GitHub URL: https://github.com/msbsyed/eo-archetype-library/

A reference model for sensor-based environmental observation data, built on OGC's SWE O&M standards. Supports archetype-based two-level modelling, LinkEHR workflows, and semantic enrichment for improved data quality, interoperability, and reuse in Earth Informatics.

📘 About This Reference Model This reference model provides a stable, semantic-friendly foundation for representing environmental monitoring data — particularly from sensor-based observations — in a reusable, interoperable, and extensible way. It follows the principles of two-level modelling, where a consistent reference model defines core structures, and domain-specific archetypes represent detailed, context-rich semantics tailored for reuse.

🔹 Foundation and Alignment This model builds upon the base reference model proposed by Dr. Paul Stacey (Towards People Oriented Technology (tPOT) Research Centre, TU Dublin), originally designed to support the translation of two-level models from e-Health into Earth Informatics. The original model is available at: https://github.com/pstacey/geo-archetype-library

Expanding on that, this version adapts and extends the core reference model using standards from the Open Geospatial Consortium (OGC) — especially the Observations & Measurements (O&M) model and the Sensor Web Enablement Common Data Model (SWE-CDM). It introduces flattened and harmonised data types suitable for:

1. Sensor-based air quality monitoring
2. Environmental and geospatial observation data
3. Archetype-driven two-level information modelling

The schema is designed to be standalone and LinkEHR-compatible, enabling archetyping and knowledge formalisation without relying on external schema imports. This makes it highly suitable for modelling observational Earth Science data while supporting semantic interoperability, data quality assurance, and reuse across distributed environmental data systems.

🧩 Use Case [Not comprehensive, can be extended to multiple subdomains] This reference model underpins the development of domain-specific archetypes, for instance:

1. Air quality parameters (e.g., PM2.5, NO_2, O_3, SO_2, CO)
2. O&M elements like observed properties, Features of interest, etc.

🔧 Modular Archetype Support This reference model is designed to support modular archetype development — enabling the separation of key metadata components such as:

1. Sensor calibration information
2. Sensor specifications
3. Deployment context
4. Sensor operational characteristics
5. Aggregated observational values (e.g., hourly means, Air Quality Index calculations)

These modular archetypes can be independently defined and linked together within higher-level archetypes (e.g., an air quality observation) to create a cohesive, semantically rich metadata model. This allows for the creation of high-quality, standards-aligned information structures that accompany observation data, improving:

1. Documentation quality
2. Semantic clarity
3. Reusability across systems and domains
4. Support for automated reasoning and validation

This compositional structure encourages reuse, refinement, and community participation, aligning with two-level modelling best practices for Earth Informatics.

🔧 Technical Highlights

Modular and Maintainable Structure: Built using a set of well-structured, modular XSD schema files, with consistent and carefully managed namespace definitions. This facilitates clean reuse and extensibility across projects and modelling layers.

LinkEHR-Compatible Design: Structured to fully support archetype-slot references in LinkEHR workflows, allowing for flexible, compositional archetype development using familiar tooling. This is essential for enabling collaborative and distributed model refinement.

Semantic Integration Ready: Designed with semantic interoperability in mind, the reference model facilitates the integration of controlled vocabularies, code lists, and external ontologies. This makes it suitable for high-quality metadata modelling and semantic enrichment of observation data.

👤 Author This reference model is developed by M.S.B. Syed, PhD candidate at Technological University Dublin and Taighde Éireann – Research Ireland Centre for Research Training in Advanced Networks for Sustainable Societies, with the financial support from Taighde Éireann – Research Ireland under Grant No. 18/CRT/6222. The work is part of an ongoing research project titled: "Using Archetype-Based Two-Level

Information Modelling to Enhance Data Quality in SDG 11 Sustainable Cities and Communities Implementation Monitoring".

