},
```

Does not make sense for NGSI-LD, which requires URI.

Jupyter Notebook: Creating OWL Ontologies from Smart Data Models **+ Demo**

https://gitlab.neclab.eu/darp-git-KI/CIU-AJG21

High-level Structure of Implementation

- Definition of constants
- Helper functions
- **Reading @context from URL**
- **Reading and parsing NGSI-LD Base Ontology**
- **Reading Smart Data Models**
- **Printing Resulting Ontology**
- **Main Function with configurations**

Single Ontology vs. Multiple Ontologies

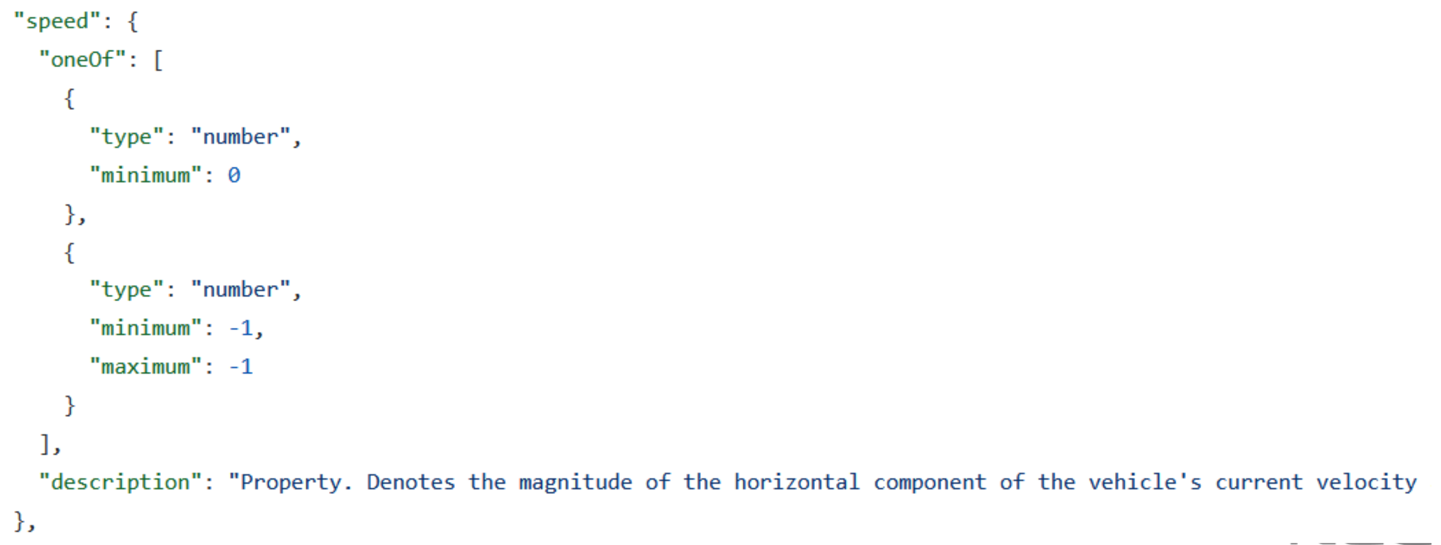
- Original idea: single ontology
- Can be done, but there are the following issues:
  - Some inconsistencies and errors in the data models, i.e. translation not always perfect
  - Examples
    - *status* property is always defined to have string values, but enumeration values differ between data models
    - some properties are modelled differently between smart data models (as array or as string)
  - Separate @contexts (more complete than integrated), each using own URI prefix
- New Proposal:
  - create multiple ontologies, one per data model, and integrate those
  - Additional advantage: only integrate what you really want to have

Limitations and specific issues

- No automatic extraction of List of data models, but can easily be extracted by hand from here: https://smartdatamodels.org/index.php/list-of-data-models-3/
- Not all types of restrictions have been implemented (array enums (see above), number range restrictions, etc.)
- Except for a few examples full URLs are used when referencing, relative URLs in very few places are currently not supported.
- Sometimes models are referenced for which there are no definitions included
- Some models have bugs, e.g. Ale

https://raw.githubusercontent.com/smart-

- Some details were too complex t
  "speed" → number



```
"speed": {
  "oneOf": [
    {
      "type": "number",
      "minimum": 0
    },
    {
      "type": "number",
      "minimum": -1,
      "maximum": -1
    }
  ],
  "description": "Property. Denotes the magnitude of the horizontal component of the vehicle's current velocity
},
```

\Orchestrating a brighter world

NEC creates the social values of safety, security,
fairness and efficiency to promote a more sustainable world
where everyone has the chance to reach their full potential.

\Orchestrating a brighter world

NEC