Profiling Standards to Improve Practical Interoperability

Francesca Noardo, Rob A. Atkinson, Alejandro Villar, Piotr Zaborowski, Ingo Simonis

Open Geospatial Consortium Europe, Technologielaan 3, 3001 Leuven Belgium
(fnoardo, ratkinson, avillar, pzaborowski, isimonis)@ogc.org

Keywords: Interoperability, Standard Profiles, Standard Data Models, Semantic Technologies

Abstract

Standard data models are key to enable a set of data integration functionalities, often characterised using the Findability, Accessibility, Interoperabilty and Reusability (FAIR) principles. However, standardisation is a process of trying to meet many requirements, and standard data models are inherently either very abstract or very comprehensive in the details. This results in several ambiguity pitfalls, inconsistent implementation of standard data models, which in turn hinders trust in the interoperability potential of standardised data, and complicates any integration processes. In practice profiling such standards is useful to overcome such issues to create more useful forms of standardised data for specific applications. However defining custom profiles typically requires a great deal of technical expertise in the underlying expression language of the standard. Maintaining access to this level of expertise is a challenge as profiles become outdated through the time and lose connection with the maintenance of the parent standard from which they originate. Therefore, in this paper, a scalable methodology is proposed, built on the OGC Building Blocks Model approach, that uses semantic modelling to support an easier composition of geospatial data models profiles which directly derive from available standards without losing the relevant dependencies that inform stakeholders which components are interoperable with other standards. The approach is tested within a digital building permit project (CHEK), in which data requirements derive from the semantics of city regulations and common geospatial standards (i.e., CityGML and INSPIRE) are used as reference.

1. Introduction

A growing number of applications in diverse fields need multi-source data integration and rely on data interoperability across multiple aspects of the integration process. Addressing this challenge is often characterised by the Findability, Accessibility, Interoperability and Reusability (F.A.I.R.) principles Wilkinson et al. (2016). An essential premise to realise the goals of these principles is the use of open standards. However, standards are intended to offer shared and general solutions to support multiple use cases, so that their scope often surpasses the needs of the single applications, and requires choices in implementation of details. In particular, standard data models, intended to describe an application domains’ information through agreed schemata to foster consistent and unambiguous data, are usually intended to cover the whole domain and a huge number of use cases. Often standards attempt to cover as much of the requirement space with a range of optional elements, and such elements can be extremely large (e.g., schema.org), yet never seem to cover all the actual application requirements.

Many aspects are represented which may not be relevant to a particular application, in an attempt to be a ”one-size-fits-all” solution (over-specification). It happens, therefore, that data models that attempt to be comprehensive both over-specify and under-specify the information needed to describe a use case’s needs (Noardo et al., 2024). Even so, at the same time, in an attempt to be generally applicable, definitions in the model may be oversimplified, covering one of many possible cases, or quite abstract to accommodate flexible needs, and precise semantics left to different implementation and presentation decisions (under-specification).

Such (necessary) freedom naturally results in implementation-specific choices that hinder the reusability and safe interpretation of standards-compliant data. Several datasets may represent a different selection of features with different modelling and filling choices but still remain compliant to the data model without being easily integrable.

Conventions for the content to populate these schemas, and any extensions, are necessary to apply the standards, but ad hoc profiling this way leads to potential for various incompatible solutions to similar requirements.

A standardised methodology to define application profiles from standards is therefore essential to support interoperability and effectiveness of standard data models. Profiles help identifying the required granularity of definitions and specifications to be extracted by the standard data models, as well as combining specifications from multiple schemes, declare rules for content and user guidelines, set additional constraints over the schema usage and syntax to suit the needs of specific application(Mourkoussis et al., 2006; Honma et al., 2013; Wuwongse, 2004; Chan and Zeng, 2006). For example, the field of Internet of Things has early recognised the need for experimenting with semantic profiles definitions, due to the high level of interoperability required by such application (Mazayev et al., 2017).

We also have examples from the domain of construction standards and representation, for which the buildingSMART Industry Foundation classes (IFC)[1] standard was developed starting from 1997. It is a very comprehensive schema intended to represent the information related to construction works, including several constuction elements and components as well as the information supporting the construction process itself (costs, actors and so on). In order to facilitate implementation and use

[1] https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-4-2024-359-2024 | © Author(s) 2024. CC BY 4.0 License.

of such an extensive model, it was supplemented with a standardised method to be properly profiled, which was the ’Model View Definition’ (MVD) standard[2] (Hietanen and Final, 2006). Recently, in the last versions of IFC, MVD was substituted by the Information Delivery Specification (IDS)[3] standard, to define semantic information requirements in machine readable format. Research exists about linked data to support it as well (van Berlo et al., 2019).

For geospatial standard data models, mostly expressed as XSD schemas because traditionally referring to GML format (e.g. CityGML[4], LandInfra[5], IndoorGML[6], INSPIRE data model[7]), although other encodings are being currently proposed for several of them (e.g. CityJSON), methods are proposed for adding extensions (Van den Brink et al., 2012). However, no methodology exists for formally profiling them, and such a gap hinders their high potential to support interoperability, as well as a reliable semantic validation process.

