ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XII-4/W1-2026
21st International 3D GeoInfo Conference 2026, 28–29 September 2026, Sofia, Bulgaria

Representation of CityGML 3.0 Dynamizers in Web Streaming Formats for 4D Urban Digital Twins

Joseph Gitahi[1]*, Thomas H. Kolbe[1]
[1] Chair of Geoinformatics, Technical University of Munich, 80333 Munich, Germany
(joseph.gitahi, thomas.kolbe)@tum.de

Keywords: CityGML 3.0 Dynamizers, OGC 3D Tiles, Urban Digital Twins, 4D Streaming

Abstract

The integration of static 3D city models with temporal data, such as IoT sensor data, is essential for monitoring and analysing highly dynamic changes in Urban Digital Twins (UDTs). CityGML 3.0 introduces the Dynamizer module, which formalises the integration of semantic 3D city models and temporal data from various sources. However, CityGML datasets cannot be visualised directly on the web and require conversion to web streaming formats, which do not natively support temporal data integration. This research presents a methodology for transferring data structures defined in CityGML 3.0 Dynamizers into web streaming formats to enable 4D web visualisation in UDTs. We demonstrate representation strategies for various Dynamizer classes within the OGC standard 3D Tiles using structured metadata, define the required metadata for web viewers to discover and interpret Dynamizers, and outline optimisation techniques for efficient rendering of city-scale 4D scenes. A prototype 4D web viewer validates the approach, providing interactive visualisation of integrated semantic 3D models and high-resolution temporal data.

1. Introduction

City systems consist of components, including the built environment, natural environment, technological systems, and citizens, all of which interact in intricate and dynamic ways. The complexity of these interactions, coupled with the rapid growth of cities worldwide, poses challenges such as increased resource demand, strained infrastructure, and environmental pollution. To address these challenges, data-driven solutions become necessary for sustainable urban planning and development. Among the data-driven technologies in use are Urban Digital Twins (UDTs), which are virtual replicas of cities that employ digital 3D city models to represent the physical built environment, along with temporal data from Internet of Things (IoT) sensors, analytics and simulations for real-time updates and historical analysis. By integrating digital 3D models with temporal data, UDTs provide a comprehensive understanding of complex urban environments, supporting applications including city infrastructure monitoring, environmental monitoring, predictive analytics, simulation of what-if scenarios, and control of city infrastructure, such as traffic signalling (Jeddoub et al., 2024, Ferré-Bigorra et al., 2022, Faliagka et al., 2024).

Users interact with UDTs through visual interfaces built with game engines, desktop applications, or web applications. Game engines like Unity and Unreal offer high visual fidelity, enabling photorealistic visualisations of urban environments and advanced interaction capabilities, such as immersive AR/VR experiences (Crampen et al., 2024, Rantanen et al., 2023). However, these applications have high graphics processing requirements, necessitating powerful hardware. In contrast, web-based visualisations have lower technical barriers, as they only require modern browsers to run. This enables cross-platform compatibility and accessibility across multiple devices, thereby providing access to a diverse range of urban stakeholders. Web-based platforms further enable straightforward integration with diverse spatial datasets via standardised spatial data APIs, whereas game engines often necessitate custom pipelines and data conversion (Rantanen et al., 2023, Würstle et al., 2022). As web applications are hosted centrally, they allow seamless deployment of updates, streamlined support and maintenance, and enhanced scalability. These benefits make web-based applications a compelling choice for visualisation in UDTs.

Data integration is identified as a major challenge in creating UDTs (Ali et al., 2024, Lei et al., 2023, Weil et al., 2023). Among the efforts to address this challenge is the Dynamizers module, introduced in the Open Geospatial Consortium (OGC) CityGML 3.0 standard, that formalises the linking of time-dependent properties of 3D city objects to data sources, such as IoT platforms, databases and external files (Kutzner et al., 2020). Although frameworks such as Dynamizers, CityThings (Santhanavanich and Coors, 2021) and ontology-based integration techniques (Chiang et al., 2020) enable integrating 3D city models and time series data streams at the modelling level, there are currently no standardised approaches for integration into 3D web streaming formats.

The proliferation of IoT devices across domains such as traffic, environment, and energy in cities presents an opportunity to integrate sensor data into semantic 3D city models to create UDTs. Concepts such as the CityGML 3.0 Dynamizers already provide a standardised mechanism for modelling time series data and linking them to time-dependent properties of 3D city models. However, these concepts are not directly supported by Web 3D streaming formats. As UDTs increasingly target citywide applications, the need to visualise spatial, thematic, and appearance changes over time makes efficient, large-scale 4D web visualisations essential.

* Corresponding author

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W1-2026-161-2026

The goal of this research is to develop a standardised representation of links between semantic 3D city models and temporal data sources for large-scale web-based 4D visualisations. This requires transferring structures that model temporal data in semantic 3D city models, such as CityGML Dynamizers, into existing 3D streaming and rendering formats, preserving the structures and semantics necessary for interpretation by web viewers. The research aims to (i) demonstrate representation of different Dynamizers types within the OGC 3D Tiles web streaming format, (ii) develop a framework for automatic discovery and interpretation of Dynamizers embedded in OGC 3D Tiles in web viewers, and (iii) develop optimisation strategies for efficient rendering of large-scale 4D web scenes. Within this scope, the research addresses the following questions:

1. How should the data structures defined by Dynamizers be modelled within existing 3D streaming and rendering formats to enable 4D web visualisations?
2. What information is required by web viewers to discover and interpret these temporal structures?
3. How do we efficiently represent semantic 3D models integrated with temporal data in large-scale 4D web-based visualisations?