------- Schema Below -------

<?xml version="1.0" encoding="UTF-8"?>

<!--

Title: SWE-CDM_Archetype_RM

Author: M.S.B. Syed

Affiliation: Towards People Oriented Technology (tPOT) Research Centre, Technological University Dublin

Funding Statement: This work is funded by Taighde Éireann – Research Ireland under Grant number 18/CRT/6222

NOTE: This schema is a **standalone implementation** of the SWE Common Data Model

designed specifically for compatibility with **LinkEHR**, which does **not support external schema imports**.

- All references/imports to external SWE namespaces (e.g., xmlns:swe, xs:import schemaLocation="swe.xsd"etc.)

have been **omitted** to allow successful parsing in LinkEHR.

- As a result, all types that originally used the `swe:` prefix have been redefined within this schema

under a **custom namespace** or with no prefix. This ensures all elements and types are available internally.

- This reference model aligns conceptually with both the **OGC Observations & Measurements (O&M)** model

and the **OGC SWE Common Data Model**, while adapting the structure to fit archetyping use cases in LinkEHR.

- Certain advanced or rarely-used types such as `BinaryEncoding` or `XMLEncoding` are intentionally **excluded**

to keep the schema **streamlined and focused** on sensor-based environmental monitoring data.

- The goal is to provide a comprehensive but lightweight set of SWE-compatible data types for building

interoperable and [reusable] archetypes for sensor-based environmental monitoring observations.

-->

<xs:schema

xmlns:xs="http://www.w3.org/2001/XMLSchema"

targetNamespace="http://tpot.tudublin.ie"

xmlns="http://tpot.tudublin.ie"

elementFormDefault="qualified"

attributeFormDefault="unqualified"

version="2.1.0">

<!-- Rather than defining two abstract classes [e.g., one for Simple Data Components and other for Complex Data Components], this schema simplifies the structure -->

<!-- A single base abstract class is defined to which both types of data components can correspond -->

<xs:complexType name="AbstractDataComponent" abstract="true"> </xs:complexType>

<!-- IDENTITY_ABSTRACT: Abstract base type providing common identification elements for data entities.

Includes human-readable name, unique identifier (OBJECT_ID), and optional archetype metadata

(ID, terminology, version) to support traceability and alignment with archetyping frameworks.

Designed to enable consistent identity structure across compositional elements in the reference [1st level] model -->

<xs:complexType name="IDENTITY_ABSTRACT" abstract="true">

<xs:sequence>

<xs:element name="name" type="xs:string"/>

<xs:element name="identifier" type="OBJECT_ID"/>

<xs:element name="archetype_id" type="OBJECT_ID" minOccurs="0"/>

<xs:element name="archetype_terminology" type="TERMINOLOGY_ID" minOccurs="0"/>

<xs:element name="archetype_version" type="xs:string" minOccurs="0"/>

</xs:sequence>

</xs:complexType>

<!-- ABSTRACT_OBS: Abstract specialisation of IDENTITY_ABSTRACT for observations.

Used as a base type for defining concrete observation classes,

ensuring each observation retains standardised identity metadata for interoperability -->