In this paper, a solution is described to define and manage standard profiles (Section 1.1) to enhance and improve standards effectiveness, for making them adhere to specific use cases requirements, as well as for enabling data validation against the defined profile, which enhances the users trust towards the standards and standardised data.

1.1 Standards profiles

Profiles of standard data models can provide simplified views by constraining and demonstrating implementation options. Moreover, profiles can also extend common patterns with application-specific capabilities. A key function, in this case, is the possibility to re-use other common standards and establish rules about how things inter-relate. A profile defines a set of constraints on a base specification. Implementations of profiles conform to the base specification. Because many technologies like JSON and RDF are permissive (by default) about additional information being present, definition of an extension is effectively defining a constraint on how additional information should be represented.

An automatic management of profiles allows all the underlying details of base standards to be automatically included in documentation, testing and validation, encapsulating the underlying complexity of base specifications. Development and usage of profiles, and, as a consequence, of standards, gets critically simplified, ensuring consistency and conformance of profiles with base specifications, and increasing standards effectiveness.

Profiles should be designed as well-documented and tested sets of constraints that can be, in turn, reused.

2. Design Approach

Before designing specific application models, the underlying design of the components and related requirements(Sadeghi et al., 2024) needs to be considered.

To realise interoperability in practice, three elements are essential for each interoperable component:

- A standard that describes a component well enough to support interoperability of applications
- An identifier for the standard that allows applications to understand which standard is in use
- A means to discover the relevant interoperability standards a component conforms to

If, and only if, all these elements are in place, applications can advertise the interoperability of available resources, clients can be configured to exploit them, and then discover when these capabilities can be applied to the resources they consume.

If we consider that resources are widely heterogenous, but have many common aspects, the overall problem is understanding how each aspect can be understood during data integration, rather than having bespoke integration models for every different resource.

Therefore, resource description involves describing how each component uses available standards. As we have seen, however, general standards often need to be specialised (profiled) to provide a single well know solution to a problem, therefore a complex application model, such as a data exchange schema or an API, needs to be able to reference the specific profiles of standardised components in use.

At this point, it becomes clear that there is a need for **referential transparency** - i.e., the ability to declare in a predictable way - the set of constraints. For example, as in the case of typical data exchange schemas, simply referencing the schema, or structural aspect of each component is insufficient.

On the other hand, developing an entire new way to describe composition of schemas is undesirable, and cannot be easily applied to existing approaches.

One way to address this is to define a composable ”building block” (Section 2.2) that carries implicitly both schema and additional constraints using a single identifier. As we shall describe, such a building block design can be expressed structurally using standard schema references, but also allow additional constraints and information to be carried into a composite specification (standard profile) and accessed by clients.

Once this meta-model for component composition is in place, it is possible to address other well known challenges in a standardised fashion. The concept of ”semantic interoperability” has been recognised as a key emerging challenge for information integration environments. Adding semantic description aspects to schemas is a natural extension of the building block description. For many years there has not been an available standard for how to do this. Past examples have proposed XML-based solutions to define application profiles schemas (Mourkoussis et al., 2006) However, Linked data technologies are now mature and represent the best option for implementation. They were identified as a solution to combine different data models and metadata schemes for the purpose of effective reuse of existing standard data models (Honma et al., 2013; Wuwongse, 2004). The emergence of JSON and JSON-LD provides a natural binding of schema elements to identifiers, which in turn can be used to identifier semantic definitions for these elements in an unambiguous way.

[2] https://technical.buildingsmart.org/standards/ifc/mvd/
[3] https://technical.buildingsmart.org/projects/information-delivery-specification-ids/
[4] https://www.google.com/search?client=safari&rls=en&q=citygml&ie=UTF-8&oe=UTF-8
[5] https://www.ogc.org/standard/infragml/
[6] https://www.ogc.org/standard/indoorgml/
[7] https://knowledge-base.inspire.ec.europa.eu/tools/inspire-data-models_en

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-4-2024-359-2024 | © Author(s) 2024. CC BY 4.0 License.

To exploit this opportunity, it must be possible to compose JSON-LD specifications at the same time as composing structural specifications using schemas. This is the key innovation that underpins the rest of the profile driven integration approach described here. Once the semantic meaning of structural elements are established, it becomes possible to define additional rules about data content and combinations using existing technologies, such as the Shapes Constraint Language (SHACL)[8].

Even more importantly, alternative data sources with different structural schema can be understood to have the same semantics, and the same rules applied. These rules themselves may be profiles specifying data requirements for some application, such as automated compliance checks for aspects of digital building permit approval.

2.1 OGC Building Blocks to support profiles

A toolkit currently under development, and applied to the concept of ”OGC specification building blocks”, is leveraged to represent the specific profiles composition useful to support automation and interoperability within a digital building permit use case.