2. Background and Related Work

2.1 Urban Data Modelling and Standardisation

Data standardisation and harmonisation are essential to ensure seamless interoperability when creating UDTs. The OGC CityGML standard provides a comprehensive and standardised framework for representing urban objects and exchanging semantic 3D city models. Semantic 3D models provide structured, data-rich digital representations of urban environments by integrating geometric information with semantic attributes that provide meaning and context to the represented objects (Kolbe et al., 2021). The standard includes various modules for representing city objects, including *Building, Bridge, Tunnel, City Furniture, Land Use, Transportation, Vegetation*, and *Waterbody*. CityGML also supports multiple Levels of Detail (LOD), allowing for basic and highly detailed representations of 3D city models. The embedded semantics and topological structures make these 3D city models suitable for urban analytics and simulations. In the current version, CityGML 3.0, several existing modules have been revised and new modules introduced, including *Dynamizer, Versioning, PointCloud*, and *Construction* (Kolbe et al., 2021, Kutzner et al., 2020). This version also separates the conceptual model and the encoding specifications, allowing for additional encoding formats beyond XML, such as the JSON-based CityJSON and the relational database schema 3DCityDB (Yao et al., 2018).

To achieve harmonised management and access to heterogeneous sensor data, standards such as the OGC SensorThings API (STA) provide a unified framework for integrating and managing IoT devices and data (Liang et al., 2016). This standard facilitates interoperability of interconnected IoT devices by employing a standard data model for data management and a RESTful API for accessing and querying IoT data. The native geospatial support and high scalability make it ideal for city-wide geosensor networks. FIWARE is another open-source platform used in smart cities, offering a data model based on the Next Generation Service Interface (NGSI) standard and standardised APIs to enable interoperability and the exchange of IoT data (Cirillo et al., 2019).

2.2 Semantic 3D City Models and Temporal Data Integration

Developing UDTs requires data integration to combine static semantic 3D city models with temporal data from IoT sensors, simulations, or analytical results within a unified platform. The Dynamizer module, introduced in CityGML 3.0, provides a framework for linking time-dependent attributes of semantic 3D city models to temporal data sources, such as IoT platforms, databases, or external tabular files. The Dynamizer class has three core attributes: *attributeRef*, which references a specific time-dependent attribute of a 3D object, and *startTime* and *endTime*, which represent the period of dynamic data (Chaturvedi, 2021, Kutzner et al., 2020). As shown in Figure 1, the dynamic data can be modelled in two ways. First, time-dependent attributes can be directly linked to sensor data sources, such as IoT platform APIs, using the *SensorConnection* data type. The *SensorConnection* includes information on connecting and retrieving dynamic data from external APIs. The connection details include the URI, authentication parameters, the type of sensor API (e.g., OGC SensorThings API 1.1), networking protocols, the unique identifier of the required data stream, and semantic information about the sensor data.

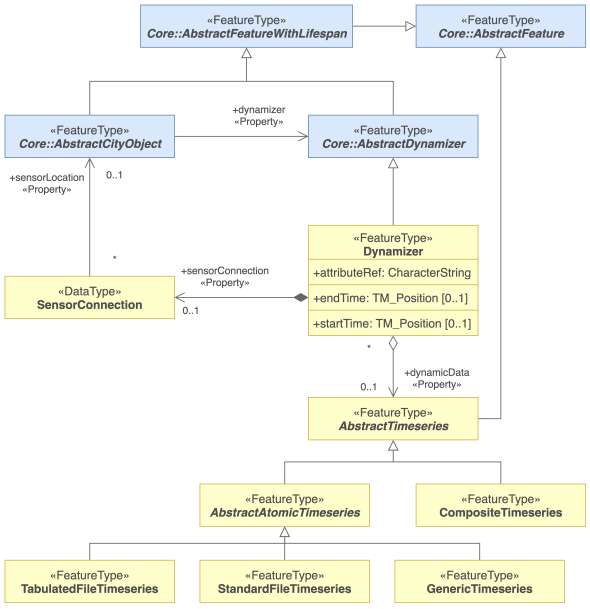

Figure 1. The UML diagram illustrates the components of the CityGML Dynamizer module, which are highlighted in yellow.

The second approach involves modelling temporal data within Dynamizer features through the *AbstractTimeseries* class. This class is further subdivided into two subclasses: *AtomicTimeseries*, which represents tabulated timestamps and values, and *CompositeTimeseries*, which

handles timestamps and values that repeat, such as those obtained from statistical analysis or simulation. In the *AtomicTimeseries*, tabulated time series data can be represented as a *StandardFileTimeseries* using open standards. The data may also be represented as a *TabulatedFileTimeseries*, referencing external tabular files such as CSV and Microsoft Excel Spreadsheets. Alternatively, time-value pairs can be stored within a CityGML dataset using the *GenericTimeseries* subclass.

2.3 3D Geospatial Visualisation Standards

Several established standards and technologies exist that enable 3D spatial web visualisations, ranging from simple interactive graphics to complex 3D scenes. The *Graphics Library Transmission Format (glTF)* is an open standard for efficiently transmitting and rendering 3D models on web platforms and gaming engines (Khronos Group, 2021, Robinet et al., 2018). It enables 3D object representation through a JSON-based structure that includes geometry, node hierarchy, materials, animations, cameras, and metadata. While glTF is effective for 3D visualisation, it lacks native support for georeferencing or Level of Detail (LOD) hierarchies. To bridge the gap between 3D visualisation and spatial data integration, other visualisation standards extend glTF with spatial hierarchy, geospatial coordinates, and spatial indexing, to enable efficient streaming and georeferencing of large-scale geospatial datasets.