<xs:complexType name="ABSTRACT_OBS" abstract="true">

<xs:complexContent>

```
            <xs:extension base="IDENTITY_ABSTRACT"/>
        </xs:complexContent>
    </xs:complexType>

    <!-- GeoData_COMPOSITION: High-level container representing a complete geospatial dataset or package of sensor observations.
    Extends IDENTITY_ABSTRACT to support unique identification and links to one or more observation sets.
    'details' allows embedding rich descriptive metadata using a structured composite format.
    Designed for use in standalone reference [1st level] model compatible with LinkEHR, aligned with OGC OM & SWE-CDM specs -->
    <xs:element name="geo_identity" type="Geo_Data_Document"/>
    <xs:complexType name="Geo_Data_Document">
        <xs:complexContent>
            <xs:extension base="IDENTITY_ABSTRACT">
                <xs:sequence>
                    <xs:element name="geoDataComposition" type="GeoData_COMPOSITION" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="details" type="DETAILS_COMPOSITE"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- OBSERVATION_SET Archetype Class -->
    <xs:complexType name="OBSERVATION_SET">
        <xs:complexContent>
            <xs:extension base="ABSTRACT_OBS">
```

```
                    <xs:sequence>
                        <xs:element name="details" type="DETAILS_COMPOSITE"/>
                        <xs:element name="observation" type="OBSERVATION" minOccurs="0" maxOccurs="unbounded"/>
                        <xs:element name="observedProperty" type="ObservedProperty"/>
                        <xs:element name="featureOfInterest" type="FeatureOfInterest"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>

        <!-- OBSERVATION Class -->
        <xs:complexType name="OBSERVATION">
            <xs:complexContent>
                <xs:extension base="ABSTRACT_OBS">
                    <xs:sequence>
                        <xs:element name="details" type="DETAILS_COMPOSITE" minOccurs="0"/>
                        <xs:element name="featureOfInterest" type="FeatureOfInterest"/>
                        <xs:element name="observedProperty" type="ObservedProperty"/>
                        <xs:element name="result" type="Result"/>
                        <xs:element name="resultTime" type="Time"/>
                        <xs:element name="resultTimeRange" type="TimeRange"/>
                        <xs:element name="procedure" type="OM_PROCESS"/>
```

```
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- Single and Multiple Result structures -->
    <xs:complexType name="Result">
        <xs:complexContent>
            <xs:extension base="AbstractDataComponent">
                <xs:sequence>
                    <xs:element name="value" type="AbstractDataComponent"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ResultsArray">
        <xs:complexContent>
            <xs:extension base="AbstractDataComponent">
                <xs:sequence>
                    <xs:element name="element" type="AbstractDataComponent" minOccurs="1" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- GeoData_COMPOSITION -->
    <xs:complexType name="GeoData_COMPOSITION">
```

```
        <xs:complexContent>
            <xs:extension base="IDENTITY_ABSTRACT">
                <xs:sequence>
                    <xs:element name="observationSet" type="OBSERVATION_SET" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="details" type="DETAILS_COMPOSITE"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- NAMED_VALUE: Base type for any element that requires a name and an optional identifier. Enables explicit naming of values, useful for metadata and nested composite structures -->
    <xs:complexType name="NAMED_VALUE" abstract="true">
        <xs:complexContent>
            <xs:extension base="AbstractDataComponent">
                <xs:sequence>
                    <xs:element name="identifier" type="OBJECT_ID" minOccurs="0"/>
                </xs:sequence>
                <xs:attribute name="name" type="xs:string" use="required"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

<!-- DETAILS_COMPOSITE: Represents a group of named metadata elements, each being a NAMED_VALUE. Useful for structured descriptions of sensors, observations, or entities like ObservedProperty -->

<xs:complexType name="DETAILS_COMPOSITE">

<xs:complexContent>

<xs:extension base="NAMED_VALUE">

<xs:sequence>

<xs:element name="element" type="NAMED_VALUE" maxOccurs="unbounded"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<!-- DETAILS_ELEMENT: Extends NAMED_VALUE to allow inclusion of actual data values. Used to attach one or more **AbstractDataComponent** values (e.g., quantity, time) to a named entry -->

<xs:complexType name="DETAILS_ELEMENT">

<xs:complexContent>

<xs:extension base="NAMED_VALUE">

<xs:sequence>

<xs:element name="data_value" type="AbstractDataComponent" maxOccurs="unbounded"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<!-- ObservedProperty, FeatureOfInterest, OM_Process -->

<!-- ObservedProperty: Represents the phenomenon or characteristic being observed (e.g., temperature, air quality). Supports semantic tagging via terminology references -->

<xs:complexType name="ObservedProperty">

<xs:sequence>

<xs:element name="identifier" type="OBJECT_ID" minOccurs="0"/>

<xs:element name="details" type="DETAILS_COMPOSITE"/>

<xs:element name="terminology" type="TERMINOLOGY_ID" minOccurs="0"/>

</xs:sequence>

</xs:complexType>

<!-- FeatureOfInterest: Describes the real-world object or geographic feature that the observation applies to (e.g., location, air parcel, river segment) -->

<xs:complexType name="FeatureOfInterest">

<xs:sequence>

<xs:element name="identifier" type="OBJECT_ID" minOccurs="0"/>

<xs:element name="details" type="DETAILS_COMPOSITE"/>

<xs:element name="terminology" type="TERMINOLOGY_ID" minOccurs="0"/>

</xs:sequence>

</xs:complexType>

<!-- OM_PROCESS: Represents the observation procedure or sensor system that generates the result (e.g., air quality station, satellite instrument) -->

<xs:complexType name="OM_PROCESS">

<xs:sequence>

<xs:element name="identifier" type="OBJECT_ID" minOccurs="0"/>

<xs:element name="details" type="DETAILS_COMPOSITE"/>