Building Blocks may be composed to create more complete models, extended to cover more cases, or constrained to standardised particular implementation choices. Applications will typically use combinations of all three approaches. In this paper we will focus on the extension and constraint, or ”profiling” approaches.

Profiles, defined as a set of constraints rules over more general standards, can be implemented through the Specification Building Blocks[9] defined by the Open Geospatial consortium (OGC)[10] as a methodology to improve standards quality towards improved reusable and modular solutions.

The concept of OGC ’Specification Building Blocks’ originated through increasing need for modularity of specification design, and recognition that similar aspects were being addressed by inclusion of design elements from other specifications into different standards.

This approach, without specific identification of borrowed or repeated specifications modules or patterns, leads to three related ”scalability” limitations:

1. As the number of specifications grows it becomes harder to identify and determine which aspects are common, and therefore which aspects of different implementations are interoperable;

2. as the number of application domains increases in more complex systems, the number of different specification structures increases, compounding the commonality identification challenge for users;

3. As a set of specifications gets applied to more application domains over time, the variety of profile constraint expressions will grow, since there is no standard governing this, and every specification may define an alternative, or no, approach.

The OGC Building Block model (BBM) is a meta-model for specification that allows for human and multiple alternative machine-readable expressions, supporting the definition of different constraint approaches. For example, we have a JSON schema with semantic annotations (in RDF), on the one hand, and a set of SHACL shapes (i.e. rules about the data contents expressed based on the RDF language). In this way, you can define a constraint in JSON (according to the schema) or in RDF (according to the SHACL shapes), and apply the appropriate validation constraints over datasets to check their compliance. The goal is to make such constraints machine-readable to the extent possible. Constraints are defined in a form that allows for human-interpretable documentation but, most importantly, validation of test cases and examples.

The other key part of the Building Block Model consists of explicit dependencies and composition. ’Dependencies’ are intended as machine readable statements that provide traceability of what components are common. This may be illustrated in (Figure1) where the documentation of a Building Block clearly identifies that a ”Feature with topology” is a profile that extends a standard Feature using the JSON-FG profile (which adds explicit support for coordinate reference systems, time and feature typing). This extension component itself is another reusable element supporting geometry composed of references to lower order geometry elements, rather than duplicated sets of coordinates. Such a model can be incorporated into different data models as required, not just the GeoJSON model of a ”Feature”.

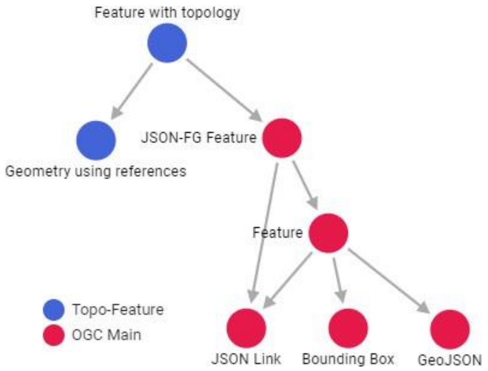


Figure 1. Profiling example by extension and composition of building blocks

The Building Block Model and its associated toolkit implements several other functions that may be extremely difficult to perform manually:

1. *Bundling* - i.e., composition of dependencies into a ready-to-use artefact;

2. *Continuous Integration/Testing/Deployment (CI/CT/CD)* - i.e., testing that all expressions conform to relevant languages and testing example data against machine-readable aspects of the specification;

3. *Transpiling* - i.e., compiling alternative versions of the specification, which are ready for use in different environments - such as the versions of the OpenAPI specification.

All these capabilities require a deep level of technical expertise in the underlying technologies, which is an unwanted burden on specification developers who should be focused on the requirements of their application domain.

[8] https://www.w3.org/TR/shacl/
[9] https://blocks.ogc.org
[10] https://www.ogc.org

By providing pre-packaged well implemented and tested modules (i.e., each OGC Building Block) the Building Block Model supports the process of standardisation by providing a simplified interface for higher-quality application-specific standards development. Standards and profiles built this way have inherent consistency, mitigating the three scalability issues identified above.

2.2 Semantic expressivity and interoperability

A key capability of the Building Blocks Model is the ability to bundle semantic annotations for schema fragments (i.e. from different standard data models) into a semantic model of an entire application schema composed of many parts.

The initial implementation of this capability uses existing semantic annotation standards, using JSON-LD contexts to link JSON schemas to vocabularies and ontologies defined in RDF (i.e., with explicit, unambiguous URI concept identifiers). This was used as the reference technology because, although other semantic annotation frameworks could be employed within the Building Blocks Model, this is nevertheless the only fully standards-supported option currently available.

Creating a JSON-LD mapping for a complex schema is not easy, since the mapping rules need to follow the schema structure to disambiguate identically named properties in different contexts. The JSON-LD tools available are inadequate to design and test complex JSON-LD contexts. However a bundling of simple JSON-LD contexts that are individually tested against simple schema-fragments is a significant enabler for semantic annotations for application schemas at realistic levels of complexity.