The *OGC 3D Tiles* specification created by Cesium is designed for streaming and rendering large-scale heterogeneous 3D geospatial content in web, mobile, and desktop applications (Cozzi et al., 2019). To efficiently transmit and visualise 3D datasets, 3D Tiles utilises progressive loading, hierarchical LOD streaming, and tiling techniques. These approaches ensure that only the data and details within the viewer's map perspective and zoom level are displayed. The specification supports visualisation of diverse spatial datasets, including mesh models, point clouds, 3D city models, and terrain, making it ideal for UDT visualisation. 3D Tiles support both geometric and semantic information, including textures and attributes. This allows for accurate spatial placement of 3D content, detailed visual properties, and contextual information. The current version, 3D Tiles 1.1, enhances the standard by supporting embedded metadata at various granularities using the *3D Metadata Specification*, improving integration with the glTF ecosystem, and introducing implicit tiling (Cozzi and Lilley, 2023).

The metadata specification provides a standardised format for structured thematic data in 3D content. It is also extensible, allowing the incorporation of domain-specific semantics. The specification is implemented in two ways: *3D Tiles Metadata*, which assigns metadata to tilesets, tiles, and contents, and *EXT_structural_metadata*, which assigns metadata to features, vertices, and texels in glTF 2.0 assets. In both approaches, the metadata structure is defined using a JSON-based schema, which is stored within a tileset, a glTF asset or referenced externally.

The metadata schema structure consists of classes, properties and enums. Each class describes an entity type, such as a building, and specifies a set of associated properties, such as name and address. Properties define the metadata for their values, including attributes such as name, description, and default values. Property definitions may also include semantic identifiers to formally specify their meaning and interpretation. These identifiers can be used to establish explicit semantic references between properties and their corresponding domain-specific definitions. Figure 2 shows the UML diagram of the *EXT_structural_metadata* schema components.

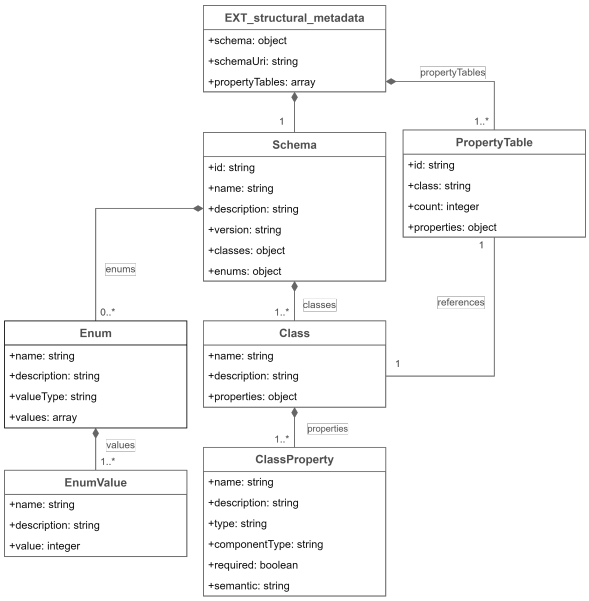

Figure 2. EXT_structural_metadata schema.

The *OGC Indexed 3D Scene Layer (I3S)*, developed by ESRI, is another visualisation standard for streaming 3D geospatial content (Reed and Belayneh, 2023). It encodes heterogeneous 3D geospatial data using JSON and binary buffers, which are stored in I3S scene layers for streaming. Like OGC 3D Tiles, I3S supports hierarchical LOD, streaming, and progressive loading to enhance rendering efficiency and performance. In the standard, semantic attributes are organised in feature attribute tables and linked to features at each hierarchical node.

3. Methodology

The proposed methodology aims to bridge the gap between integrating temporal data into semantic 3D city modelling and representing temporal properties in 4D web visualisation within the context of UDTs. 3D spatial web streaming specifications, such as OGC 3D Tiles and OGC I3S, efficiently deliver massive, heterogeneous 3D models. However, these standards lack a structured semantic model for representing high-resolution time series data, such as IoT sensor data. Consequently, there is no standard way to discover and interpret data structures that connect 3D city models with temporal data, such as those defined by CityGML Dynamizers. Additionally, these visualisation specifications focus on geometric and appearance optimisations using tiling, LoD, and progressive streaming, but these optimisation techniques do not support querying, updating or storing temporal data

in 3D city models. These limitations hinder the transferability of 4D visualisation workflows across UDT platforms and the scalability to city-wide datasets. A generic framework is thus required to formally identify time-dependent properties of semantic 3D models, discover their temporal data sources and define update strategies. Furthermore, optimisation strategies must extend beyond spatial aspects to include temporal dimensions for efficient rendering in 4D visualisations. We present a methodology for standardised encoding of CityGML Dynamizers within the 3D Metadata Specification based on the schema defined by the *EXT_structural_metadata* extension.