```
                <xs:element name="terminology" type="TERMINOLOGY_ID" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>

        <!-- Object_ID definitions to enable support for diverse identifier systems beyond URIs,
        including UUIDs, GUIDs, and DOIs. These are useful in systems requiring flexible,
        globally unique identifiers (e.g., for observations, sensors, or features of interest) -->
        <xs:complexType name="OBJECT_ID" abstract="true">
            <xs:sequence>
                <xs:element name="value" type="xs:token"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="UID_BASED_ID" abstract="true">
            <xs:complexContent>
                <xs:extension base="OBJECT_ID"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="UUID">
            <xs:complexContent>
                <xs:extension base="UID_BASED_ID"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="GUID">
            <xs:complexContent>
                <xs:extension base="UID_BASED_ID"/>
            </xs:complexContent>
```

```
    </xs:complexType>
    <xs:complexType name="DOI">
        <xs:complexContent>
            <xs:extension base="OBJECT_ID"/>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="URI_ID">
        <xs:complexContent>
            <xs:extension base="OBJECT_ID"/>
        </xs:complexContent>
    </xs:complexType>

    <!-- Terminology references to enable semantic alignment and ontology-based
    interoperability (e.g., for code systems [internal/external/online/ffline] or value sets in vocabularies) -->
    <xs:complexType name="TERMINOLOGY_ID">
        <xs:complexContent>
            <xs:extension base="OBJECT_ID"/>
        </xs:complexContent>
    </xs:complexType>


    <!-- SWE Common Data Model Data Types -->
    <!-- Accessible at: https://schemas.opengis.net/sweCommon/2.0/ -->

    <!-- Quantity -->
    <!-- Represents a continuous numeric value with an optional unit of measure (e.g., temperature, pressure) -->
    <xs:complexType name="Quantity">
```

```
            <xs:complexContent>
                <xs:extension base="AbstractDataComponent">
                    <xs:sequence>
                        <xs:element name="uom" type="UnitReference" minOccurs="0"/>
                        <xs:element name="value" type="xs:double" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>

        <!-- Time -->
        <!-- Timestamp (ISO 8601 format) with optional unit; represents when an observation was made -->
        <xs:complexType name="Time">
            <xs:complexContent>
                <xs:extension base="AbstractDataComponent">
                    <xs:sequence>
                        <xs:element name="uom" type="xs:string" minOccurs="0"/>
                        <xs:element name="value" type="xs:dateTime" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>

        <!-- TimePair -->
```

```
<!-- TimePair: Represents a start and end timestamp. Used in TimeRange to specify a bounded duration. Useful for interval-based sensor data (e.g., from-to time) -->
<xs:complexType name="TimePair">
    <xs:sequence>
        <xs:element name="start" type="xs:dateTime"/>
        <xs:element name="end" type="xs:dateTime"/>
    </xs:sequence>
</xs:complexType>

<!-- TimeRange -->
<!-- Start and end times to represent a temporal interval, useful for durations or events -->
<xs:complexType name="TimeRange">
    <xs:complexContent>
        <xs:extension base="AbstractDataComponent">
            <xs:sequence>
                <xs:element name="uom" type="xs:string" minOccurs="0"/>
                <xs:element name="value" type="TimePair" minOccurs="0"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<!-- Text -->
<!-- Free-form string for comments, descriptions, or unstructured text data -->
<xs:complexType name="Text">
    <xs:complexContent>
```