Once a profile has been defined with semantic annotations, then RDF representations of data can be generated, and SHACL and other validations can be performed. The Building Blocks Model allows definition of a knowledge base against which data can be tested. Thus, the Building Blocks Model can define a profile using a common schema and controlled vocabularies. Future developments will extend it to a more general mechanism to check online resources to ascertain data validity.

An example of this profiling using vocabulary choices is the set of ANZLIC jurisdiction profiles of a 3D Cadastre Survey Data Exchange Model[11]/ Each jurisdiction defines the set of controlled vocabularies it will use to implement the common profile for the region, which in turn defines common vocabularies and schemas for addressing.

Custom validators can be added to the validation workflow if required, and these can perform arbitrary checks on data - such as void detection in a 3D environment, which is not possible with available constraint languages.

In summary, if base specifications are based on (or described with) the OGC Building Blocks Model, many profiles can be built for specific applications, leveraging a centralised and shared effort in design, testing and validation capabilities through Continuous Integration/Continuous Testing/Continuous Deployment. This guarantees higher quality for standard profiles, on the one hand, and enhanced interoperability for the resulting schemas and compliant datasets, Moreover, profiles based on OGC Building Blocks also use the same structures as the underlying standards, so they can be possibly profiled in turn.

3. Use case application for digital building permit

The profiling technology was applied and tested to support a digital building permit use case, within the HORIZON ’Change Toolkit for Digital Building Permit’ (CHEK) project. The project investigates the digitalisation of building permits by means of digital datasets, as Building Information Models (BIM) and 3D city models or Geographical Information Systems. In both cases, the datasets needed for digital building permit automatic checks need to comply to specific data requirements. While the buildingSMART Information Delivery Specification is being investigated to define BIM data requirements based on infdustry Foundation Classes standard, the approach presented in this paper is being developed to profile and validate the semantics of 3D city models data. In the project, these were first manually defined within excel tables, to map the standards to the geodata needed to support building permit regulations checking.

Two standard data models were considered: CityJSON[12], which is the JSON implementation of the OGC CityGML standard[13], and the INSPIRE data model, provided as part of the INSPIRE European Union Directive[14]).

Moreover, as data needed to be specified further than the definitions in the data models, additional attributes were considered, referring to external standards, such as GeoDCAT[15], metadata standard, and even adding additional attributes as extensions.

3.1 CHEK CityGML-INSPIRE profiles implementation

For the implementation of CHEK data requirements, the OGC Data Exchange Toolkit[16] was used, which builds on the technology previously explained.

To be able to perform complex data validation rules checking that may span datasets across different formats, some preliminary steps were necessary. Input data is first converted into RDF, which allows accurately describing the data and describing links for entities across datasets, as a common metamodel, using a procedure called ”semantic uplift” (a process that can apply predetermined transformations to input data and embed JSON-LD context to semantically annotate it); figure 5 shows an example of what a GroundSurface CityGML element looks like throughout the uplift pipeline in CHEK. Additionally, the Shapes Constraint Language (SHACL)[17], a specification for defining validation constraints for RDF graphs, can be used to codify the necessary requirements.

The data requirements defined in the CHEK spreadsheets were translated to SHACL (usually with a 1-to-1 correspondence between requirements and SHACL shapes). This task was performed by using a specially designed web interface (Figure 3) that can generate not only the required SHACL shapes, but also bundle them as profiles using the RDF Profiles Vocabulary [18], each with a set of metadata (name or title, as well as any variables/parameters/arguments required, among others) to facilitate its discovery and use. Once in RDF format, profiles can be published in an instance of the OGC RAINBOW[19] (the full-spectrum semantic interoperability platform developed by the

[11] -https://icsm-au.github.io/3d-csdm-profile-icsm

[12] https://www.cityjson.org

[13] https://www.ogc.org/standard/citygml/

[14] https://knowledge-base.inspire.ec.europa.eu/index_en

[15] https://semiceu.github.io/GeoDCAT-AP/drafts/latest/

[16] https://github.com/ogcincubator/chek-profiles

[17] https://www.w3.org/TR/shacl/

[18] https://www.w3.org/TR/dx-prof/

[19] https://www.ogc.org/resources/rainbow/

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-4-2024-359-2024 | © Author(s) 2024. CC BY 4.0 License.