3.1 Encoding CityGML Dynamizers in 3D Tiles

To enable standardised 4D web visualisation, the encoding of CityGML Dynamizers into web streaming formats must preserve the data structures that link the time-dependent properties of 3D city objects to their temporal data sources. Additionally, the encoding should be web-compatible and machine-readable to support automated discovery and interpretation by web applications. We propose a JSON-based data structure with a simple syntax that preserves the Dynamizer schema, can be stored as a property within web streaming formats, and is inherently suitable for web applications. The enhanced metadata integration in 3D Tiles 1.1, using the *3D Metadata Specification*, enables embedding structured semantic information within 3D city models. At the feature level, the glTF extension *EXT_structural_metadata* defines a schema for embedding properties of 3D city objects, such as those modelled in CityGML, specifying their data types and semantics. Although JSON is not a supported data type, it may be stored as a string property, which can then be parsed and decoded in web viewers.

In the proposed data structure, CityGML semantics are mapped to *EXT_structural_metadata* as follows: feature classes are represented as metadata classes, feature instances as property table rows, and feature attributes as metadata properties. Each property name is prefixed with the CityGML module's namespace, and semantic identifier URIs are included to formally link properties to CityGML schema definitions, thereby ensuring inherent semantic meaning. For each feature, a *"dyn_dynamizers"* property field is included. This field contains an array of Dynamizers, each represented as a JSON string, supporting zero or more time-dependent properties per feature. Dynamizers establish explicit links between time-dependent properties and their corresponding Dynamizers using the *attributeRef*, which is stored as an XPath in GML files. In the proposed JSON data structure, the resolved time-dependent property referenced by the *attributeRef* is stored directly. This approach enables web viewers to identify exactly which property is updated by a given Dynamizer without additional parsing.

As discussed in Section 2.2, the CityGML Dynamizer module provides multiple methods for integrating temporal data into 3D city models, which vary depending on the data source type and its characteristics. In the *dyn_dynamizers* array property, each Dynamizer is instantiated with the necessary attributes.

Listing 1. The code snippet demonstrates CityGML 3.0 semantics mapped to the *EXT_structural_metadata* schema. It includes a building object with a generic time-dependent attribute, *gen_temperature*, and an associated *SensorConnection* Dynamizer that links the attribute to a data stream of an OGC SensorThings API service.

```
{
"EXT_structural_metadata": {
   "schema": {
      "id": "citygml_3",
      "name": "CityGML 3.0",
      "classes": {
         "Building": {
            "name": "CityGML 3.0 Building Module",
            "properties": {
               "gml_id": {"name": "gml:id",
                 "type": "STRING",
                 "required": true,
                 "semantic": "https://schemas.opengis.net/
                      ↪ gml/3.2.1/gmlBase.xsd#id"
                 },
               "gen_temperature": {"name": "temperature",
                 "type": "SCALAR",
                 "componentType": "FLOAT64",
                 "semantic": "https://schemas.opengis.net/
                      ↪ citygml/generics/3.0/generics.xsd#
                      ↪ DoubleAttribute"
                 },
               "dyn_dynamizers": {
                  "name": "dynamizers",
                  "type": "STRING",
                  "semantic": "https://schemas.opengis.net/
                       ↪ citygml/dynamizer/3.0/dynamizer.
                       ↪ xsd",
                  "array": true
                 }
            }
         }}},
   "propertyTables": [
    {
      "class": "Building",
      "count": 1,
      "properties": {
         "gml_id": {"values": ["DEBY_LOD2_49XYZ"]},
         "gen_temperature": {"values": [22.5]},
         "dyn_dynamizers": {
            "values": [[
               {
                 "dyn_startTime": "2025-01-01T00:00:00Z",
                 "dyn_endTime": "2025-05-31T23:59:59Z",
                 "dyn_attributeRef": "gen_temperature",
                 "dyn_SensorConnection": {
                    "dyn_authType": "None",
                    "dyn_connectionType": "ogc_sta_1.1",
                    "dyn_baseURL": "https://sta-server.com",
                    "dyn_datastreamID": "1",
                    "dyn_linkToObservation": "https://sta-
                         ↪ server.com/Datastreams(1)/
                         ↪ Observations?$filter=during(
                         ↪ phenomenonTime,2025-01-01T00
                         ↪ :00:00Z/2025-05-31T23:59:59Z)",
                    "dyn_linkToSensorDescription": "https://
                         ↪ sta-server.com/Sensors(5)",
                    "dyn_observationID": "2",
                    "dyn_observationProperty": "Temperature",
                    "dyn_sensorID": "5",
                    "dyn_sensorName": "GroveBME680",
                    "dyn_uom": "Celsius"
                       }
                 }]]
      }}}]}}
```

