

Article

Extending INSPIRE to the Internet of Things through SensorThings API

Alexander Kotsev ^{1,*†,‡,§} , Katharina Schleidt ^{2,‡}, Steve Liang ^{3,‡}, Hylke van der Schaaf ^{4,‡},
Tania Khalafbeigi ^{3,‡}, Sylvain Grellet ^{5,‡}, Michael Lutz ^{1,‡}, Simon Jirka ^{6,‡} and Mickaël Beaufils ^{5,‡}

¹ European Commission, Joint Research Centre, 21027 Ispra, Italy; michael.lutz@ec.europa.eu

² DataCove e.U., 1150 Vienna, Austria; kathi@datacove.eu

³ Department of Geomatics Engineering, University of Calgary, Canada; steve.liang@sensorup.com (S.L.);
tania.khalafbeigi@sensorup.com (T.K.)

⁴ Fraunhofer IOSB, 76131 Karlsruhe, Germany; hylke.vanderschaaf@iosb.fraunhofer.de

⁵ French Geological Survey (BRGM), 45060 Orleans, France; s.grellet@brgm.fr (S.G.);
m.beaufils@brgm.fr (M.B.)

⁶ 52°North GmbH, 48155 Münster, Germany; jirka@52north.org

* Correspondence: alexander.kotsev@ec.europa.eu; Tel.: +39-0332-78-9069

† Current address: Via E. Fermi, 2749, 21027 Ispra (VA), Italy

‡ These authors contributed equally to this work.

§ The views expressed are purely those of the authors and may not in any circumstances be regarded as stating an official position of the European Commission.

Received: 30 April 2018; Accepted: 12 June 2018; Published: 16 June 2018



Abstract: Spatial Data Infrastructures (SDI) established during the past two decades “unlocked” heterogeneous geospatial datasets. The European Union INSPIRE Directive laid down the foundation of a pan-European SDI where thousands of public sector data providers make their data, including sensor observations, available for cross-border and cross-domain reuse. At the same time, SDIs should inevitably adopt new technology and standards to remain fit for purpose and address in the best possible way the needs of different stakeholders (government, businesses and citizens). Some of the recurring technical requirements raised by SDI stakeholders include: (i) the need for adoption of RESTful architectures; together with (ii) alternative (to GML) data encodings, such as JavaScript Object Notation (JSON) and binary exchange formats; and (iii) adoption of asynchronous publish–subscribe-based messaging protocols. The newly established OGC standard SensorThings API is particularly interesting to investigate for INSPIRE, as it addresses together all three topics. In this manuscript, we provide our synthesised perspective on the necessary steps for the OGC SensorThings API standard to be considered as a solution that meets the legal obligations stemming out of the INSPIRE Directive. We share our perspective on what should be done concerning: (i) data encoding; and (ii) the use of SensorThings API as a download service.

Keywords: SensorThings API; INSPIRE; download services; spatio-temporal data interoperability; Internet of Things

1. Introduction

Spatial Data Infrastructures (SDI) built during the past two decades brought numerous novelties and triggered a process that considerably improved the availability and accessibility of spatial data. A similar approach might therefore be adopted in fields where the sharing and reuse of data are needed. The Internet of Things (IoT) is such a field that brings both opportunities and challenges for the evolution of existing data infrastructures. The issues associated with the utilisation and reuse of IoT data are complex, including: (i) the heterogeneity of devices; (ii) different data encoding formats;

(iii) constraints determined by limited connectivity and low computational capabilities of sensor platforms; and (iv) lack of interoperability between different, often proprietary, platforms.

Within this complex setting, we investigated how spatial data infrastructures can be extended to the IoT. We chose the European SDI built as result of the implementation of the European Union INSPIRE Directive [1] as our test case for several reasons. Firstly, Europe's diversity, on the one side its greatest strength, is at the same time one of its greatest weaknesses. While many data are available from institutions across Europe, it is exceedingly difficult to unlock their potential due to a multitude of technical and cultural approaches to data assay and management. To leverage the potential of these data in their entirety, so essential to solving the ever more complex problems that need to be faced in today's rapidly evolving world, a painstaking process of data harmonization and alignment is required. Secondly, INSPIRE conceptualises requirements on an abstract level without requiring a particular set of technologies to be used. This provides opportunities for multiple alternative solutions to be investigated, including the possibility to extend to different domains.