```
                    <xs:extension base="AbstractDataComponent">
                        <xs:sequence>
                            <xs:element name="value" type="xs:string" minOccurs="0"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>

            <!-- Count -->
            <!-- Integer-based discrete values used for counting (e.g., number of particles, events) -->
            <xs:complexType name="Count">
                <xs:complexContent>
                    <xs:extension base="AbstractDataComponent">
                        <xs:sequence>
                            <xs:element name="value" type="xs:integer" minOccurs="0"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>

            <!-- Boolean -->
            <!-- Logical true/false value for sensors or flags -->
            <xs:complexType name="Boolean">
                <xs:complexContent>
                    <xs:extension base="AbstractDataComponent">
                        <xs:sequence>
```

```
                    <xs:element name="value" type="xs:boolean" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- Category -->
    <!-- Categorical value from a controlled vocabulary (e.g., sensor status like "normal" or "fault") -->
    <xs:complexType name="Category">
        <xs:complexContent>
            <xs:extension base="AbstractDataComponent">
                <xs:sequence>
                    <xs:element name="codeSpace" type="xs:string" minOccurs="0"/>
                    <xs:element name="value" type="xs:string" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- QuantityRange -->
    <!-- Represents a range between two numeric values (e.g., "5.0 10.0") to express uncertainty or tolerance -->
<xs:complexType name="QuantityRange">
  <xs:complexContent>
    <xs:extension base="AbstractDataComponent">
```

```
    <xs:sequence>
      <xs:element name="uom" type="UnitReference" minOccurs="0"/>
      <xs:element name="value" type="xs:string" minOccurs="0"/> <!-- e.g. "5.0 10.0" -->
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

        <!-- CountRange -->
        <!-- Integer range to express uncertainty or variability in discrete counts (e.g., "1 5")
-->
        <xs:complexType name="CountRange">
                <xs:complexContent>
                        <xs:extension base="AbstractDataComponent">
                                <xs:sequence>
                                        <xs:element        name="value"        type="xs:string"
minOccurs="0"/>
                                </xs:sequence>
                        </xs:extension>
                </xs:complexContent>
        </xs:complexType>

        <!-- CategoryRange -->
        <!-- Represents an ordinal or categorical range (e.g., "low high") in a code space -
->
        <xs:complexType name="CategoryRange">
                <xs:complexContent>
                        <xs:extension base="AbstractDataComponent">
                                <xs:sequence>
```

```
                                    <xs:element name="codeSpace" type="xs:string" minOccurs="0"/>
                                    <xs:element name="value" type="xs:string" minOccurs="0"/>
                                </xs:sequence>
                            </xs:extension>
                        </xs:complexContent>
                    </xs:complexType>

                    <!-- Vector -->
                    <!-- Multi-dimensional numeric values with reference frames (e.g., 3D coordinates, wind direction) -->
                    <xs:complexType name="Vector">
                        <xs:complexContent>
                            <xs:extension base="AbstractDataComponent">
                                <xs:sequence>
                                    <xs:element name="referenceFrame" type="xs:string" minOccurs="0"/>
                                    <xs:element name="localFrame" type="xs:string" minOccurs="0"/>
                                    <xs:element name="coordinates" minOccurs="1" maxOccurs="unbounded">
                                        <xs:complexType>
                                            <xs:sequence>
                                                <xs:element name="name" type="xs:string"/>
                                                <xs:element name="value" type="xs:double"/>
                                            </xs:sequence>
                                        </xs:complexType>
```