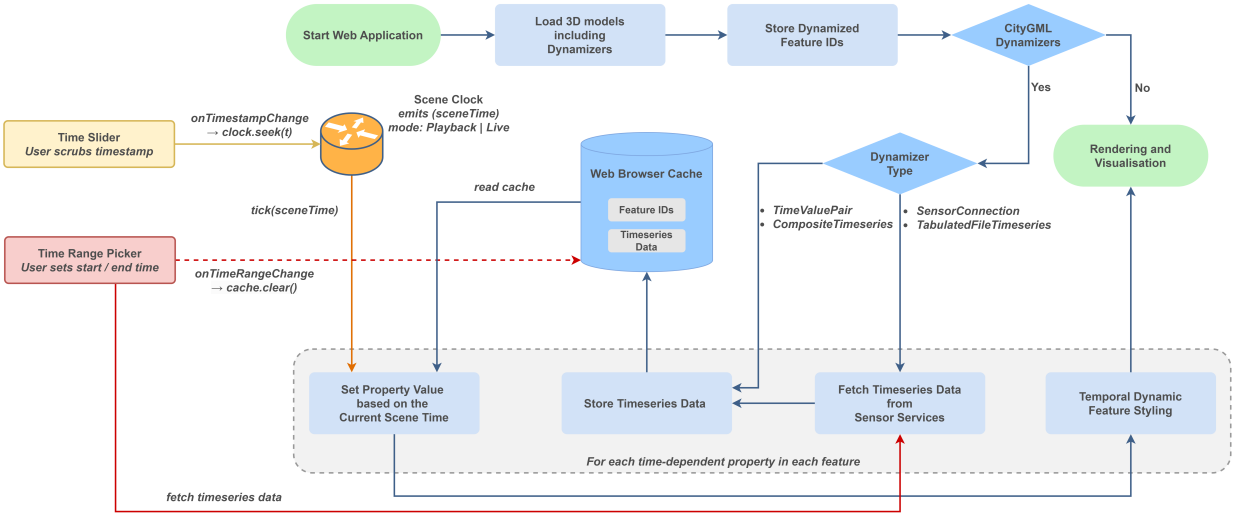

Figure 3. The proposed workflow for discovery and interpretation of CityGML Dynamizers in web viewers.

Listing 1 shows the proposed mapping of CityGML semantics to *EXT_structural_metadata*, which includes a *SensorConnection* Dynamizer. In this example, property values in the property table are stored in JSON for demonstration purposes. However, in practical applications, these values are typically stored in binary buffers optimised for high performance. The schema may be embedded directly within glTF files or referenced from an external file.

3.2 Discovery and Interpretation of CityGML Dynamizers in Web Viewers

The proposed methodology introduces a client-side interpretation logic that allows web applications to discover and interpret CityGML Dynamizers embedded within 3D web streaming formats. This logic relies on the structured metadata schema discussed in Section 3.1 to identify time-dependent feature properties and determine suitable update mechanisms. The workflow begins when a web viewer loads 3D city models together with the corresponding metadata schema, which specifies feature attributes and any associated Dynamizer information, as illustrated in the workflow diagram in Figure 3. For each feature, the viewer traverses through the attributes and checks for the presence of the Dynamizers array attribute, identified by the CityGML module prefix *"dyn"*. Upon discovery, the client processes the array of Dynamizers and determines their class for further interpretation, which may include a *TimeValuePair* series, a *CompositeTimeseries*, a *TabulatedFileTimeseries*, or a *SensorConnection*. For *TimeValuePair Dynamizers*, temporal values are stored directly in the web client storage, whereas for *CompositeTimeseries*, temporal data is first computed based on the temporal extents and patterns defined in the Dynamizers. For *TabulatedFileTimeseries* and *SensorConnection* Dynamizers, temporal data is retrieved from external sources and stored in the client's storage. Once the temporal data is available in client storage, the viewer updates each time-dependent attribute of every feature in real time, synchronising their values with the current scene timestamp. These updates facilitate real-time dynamic styling of 3D features to visualise temporal changes.

To ensure interactive performance in large-scale 4D urban digital twins, the methodology incorporates several optimisation techniques. For externally sourced data, such as *TabulatedFileTimeseries* and *SensorConnection* Dynamizers, temporal data is loaded on demand and cached. The client fetches and updates time series only for features currently present in the active scene, employing feature tracking to minimise unnecessary network requests for off-screen objects. Additionally, dynamic updates to time-dependent attributes and styling are limited to features within the user's view frustum. This approach enables 4D web visualisations to advance over time while recomputing only the affected attributes, thereby balancing temporal accuracy, rendering efficiency, and network resource usage.

4. Implementation

This proof of concept demonstrates the practical implementation of the proposed framework for encoding CityGML Dynamizers in web streaming formats and visualising temporal changes in UDTs. The methodology consists of two main steps: data preparation and the development of a 4D web visualisation application.

4.1 Data Preparation:

To demonstrate the various approaches to temporal data integration, three CityGML building datasets were enriched with different Dynamizer classes. These datasets were derived from the LOD2 building models released as open data by the Bavarian State Office for Digitising, Broadband and Survey (Bayerisches Landesamt für Digitalisierung, Breitband und Vermessung (LDBV), 2026). In the first dataset, three Dynamizers of type *SensorConnection* were integrated into a 3D building model of TUM Campus. These Dynamizers link three time-dependent indoor climate properties *(temperature, humidity, and CO_2)* to data streams in an OGC SensorThings API server, for real-time retrieval of sensor measurements.

The second dataset contained four buildings in Munich's Harthof district, each enriched with Dynamizers of type

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W1-2026-161-2026 | © Author(s) 2026. CC BY 4.0 License.