In 2007, the foundation of the pan-European spatial data infrastructure (SDI) was established, allowing the combined use of environmental data across borders and across domains, and ameliorating the downfalls of diversity as described above. The legislative basis was established, and has ever since been catalysing digital transformation within a vast number of public sector authorities. The adoption of the Directive brought multiple novelties to public sector authorities, including: (i) the requirement to expose data and metadata in a service oriented architecture; (ii) strong reliance on international standards; and (iii) a collaborative consensus-based approach during the specification drafting. Those developments, combined, provide access to thousands of spatial datasets. Currently, more than 100,000 datasets are made available in INSPIRE by roughly 7000 institutions, whereby many datasets are still provisionally provided in local non-aligned formats, awaiting harmonization by the 2020 deadline. As the geospatial data available within SDIs are an important contributor to the EU data economy, which is estimated at EUR 300 billion in 2016 (1.99% of the EU GDP), unlocking the full potential of INSPIRE will provide an additional impulse to this already burgeoning industry. The estimated growth rate is also remarkable, as the contribution of data to the European economy is expected to increase to EUR 739 billion by 2020, representing thus 4.00% [2].

One of the challenges faced in the specification of the requirements to INSPIRE was the integration of measurement data with the spatial data that usually comprises an SDI. This was due to the scope of the INSPIRE Directive; in addition to purely spatial data, some data themes such as Environmental Monitoring Facilities, which *"includes observation and measurement of emissions, of the state of environmental media and of other ecosystem parameters"* [1]. After the creation of initial prototypes utilising standardised spatial data services such as the OGC Web Feature Service (WFS), the conclusion was reached that these technologies are not suited to the provision of measurement data. The OGC Sensor Web Enablement Suite (SWE) was then analysed for suitability [3,4]. This process ended in the adoption of the OGC Sensor Observation Service (SOS) as an INSPIRE download service for measurement data [5]. While being proven as a powerful tool for unlocking the wealth of measurement information available, it has also taken the GIS departments of the environmental agencies managing these data far from their comfort zones, as greater power often comes at the cost of greater complexity.

Today, more than ten years after the adoption of the Directive, the technological scenery is rather different due to the penetration and uptake of technology in literally all human activities. Nowadays, the amount of information generated every two days is equal to all data created from the dawn of civilisation until 2003 [6]. INSPIRE remains a driver for change, with a profound impact on all actors involved. The Directive is seen on a European level as a best practice that should be extended beyond the environmental domain [2]. The legislation and guidance documents in INSPIRE are designed to be: (i) neutral from a particular software solution; and (ii) open to emerging technological trends. Various ongoing activities and emerging technologies are contributing to this technological evolution of INSPIRE. The objectives of these activities are different, but they share the commonality of proposing solutions that add value to data providers' infrastructures, while preserving semantics and,

wherever possible, ensuring backwards compatibility with already established solutions. Multiple European actors from governmental institutions, research and industry are involved on different levels (local, regional, and international) in activities dedicated to the technological evolution of SDIs and INSPIRE. Experts collaborate on improvements, such as linked data, RESTful architectures and industry-standard data encodings [7] that ensure that the infrastructure remains fit for purpose despite the rapidly changing technological scenery.

Within this context, some of the recurring technical requirements raised by stakeholders during the recent “INSPIRE - What if...?” workshop [7] include: (i) the need for adoption of RESTful architectures; together with (ii) alternative (to GML) data encodings, such as JavaScript Object Notation (JSON) and binary exchange formats; and (iii) adoption of asynchronous publish–subscribe-based messaging protocols. The research objective being addressed by this paper pertains to the challenge of the seamless integration of these new technologies into the emerging INSPIRE landscape, making best use of the potential provided while minimizing the disruptive effects, thus simplifying the data provision process, while making the existing data more available to all. Given the bounding conditions and overarching research objective outlined above, the newly established OGC standard SensorThings API [8] is of particular interest for investigation pertaining to INSPIRE, as it addresses all three topics together. Developed to cover a multitude of Internet of Things (IoT) use cases, the standard allows for lightweight provision of measurement data, while also being designed to be “developer friendly”. While the utilization of the SensorThings API within the context of INSPIRE greatly simplifies the process of providing and utilizing measurement data, its adoption can also be seen as a new paradigm in the evolution of the INSPIRE infrastructure, contributing to its simplification as well as allowing for its extension to new domains within the IoT realm.