```
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<!-- DataRecord -->
<!-- Composite structure grouping multiple named data fields; used for structured sensor output -->
<xs:complexType name="DataRecord">
    <xs:complexContent>
        <xs:extension base="AbstractDataComponent">
            <xs:sequence>
                <xs:element name="field" minOccurs="1" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="name" type="xs:string"/>
                            <xs:element name="value" type="ChoiceComponent"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

```
        <!-- DataArray -->
        <!-- Repeating list of identical components (e.g., time series of temperature values) -->
        <xs:complexType name="DataArray">
            <xs:complexContent>
                <xs:extension base="AbstractDataComponent">
                    <xs:sequence>
                        <xs:element name="elementCount" type="xs:integer"/>
                        <xs:element name="elementType" type="ChoiceComponent"/>
                        <xs:element name="values" type="xs:string" minOccurs="0"/> <!-- Could be comma-separated values -->
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>

        <!-- NilValues -->
        <!-- Describes missing or invalid values along with a reason (e.g., "sensor offline") -->
        <xs:complexType name="NilValues">
 <xs:sequence>
  <xs:element name="nilValue" minOccurs="1" maxOccurs="unbounded">
   <xs:complexType>
    <xs:simpleContent>
     <xs:extension base="xs:string">
      <xs:attribute name="reason" type="xs:string" use="required"/>
     </xs:extension>
```

```
    </xs:simpleContent>
  </xs:complexType>
 </xs:element>
</xs:sequence>
</xs:complexType>

    <!-- Matrix -->
    <!-- Two-dimensional structure representing tabular or spatial sensor data with defined rows and columns -->
    <xs:complexType name="Matrix">
        <xs:complexContent>
            <xs:extension base="AbstractDataComponent">
                <xs:sequence>
                    <xs:element name="referenceFrame" type="xs:string" minOccurs="0"/>
                    <xs:element name="localFrame" type="xs:string" minOccurs="0"/>
                    <xs:element name="elementCount" type="xs:integer"/>
                    <xs:element name="elementType" type="ChoiceComponent"/>
                    <xs:element name="values" type="xs:string" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- DataStream -->
```

```
<!-- Streamed sensor values with encoding and count — suited for continuous data transmission -->
<xs:complexType name="DataStream">
    <xs:complexContent>
        <xs:extension base="AbstractDataComponent">
            <xs:sequence>
                <xs:element name="elementCount" type="xs:integer" minOccurs="0"/>
                <xs:element name="elementType" type="ChoiceComponent"/>
                <xs:element name="values" type="xs:string" minOccurs="0"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<!-- Quality -->
<!-- Metadata about the reliability, accuracy, or confidence of a data value -->
<xs:complexType name="Quality">
    <xs:choice>
        <xs:element name="quantity" type="Quantity"/>
        <xs:element name="quantity_range" type="QuantityRange"/>
        <xs:element name="category" type="Category"/>
        <xs:element name="text" type="Text"/>
    </xs:choice>
</xs:complexType>

<!-- Unit of Reference -->
```

```
<!-- Reference to a unit of measure, used to clarify the semantic meaning of numeric values -->
<xs:complexType name="UnitReference">
    <xs:attribute name="code" type="xs:string" use="optional"/>
</xs:complexType>

<!-- ChoiceComponent -->
<!-- Flexible placeholder to hold any of the defined SWE types — used in structures like records and arrays -->
<xs:complexType name="ChoiceComponent">
    <xs:choice>
        <xs:element name="quantity" type="Quantity"/>
        <xs:element name="quantity_range" type="QuantityRange"/>
        <xs:element name="count" type="Count"/>
        <xs:element name="count_range" type="CountRange"/>
        <xs:element name="time" type="Time"/>
        <xs:element name="time_range" type="TimeRange"/>
        <xs:element name="text" type="Text"/>
        <xs:element name="boolean" type="Boolean"/>
        <xs:element name="category" type="Category"/>
        <xs:element name="category_range" type="CategoryRange"/>
        <xs:element name="vector" type="Vector"/>
        <xs:element name="matrix" type="Matrix"/>
        <xs:element name="record" type="DataRecord"/>
        <xs:element name="array" type="DataArray"/>
        <xs:element name="stream" type="DataStream"/>
        <xs:element name="unit_reference" type="UnitReference"/>
        <xs:element name="quality" type="Quality"/>
        <xs:element name="nilValues" type="NilValues"/>
```

```
            </xs:choice>
        </xs:complexType>

</xs:schema>
```