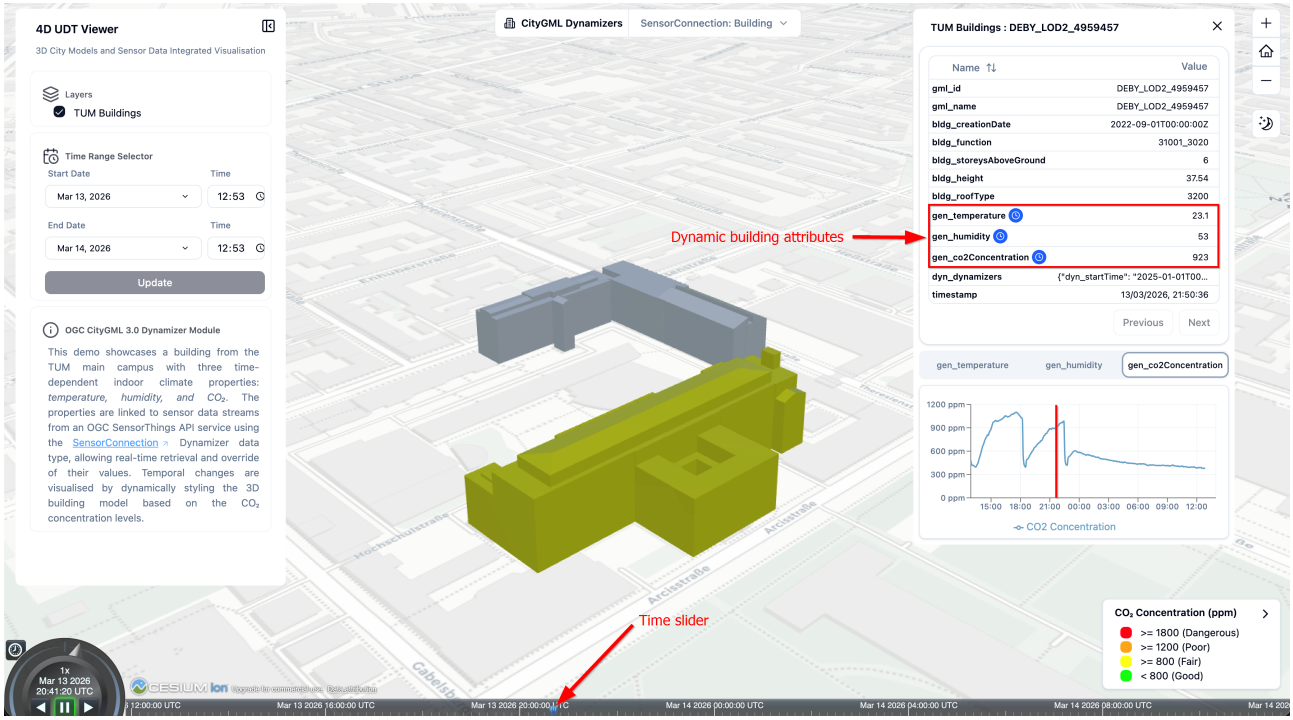

Figure 4. The 4D viewer application displays the UI controls, data panels, and a building model featuring time-dependent indoor climate attributes, which are indicated by a blue clock icon in the properties table.

GenericTimeseries. These Dynamizers embedded pre-computed time-value pairs of global solar irradiation, aggregated monthly over one year.

Listing 2. Code snippet illustrating the encoding of a *CompositeTimeseries* Dynamizer to model typical hourly electricity consumption for a building, with separate demand profiles time series components for weekdays and weekends.

```
"dyn_dynamizers": [
 {
   "dyn_startTime": "2024-01-01T00:00:00Z",
   "dyn_endTime": "2024-01-07T23:00:00Z",
   "dyn_attributeRef": "gen_electricityConsumption",
   "dyn_CompositeTimeseries": {
     "dyn_TimeSeriesComponent": [
       { "dyn_repetitions": 5,
         "dyn_GenericTimeSeries": {
           "dyn_firstTimestamp": "T00:00:00",
           "dyn_lastTimestamp": "T23:00:00",
           "dyn_observationProperty": "ElectricityConsumption
                ↪ ",
           "dyn_uom": "Wh",
           "dyn_valueType": "double",
           "dyn_TimeValuePair": [
             {"dyn_timestamp": "T00:00:00", "dyn_value": 375},
             ...
             {"dyn_timestamp": "T23:00:00","dyn_value": 529}
           ]}
       },
       { "dyn_repetitions": 1,
         "dyn_GenericTimeSeries": {...}
       },
       { "dyn_repetitions": 1,
         "dyn_GenericTimeSeries": {...}
       }
]}}]
```

The third dataset uses a *CompositeTimeseries* Dynamizer to represent typical hourly electricity consumption for a building in Munich's Harthof district, with separate time series components that distinguish between weekday and weekend demand profiles. The *CompositeTimeseries* Dynamizer is represented within *EXT_structural_metadata*, as illustrated in Listing 2.

The CityGML datasets were converted to the OGC 3D Tiles 1.1 format using a multistep process. First, the datasets were converted to the 3D Tiles 1.0 format using FME Form. The resulting 3D Tiles 1.0 datasets were then upgraded to 3D Tiles 1.1 with the open-source *3d-tiles-tools* suite of utilities[1] developed by Cesium, ensuring compatibility with structured metadata as defined in the 3D Metadata Specification. To create and preserve the semantics of CityGML Dynamizers, the proposed JSON-based encoding approach discussed in Section 3.1 was implemented. This was achieved using custom Python scripts to edit the *EXT_structural_metadata* schema and the properties of the glTF 2.0 assets containing the tile content.