In this manuscript, we provide our synthesised perspective on the necessary steps for the OGC SensorThings API standard to be considered as a solution that meets the legal obligations stemming from the INSPIRE Directive, thus simplifying the process for extending existing spatial data infrastructures to the IoT. We share our perspective on what should be done with regards to: (i) data encoding; and (ii) the use of SensorThings API as a download service. Structurally, the paper is divided into four sections. Following this brief Introduction (Section 1), in Section 2, we describe the background with an emphasis on the legislative and technological context in Europe, as well as the SensorThings API standard along with the existing implementations and an overview of use cases. Section 3 describes a mapping between the standard and the legal requirements of INSPIRE. Both data encoding and web service operations are covered. Consequently, in Section 4, we discuss our main findings, pending challenges, and give an outlook to our future research directions.

From our perspective the benefits of the proposed approach are manifold as the technological lessons learned from the establishment of SDIs can be reused in other contexts that face similar challenges. Those relate to: (i) multiple actors involved on different levels; (ii) heterogeneous data management practices; (iii) semantic issues; and (iv) differences in the discoverability and reusability of data. The IoT with its exponential growth and heterogeneity faces all of these issues together, making it an appropriate candidate for the exploration of methods for the evolution of an SDI beyond the purely spatial aspects. It makes no sense to address such issues in isolation from open data and SDI developments; considered together, in an integrated and aligned manner, they are far more than the sum of their parts, providing the opportunity for a truly diverse and embracing spatial data landscape for Europe. In doing so, we not only ensure that the pan-European data infrastructure remains fit-for-purpose by encompassing huge amounts of IoT data (evolving it from a spatial to a spatio-temporal data infrastructure), but also use the opportunity to address some of the recurring technical requirements posed by the evolution of shared infrastructures pertaining to diverse types of data across the world. We therefore consider our results applicable beyond Europe as well.

2. Background

2.1. INSPIRE in a Nutshell

INSPIRE came into force on 15 May 2007, with full implementation in every EU Member State required by 2021. It combines a legal and a technical framework for all EU Member States, to make relevant spatial data accessible for further reuse [9]. In particular, this means that data shall be discoverable and interoperable through the implementation of a common set of standards, data models and network (web-based) services. The thematic scope of the Directive covers 34 interdependent themes (Figure 1). The legal obligations are now transposed into the legislation of the 28 EU Member States, and the implementation is ongoing.

Annex I 1. Coordinate reference systems 2. Geographical grid systems 3. Geographical names 4. Administrative units 5. Addresses 6. Cadastral parcels 7. Transport networks 8. Hydrography 9. Protected sites	Annex III 1. Statistical units 2. Buildings 3. Soil 4. Land use 5. Human health and safety 6. Utility and governmental services 7. Environmental monitoring facilities 8. Production and industrial facilities 9. Agricultural and aquaculture facilities 10. Population distribution – demography 11. Area management/ restriction/regulation zones & reporting units 12. Natural risk zones 13. Atmospheric conditions 14. Meteorological geographical features 15. Oceanographic geographical features 16. Sea regions 17. Bio-geographical regions 18. Habitats and biotopes 19. Species distribution 20. Energy Resources 21. Mineral resources
Annex II 1. Elevation 2. Land cover 3. Ortho-imagery 4. Geology	
Implementation 2012/2017	Implementation 2015/2020

Figure 1. Thematic scope of INSPIRE.

2.1.1. Observation Data

The Directive is in its core about the provision of geospatial data. Nonetheless, several of the data themes explicitly focus on observation or measurement data pertaining to different environmental media at specific locations. Those include, but are not limited to, meteorology, hydrology, oceanography, and geology. Technical guidance documents are prepared that cover the encoding of observation data [10], as well as the implementation of services for downloading the data [11]. In addition, INSPIRE includes the “Environmental Monitoring Facilities” (EF) [12]. It describes the facility where measurements are taken, together with all relevant metadata on the measurement process, as well as the measurement data available from the facility. The EF Theme (Figure 2) is relevant for measurements pertaining to all environmental media, and is independent of whether a data theme exists for this particular environmental media.

- Environmental Monitoring Facilities

The INSPIRE EF model can be broken down into three interdependent parts (Figure 2) as follows:

- A description of the Environmental Monitoring Facility (blue classes)
- A description of the measurement methodology, referenced by both the Environmental Monitoring Facility and the Observations (green classes)
- Observations providing the actual spatio-temporal data (red class)

- Observation Model

In addition to the data specification detailing what information related to an Environmental Monitoring Facility is to be provided, the Observation Model provides specifications for concepts that go beyond the EF Theme. The observation model is based on the ISO 19156:2011 (Geographic information—observations and measurements) standard [13].