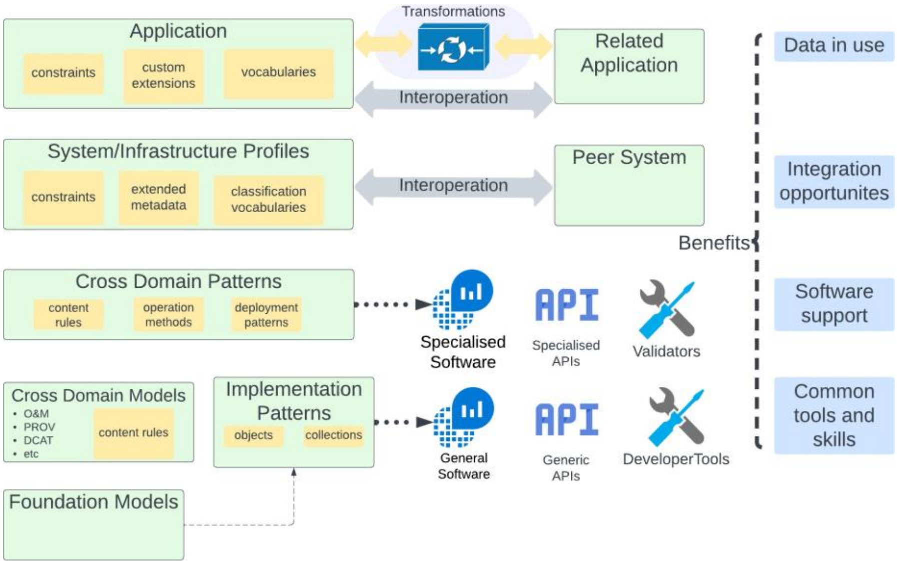
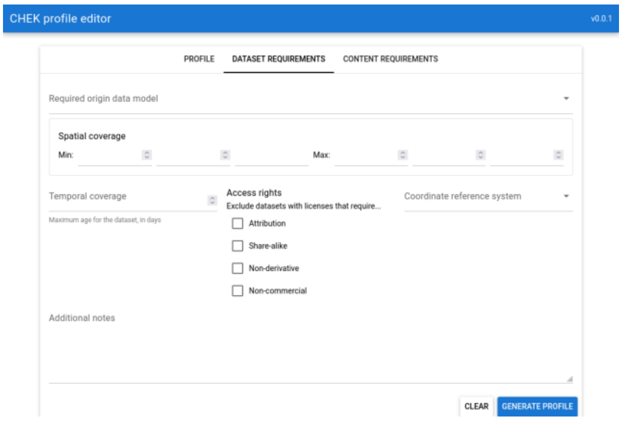


Figure 2. Profiling Building blocks methodology


CHEK profile editor v0.0.1

PROFILE DATASET REQUIREMENTS CONTENT REQUIREMENTS

Required origin data model

Spatial coverage
Min:
Max:

Temporal coverage
Maximum age for the dataset, in days

Access rights
Exclude datasets with licenses that require...
□ Attribution
□ Share-alike
□ Non-derivative
□ Non-commercial

Coordinate reference system

Additional notes

CLEAR GENERATE PROFILE

Figure 3. CHEK profiles definition interface

OGC) that, acting as a repository, will make them available for later use.

Figure 4 depicts the CHEK profile definition and validation methodology, starting from the encoding of standard data models into semantic format, as primary reference for the OGC Data Exchange Toolkit, complemented by the profile definition as SHACL-based data requirements and ending with the validation of datasets against them, once such datasets are converted from INSPIRE and CityGML into RDF.

3.2 Using the data requirements definition through standard profiles for data validation

A web service was developed to provide an endpoint through which data validation according to a standard profile, or set of profiles, could be performed. The service allows for the upload of datasets (in Figure 6, the data used for initial testing, related to the CHEK pilot case in Ascoli Piceno), the selection of profiles by their assigned identifier (URI), and, where applicable, the input of any parameters required by the rules (such as the location of interest), executing the following tasks:

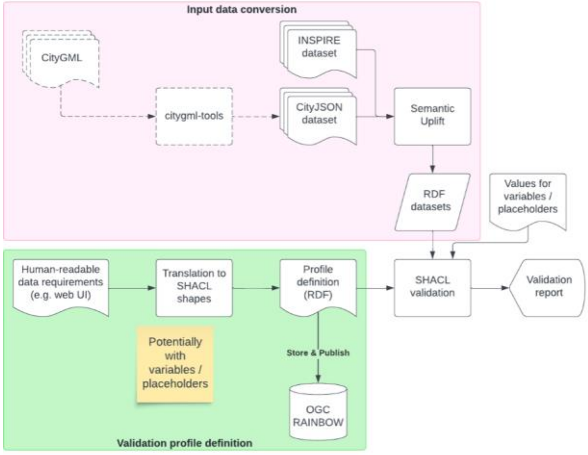


Figure 4. CHEK semantic uplift methodology

1. Profile resolution, obtaining both its metadata and set of SHACL shapes. This is done recursively, so for profiles that are specialisations of other profiles, their *parent* profiles are also fetched.