4.2 4D Web Viewer Development

The 4D web viewer was developed as a Next.js application built on CesiumJS. It implements the Dynamizers discovery and interpretation logic discussed in Section 3.2. The application uses a centralised state management layer for shared state across multiple components, 3D features tracking, and temporal data storage. When loading 3D Tiles datasets, the application offers layer configuration parameters for labelling, styling, and legends. Declarative styling rules bind cartographic visual variables to time-dependent attributes, which enables rendering changes based on the temporal dimen-

[1] https://github.com/CesiumGS/3d-tiles-tools

sion. The user interface is designed to support intuitive 4D interaction. A layer control panel enables toggling layers, while a time-range picker lets users select the scene temporal range. This picker also overrides the start and end timestamps set by the *SensorConnection* Dynamizer class. In addition to displaying semantic attributes of selected features, interactive charts visualise detailed time series of the time-dependent attributes, allowing users to inspect trends. The CesiumJS timeline slider provides scrubbing functionality to animate temporal changes based on the 4D scene timestamp. The UI of the developed viewer application is illustrated in Figure 4, which includes key control elements for interactive rendering of the 4D scene, along with data panels for detailed context, and a building model enriched with three *SensorConnection* Dynamizers to update time-dependent indoor climate variables. The 4D viewer demo is publicly accessible on `https://4dcity.gis.lrg.tum.de`.

4.3 Runtime Performance Assessment

To quantify the runtime overhead introduced by temporal data integration, the 4D viewer was evaluated using a paired benchmark mode that compares two execution states of the same application: with temporal updates disabled and enabled. The benchmark is conducted through the deterministic route `/?benchmark=paired`, which first executes a baseline stage with `temporalMode=off`, captures a snapshot of live runtime metrics, then switches to `temporalMode=on`, captures another snapshot, and exports paired JSON and CSV results. Each benchmark run preserves the same camera state, dataset, and viewer configuration.

The assessment focuses on rendering and memory workload indicators that directly reflect interactive web performance. The primary rendering metric, frames per second (FPS), is calculated as `1000 / mean(frameTimes)` over a 60-frame rolling window; higher values indicate better performance. The mean frame time is derived from frame-to-frame deltas recorded via the CesiumJS `scene.postRender` event with lower frame-time values indicating faster rendering. Jank frames, an indicator of visual lag, are counted when frame time exceeds 33.3 ms, corresponding to rendering below 30 FPS, while JavaScript heap usage and total heap are measured in the browser's memory.

Table 1. Runtime comparison of the web viewer with temporal data disabled and enabled.

Metric	Better	Temporal Data Off	Temporal Data On	Δ	%
FPS	↑	22.10	18.55	-3.55	-16.1
Mean frame time (ms)	↓	45.25	53.85	8.60	19.0
Jank frames (>33ms)	↓	32.00	46.50	14.50	45.3
Heap memory used (MB)	↓	62.23	70.45	8.22	13.2
Heap memory total (MB)	↓	108.57	118.30	9.74	9.0

Table 1 shows the rendering performance and memory workload, summarised from the median results of 10 repeated benchmark runs of the first dataset with three Dynamizers of type *SensorConnection*. Overall, the results show a moderate runtime overhead when temporal data is enabled. The main impact is a reduction in FPS and an increase in mean frame time and jank frames, while memory usage increases only modestly. This suggests that the implemented temporal visualisation strategy remains computationally feasible, although further optimisation may be needed to reduce frame jank and improve rendering smoothness.

5. Discussion

Converting semantic 3D city models to web streaming formats often results in a degradation or loss of semantics and hierarchical structure. In many visualisation workflows, semantic information is incorporated in an ad hoc and unstructured manner, often tailored to specific use cases and viewers. This practice restricts the interoperability and reuse of datasets across UDT web platforms. This research demonstrates that structured encoding of CityGML semantics, including data structures defined by the Dynamizer module, can be transferred to OGC 3D Tiles using the 3D Metadata Specification and the glTF extension *EXT_structural_metadata*.

The developed discovery and interpretation framework establishes a standardised method for web viewers to automatically identify, parse, and process Dynamizers embedded within 3D Tiles datasets. The client-side interpretation logic relies on the structured semantic data and supports multiple Dynamizer types, enabling the visualisation of highly dynamic temporal changes in UDTs. Furthermore, the research proposes optimisation strategies that extend beyond spatial tiling to the temporal dimension. By maintaining active feature tracking, on-demand loading of external time series data, and dynamic styling of features within a scene view, the methodology seeks to achieve interactive 4D rendering performance for large-scale citywide datasets.

Overall, this research bridges the gap between semantic 3D city modelling and web streaming formats by leveraging widely adopted standards, OGC CityGML and OGC 3D Tiles. This standards-based approach enables interoperable, integrated visualisation of 3D city models alongside highly dynamic temporal changes. Although the primary focus was on updating time-dependent thematic properties of semantic 3D city models, the Dynamizer module also offers mechanisms for updating time-dependent spatial and appearance properties, which were not addressed in this study.

6. Conclusions and Future Work

This study demonstrates that the structured transfer of CityGML semantics, including Dynamizers, into web streaming specifications enables web viewers to systematically discover and interpret time-dependent properties within 4D visualisations of semantic 3D city models. In addition to supporting the discovery and interpretation of Dynamizers, preserving structured semantics could also unlock further benefits, such as determining optimal cartographic variables for visualising temporal data within 4D web visualisations.

Although CityGML Dynamizers formalise the association of time-dependent properties with their data

sources, they do not specify visualisation rules for web viewers. Consequently, users must define styling parameters in advance, which poses challenges for visualising city-wide datasets containing heterogeneous temporal data. Further research should investigate how inherent semantic and temporal data characteristics can be leveraged to automatically infer and apply appropriate cartographic styling. Additionally, enhanced tool support is required to automate the generation of Dynamizers, particularly as cities expand the deployment of IoT sensor networks to enable large-scale data integration with semantic 3D city models. Such advancements would accelerate the adoption of 4D web visualisations in UDTs for trend analysis and monitoring.