Structurally, the Observation model is subdivided into four sub-packages (Figure 3):

- *Package 1. Specialised Observations.* Specialisations of the base O&M Observation have been defined through the application of constraints. Specialised observations provide an approach that is tailored to the needs of a multitude of environmental use cases. The following types of observations are defined in [10]:
 - *Profile observation:* Observation representing the measurement of a property along a vertical profile in space at a single time instant.
 - *Trajectory observation:* Observation representing the measurement of a property along a meandering curve in time and space.
 - *Point observation:* Observation that represents a measurement of a property at a single point in time and space.
 - *MultiPointObservation:* Observation that represents a set of measurements all made at exactly the same time but at different locations.
 - *Specimen observation:* Observation that represents a measurement of a property of a Specimen at a single point in time.
 - *GridObservation:* Observation representing a gridded field at a single time instant.
 - *GridSeriesObservation:* Observation representing an evolving gridded field at a succession of time instants.
- *Package 2. Observable Properties.* The INSPIRE Observable Properties model allows for the specification of complex properties. The complex properties enable the creation of composite properties, for the joint provision of strongly linked properties, e.g., wind speed and direction. Further options provided by the INSPIRE Observable Properties model include the description of various statistical aggregates being provided, as well as the provision of constraints on the types of result values that may be provided.
- *Package 3. Processes.* Within the OGC SWE, SensorML [14] is the preferred method for the provision of information on the measurement process. However, this model is quite abstract and not easy to understand and use. Thus, within INSPIRE, a simple feature type has been defined for the provision of information on the measurement process. This feature type is integrated within the EF model, and is presented in Figure 3.
- *Package 4. Observation References.* The Observation References section of the Observation Model provides the mechanisms required for linking between features and OM_Observation object. Thus, an Environmental Monitoring Facility can provide association references to observations stemming from this facility. In the same manner, an OM_Observation can provide a reference to the facility at which the observation was made.

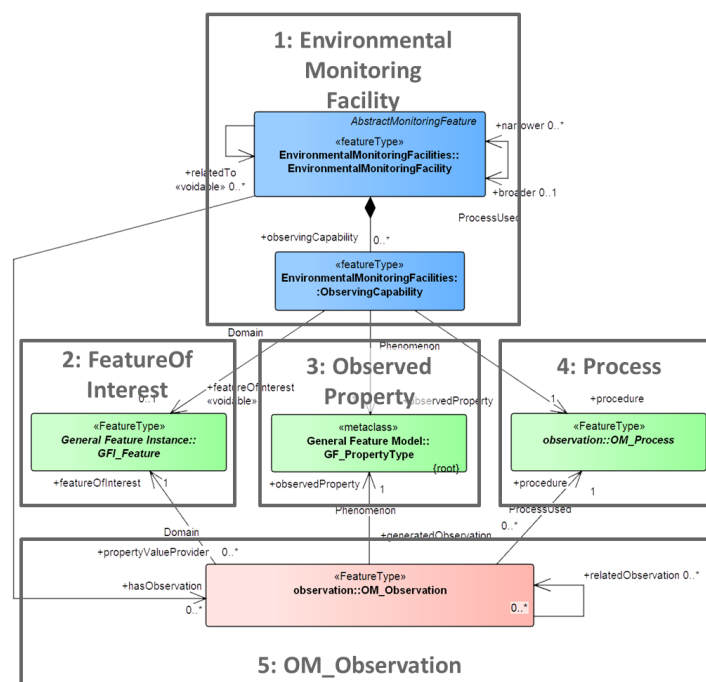


Figure 2. Environmental Monitoring Facilities.

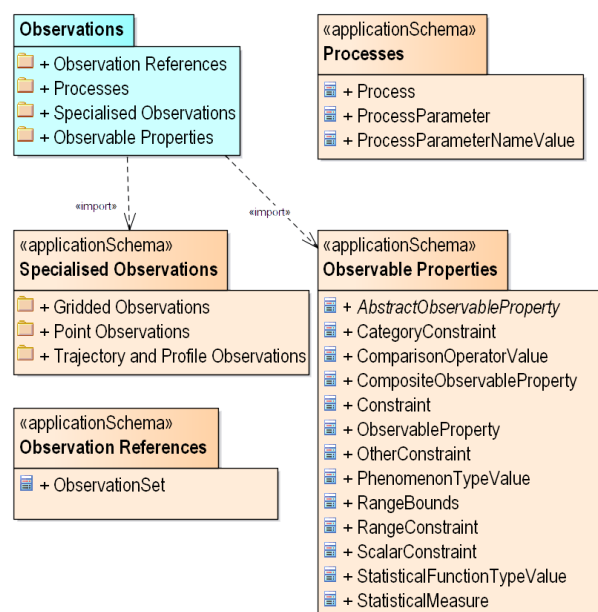


Figure 3. Observation model in INSPIRE.