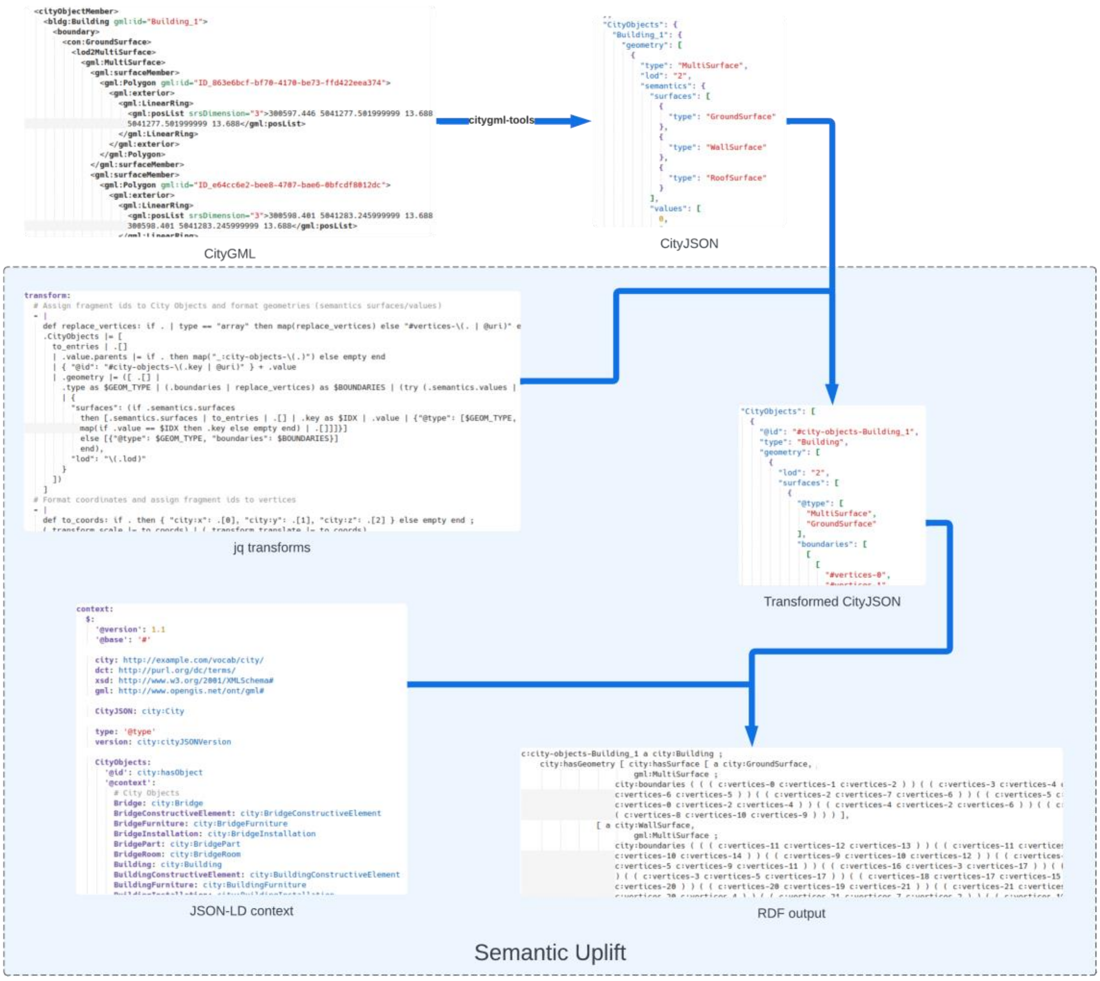
2. Data conversion.

3. Validation of the input data using the derived set of SHACL shapes.

4. Generation of a validation report, which is returned to the user as the result of the process.

The profile resolution mechanism allows for the definition of inheritance (or inclusion) chains for the rules, thus fostering reusability. It also enables profile creators to focus on distinctive or domain-specific features, while maintaining compatibility with more general specifications.