Acknowledgement

The authors acknowledge the City of Munich for its cooperation in the Connected Urban Twins (CUT) project, which was funded by the Federal Ministry for Housing, Urban Development and Building of Germany.

References

Ali, M. E., Cheema, M. A., Hashem, T., Ulhaq, A., Babar, M. A., 2024. Enabling Spatial Digital Twins: Technologies, Challenges, and Future Research Directions. *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 92(6), 761–778.

Bayerisches Landesamt für Digitalisierung, Breitband und Vermessung (LDBV), 2026. 3D-Gebäudemodelle (LoD2). Accessed: July 15, 2026.

Chaturvedi, K., 2021. Integration and Management of Time-Dependent Properties with Semantic 3D City Models. PhD thesis, Technical University of Munich.

Chiang, Y.-H., Huang, C.-Y., Fuse, M., 2020. The Integration of OGC SensorThings API and OGC CityGML via semantic web technology. S. Di Martino, Z. Fang, K.-J. Li (eds), *Web and Wireless Geographical Information Systems*, Lecture Notes in Computer Science, Springer International Publishing, Cham, 55–67.

Cirillo, F., Solmaz, G., Berz, E. L., Bauer, M., Cheng, B., Kovacs, E., 2019. A Standard-based Open Source IoT Platform: FIWARE. *IEEE Internet of Things Magazine*, 2(3), 12–18.

Cozzi, P., Lilley, S., 2023. 3D Tiles Specification Version 1.1. https://docs.ogc.org/cs/22-025r4/22-025r4.html.

Cozzi, P., Lilley, S., Getz, G., 2019. 3D Tiles Specification Version 1.0. https://docs.ogc.org/cs/18-053r2/18-053r2.html.

Crampen, D., Yadav, Y., Ishar, S., Zlatanova, S., Tian, W., 2024. Development of 3D UDTs for Traffic Monitoring in Unreal 5 Game Engine. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-4-2024, 131–138.

Faliagka, E., Christopoulou, E., Ringas, D., Politi, T., Kostis, N., Leonardos, D., Tranoris, C., Antonopoulos, C. P., Denazis, S., Voros, N., 2024. Trends in Digital Twin Framework Architectures for Smart Cities: A Case Study in Smart Mobility. *Sensors*, 24(5), 1665.

Ferré-Bigorra, J., Casals, M., Gangolells, M., 2022. The Adoption of Urban Digital Twins. *Cities*, 131, 103905.

Jeddoub, I., Nys, G.-A., Hajji, R., Billen, R., 2024. Data Integration across Urban Digital Twin Lifecycle: A Comprehensive Review of Current Initiatives. *Annals of GIS*, 31(3), 367–386.

Khronos Group, 2021. glTF™ 2.0 Specification. https://registry.khronos.org/glTF/.

Kolbe, T. H., Kutzner, T., Smyth, C. S., Nagel, C., Roensdorf, C., Heazel, C., 2021. OGC City Geography Markup Language (CityGML) Part 1: Conceptual Model Standard.

Kutzner, T., Chaturvedi, K., Kolbe, T. H., 2020. CityGML 3.0: New Functions Open up New Applications. *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88(1), 43–61.

Lei, B., Janssen, P., Stoter, J., Biljecki, F., 2023. Challenges of Urban Digital Twins: A Systematic Review and a Delphi Expert Survey. *Automation in Construction*, 147, 104716.

Liang, S., Huang, C.-Y., Khalafbeigi, T., 2016. OGC SensorThings API Part 1: Sensing. https://docs.ogc.org/is/15-078r6/15-078r6.html.

Rantanen, T., Julin, A., Virtanen, J.-P., Hyyppä, H., Vaaja, M. T., 2023. Open Geospatial Data Integration in Game Engine for Urban Digital Twin Applications. *ISPRS International Journal of Geo-Information*, 12(8), 310.

Reed, C., Belayneh, T., 2023. OGC Indexed 3d Scene Layer (I3S) and Scene Layer Package (*.Slpk) Format Community Standard. https://docs.ogc.org/cs/17-014r9/17-014r9.html.

Robinet, F., Arnaud, R., Parisi, T., Cozzi, P., 2018. glTF: Designing an open-standard runtime asset format. *GPU Pro 360 Guide to 3D Engine Design*, A K Peters/CRC Press, 393–410.

Santhanavanich, T., Coors, V., 2021. CityThings: An Integration of the Dynamic Sensor Data to the 3D City Model. *Environment and Planning B: Urban Analytics and City Science*, 48(3), 417–432.

Weil, C., Bibri, S., Longchamp, R., Golay, F., Alahi, A., 2023. Urban Digital Twin Challenges: A Systematic Review and Perspectives for Sustainable Smart Cities. *Sustainable Cities and Society*, 99, 104862.

Würstle, P., Padsala, R., Santhanavanich, T., Coors, V., 2022. Viability Testing of Game Engine Usage for Visualization of 3D Geospatial Data with OGC Standards. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W2-2022, 281–288.

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., Adolphi, T., Kolbe, T. H., 2018. 3DCityDB - a 3D Geodatabase Solution for the Management, Analysis, and Visualization of Semantic 3D City Models Based on CityGML. *Open Geospatial Data, Software and Standards*, 3(1), 5.