The service interface was developed following the OGC API -



```
<cityObjectMember>
  <bldg:Building gml:id="Building_1">
    <boundary>
      <con:GroundSurface>
        <lod2MultiSurface>
          <gml:MultiSurface>
            <gml:surfaceMember>
              <gml:Polygon gml:id="ID_863e6bcf-bf70-4170-be73-ffd422eea374">
                <gml:exterior>
                  <gml:LinearRing>
                    <gml:posList srsDimension="3">300597.446 5041277.501999999 13.688
                    5041277.501999999 13.688</gml:posList>
                  </gml:LinearRing>
                </gml:exterior>
              </gml:Polygon>
            </gml:surfaceMember>
            <gml:surfaceMember>
              <gml:Polygon gml:id="ID_e64cc6e2-bee8-4707-bae6-0bfcdf8012dc">
                <gml:exterior>
                  <gml:LinearRing>
                    <gml:posList srsDimension="3">300598.401 5041283.245999999 13.688
                    300598.401 5041283.245999999 13.688</gml:posList>
```

```
"CityObjects": {
  "Building_1": {
    "geometry": [
      {
        "type": "MultiSurface",
        "lod": "2",
        "semantics": {
          "surfaces": [
            {
              "type": "GroundSurface"
            },
            {
              "type": "WallSurface"
            },
            {
              "type": "RoofSurface"
            }
          ],
          "values": [
            0,
```

```
transform:
  # Assign fragment ids to City Objects and format geometries (semantics surfaces/values)
  - |
    def replace_vertices: if . | type == "array" then map(replace_vertices) else "#vertices-\(. | @uri)" e
    .CityObjects |= [
      to_entries | .[]
      | .value.parents |= if . then map("_:city-objects-\(.)") else empty end
      | { "@id": "#city-objects-\(.key | @uri)" } + .value
      | .geometry |= ([ .[] |
        .type as $GEOM_TYPE | (.boundaries | replace_vertices) as $BOUNDARIES | (try (.semantics.values |
        | {
          "surfaces": (if .semantics.surfaces
            then [.semantics.surfaces | to_entries | .[] | .key as $IDX | .value | {"@type": [$GEOM_TYPE,
            map(if .value == $IDX then .key else empty end) | .[]]]}]
            else [{"@type": $GEOM_TYPE, "boundaries": $BOUNDARIES}]
            end),
          "lod": "\(.lod)"
        }
      ])
    ]
  # Format coordinates and assign fragment ids to vertices
  - |
    def to_coords: if . then { "city:x": .[0], "city:y": .[1], "city:z": .[2] } else empty end ;
```

```
"CityObjects": [
  {
    "@id": "#city-objects-Building_1",
    "type": "Building",
    "geometry": [
      {
        "lod": "2",
        "surfaces": [
          {
            "@type": [
              "MultiSurface",
              "GroundSurface"
            ],
            "boundaries": [
              [
                [
                  "#vertices-0",
```

```
context:
  $:
    '@version': 1.1
    '@base': '#'

    city: http://example.com/vocab/city/
    dct: http://purl.org/dc/terms/
    xsd: http://www.w3.org/2001/XMLSchema#
    gml: http://www.opengis.net/ont/gml#

    CityJSON: city:City

    type: '@type'
    version: city:cityJSONVersion

    CityObjects:
      '@id': city:hasObject
      '@context':
        # City Objects
        Bridge: city:Bridge
        BridgeConstructiveElement: city:BridgeConstructiveElement
        BridgeFurniture: city:BridgeFurniture
        BridgeInstallation: city:BridgeInstallation
        BridgePart: city:BridgePart
        BridgeRoom: city:BridgeRoom
        Building: city:Building
        BuildingConstructiveElement: city:BuildingConstructiveElement
        BuildingFurniture: city:BuildingFurniture
```

```
c:city-objects-Building_1 a city:Building ;
    city:hasGeometry [ city:hasSurface [ a city:GroundSurface,
                gml:MultiSurface ;
            city:boundaries ( ( ( c:vertices-0 c:vertices-1 c:vertices-2 ) ) ( ( c:vertices-3 c:vertices-4
            c:vertices-6 c:vertices-5 ) ) ( ( c:vertices-2 c:vertices-7 c:vertices-6 ) ) ( ( c:vertices-5 c:
            c:vertices-0 c:vertices-2 c:vertices-4 ) ) ( ( c:vertices-4 c:vertices-2 c:vertices-6 ) ) ( ( c:
            ( c:vertices-8 c:vertices-10 c:vertices-9 ) ) ) ],
          [ a city:WallSurface,
                gml:MultiSurface ;
            city:boundaries ( ( ( c:vertices-11 c:vertices-12 c:vertices-13 ) ) ( ( c:vertices-11 c:vertice:
            c:vertices-10 c:vertices-14 ) ) ( ( c:vertices-9 c:vertices-10 c:vertices-12 ) ) ( ( c:vertices-
            c:vertices-5 c:vertices-9 c:vertices-11 ) ) ( ( c:vertices-16 c:vertices-3 c:vertices-17 ) ) (
            ) ( ( c:vertices-3 c:vertices-5 c:vertices-17 ) ) ( ( c:vertices-18 c:vertices-17 c:vertices-15
            c:vertices-20 ) ) ( ( c:vertices-20 c:vertices-19 c:vertices-21 ) ) ( ( c:vertices-21 c:vertice:
```

Figure 5. Uplift example for a GroundSurface instance in CityGML

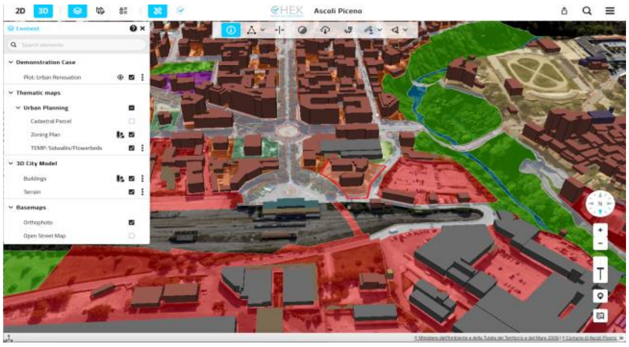


Figure 6. Validated 3D city model dataset for the CHEK pilot in Ascoli Piceno, in the VirtualCitySystems viewer

Processes specification[20], enabling third party consumers and applications to easily integrate with it.

An additional endpoint is provided for consumers to query the set of predefined CHEK profiles. The service retrieves from a preconfigured RAINBOW instance, making profile selection easier for users.

The use of the tool for validating data ensures that data can be unambiguously read and understood by software tools and the resulting analysis are reliable. This is even more important for the use case of digital building permits, for which the legal implications play a relevant role. Therefore, the demonstration of data reliability is essential to allow an actual operational uptake of digital and automatic solutions.

4. Conclusion and future works

In this paper the rationale and methodology underlying the OGC Building Blocks Model approach to profiling is explained as a means to simplify integration using standards for specific components of data models. This informed the development of a profiling tool to define modular application specific suites of simplified profiles for complex standard data models. Developing complete application data models, composed by formally defined bundles of modular standard data models profiles brings clear advantages in terms of underlying standard quality and sustainability. Explicitly capturing and testing compliance against dependencies allows taking into account any changes of upstream standards from which the profiles are referred.

Data requirements specified by such profiles in this way can be used to support data validation and ensure that reliable results come from analysis and processing using the data themselves. It allows therefore leveraging the general forms of standard data models to support interoperability and reusability of data at a finder level of detail. Such transformations into a semantically explicit model also allow Linked Data technologies to support connections and integration between different types of information not easily handled in a single system. This allows discovery and integration of, for example, geoinformation and Building Information Models (BIM) for purposes of further applications in the construction field.

The approach for data requirements definition and validation was tested in a case study for a project on building permits digitalisation (CHEK), with a dataset from Ascoli Piceno, Italy, using profiles of CityGML/CityJSON and INSPIRE data models.

It sets an essential milestone for enabling reliability in standardised data (re-)use for automatic analysis and workflows, integration and exchange.

In the next steps, other relevant building blocks, such as those related to the topology representation (Figure1), or provenance, could be used to extend the suite of data profiles to enable enhanced functionalities for additional requirements.

Finally, encoding relevant data requirements into SHACL rules was initially done manually, which required both knowledge of the RDF representation of the input data and the ability to craft SHACL validation shapes with varying complexity. The same issues will be present for any additional rules languages that can be applied. Therefore, a web form has been developed to supersede the simple spreadsheets described, to make rule definition easier for stakeholders that may not be expert developers. A web form can incrementally prompt for related information that will support generation of the required SHACL rules automatically. For any such mechanisms the goal is ensuring total consistency of rule expression with standardised data models through automatic linking and selection of schema elements through URIs.

ACKNOWLEDGEMENTS

This work has been supported by the CHEK project, which receives funding from the European Union's Horizon Europe programme under Grant Agreement No.101058559'

References

Chan, L. M., Zeng, M. L., 2006. Metadata interoperability and standardization–a study of methodology part I. *D-Lib magazine*, 12(6), 1082–9873.

Hietanen, J., Final, S., 2006. IFC model view definition format. *International Alliance for Interoperability*, 1–29.

Honma, T., Nagamori, M., Sugimoto, S., 2013. Find and combine vocabularies to design metadata application profiles using schema registries and lod resources. *International Conference on Dublin Core and Metadata Applications*, 104–114.

Mazayev, A., Martins, J. A., Correia, N., 2017. Interoperability in IoT through the semantic profiling of objects. *Ieee Access*, 6, 19379–19385.

Mourkoussis, N., Patel, M., White, M., 2006. A framework for the implementation of Application Profiles in XML Schemas. *Journal of Digitial Information*, 7(2).

Noardo, F., Atkinson, R., Simonis, I., Villar, A., Zaborowski, P., 2024. Ogc data exchange toolkit: Interoperable and reusable 3d data at the end of the ogc rainbow. T. H. Kolbe, A. Donaubauer, C. Beil (eds), *Recent Advances in 3D Geoinformation Science*, Springer Nature Switzerland, Cham, 761–779.

Sadeghi, M., Carenini, A., Corcho, O., Rossi, M., Santoro, R., Vogelsang, A., 2024. Interoperability of heterogeneous Systems of Systems: from requirements to a reference architecture. *The Journal of Supercomputing*, 80(7), 8954–8987.

[20] https://ogcapi.ogc.org/processes/

van Berlo, L., Willems, P., Pauwels, P., 2019. Creating information delivery specifications using linked data. *36th CIB W78 Conference*, 647–660.

Van den Brink, L., Stoter, J., Zlatanova, S., 2012. Modeling an application domain extension of CityGML in UML. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38, 11–14.

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E. et al., 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data*, 3(1), 1–9.

Wuwongse, V., 2004. Towards a language for metadata schemas for interoperability. *International Conference on Dublin Core and Metadata Applications*.