2.1.2. Download Services

Apart from data encoding, a set of network (web)services are defined in INSPIRE. They cover the: (i) discoverability of datasets through metadata; (ii) data visualisation; (iii) download; and (iv) transformation of data (if not done offline) from their native format to the exchange format required by INSPIRE.

Download services shall provide a means for downloading of whole datasets (pre-defined access), or their subsets (direct access) through the use of a query. In addition, the specification

of INSPIRE Download Services [15] requires functionality for downloading pre-defined data sets that are characterised by the following aspects:

- A pre-defined data set shall have a metadata record, and be discoverable through an INSPIRE conformant discovery service.
- A resolvable link (URL) must be provided, allowing that the dataset be immediately downloaded through a simple HTTP-protocol GET-request.

Guidance for data providers is made available that covers several possible technological options (Table 1).

Table 1. Types of download services in INSPIRE.

Name	Service type	Type of data
1. ATOM Syndication Format	Pre-defined	Whole datasets
2. WCS 2.0 (Web Coverage Service)	Pre-defined and direct access	Coverages (including multidimensional)
3. WFS 2.0 (Web Feature Service)	Pre-defined and direct access	Spatial features (vector)
4. SOS 2.0 (Sensor Observation System)	Pre-defined and direct access	Spatio-temporal observations

All the solutions described in Table 1 might be used to serve spatio-temporal data with certain limitations. Most relevant to our work is the Sensor Observation Service, as it is to a large extent similar to the SensorThings API. The limitations of the SOS approach relate to: (i) the missing functionality of the SOS to filter observations based on their result values, although a corresponding extension has recently been proposed; (ii) lack of pagination; and (iii) the lack of an adopted REST/JSON binding of the SOS standard.

2.2. The OGC SensorThings API

The OGC SensorThings API [8] is an open, geospatially-enabled standard designed to help overcome the interoperability challenge in the IoT domain. The API can be described as “Sensor Web Enablement for the Internet of Things”. SensorThings is designed to interconnect heterogeneous devices, data, and applications through the web. The standard defines a REST-like application programming interface (API) to manipulate the data. A publish–subscribe-based messaging protocol extension for real-time operations is available. The protocol uses the ISO Message Queuing Telemetry Transport (MQTT) standard [16]. The SensorThings API benefits from the use JSON as the data encoding technology. In addition, SensorThings API is based on the OGC SWE standards and OData [3] with the focus on being light-weight, providing comprehensive data model applicable for different IoT use cases, and ease of use. An increasing number of implementations are using the standard for handling spatio-temporal observation data. They span across different application domains such as air pollution monitoring [17], smart city services [18,19] and other Internet of Things use cases [20].

The standard consists of two parts: “Sensing” and “Tasking”. The “Sensing” part, which is the focus of this paper, is for sensing and gathering observations from sensing devices. The “Tasking” part [21] focuses on controlling IoT devices, and is out of the scope for this paper. Therefore, in this paper SensorThings API refers to the “Sensing” part only.

2.2.1. The Data Model

SensorThings incorporates the OGC Observations and Measurement standard [13] for data encoding. The comprehensive data model (Figure 4) makes SensorThings API adaptable to wide variety of IoT use cases.

Conceptually, the data model is subdivided into eight entities (Figure 4) that are further described below. A ‘Thing’ is the central entity in the data model. It can be physical or virtual, and is equipped with one or more ‘Sensor’ to collect Observations. Depending on the use case this can be the object being observed, or the sensor platform, such as a satellite. Each Thing has a Location. If the Thing

is static this Location would never change. However, if the ‘Thing’ is moving, its ‘Location’ is also changing frequently. In those cases the HistoricalLocation is the entity for keeping track of previous Locations of the moving Thing. The data model is defined, so that a Thing can be linked to more than one Location. However, all the Locations that are connected to one Things should be different representations of the same physical location. This feature is useful when multiple representations of a Thing’s Location should be modelled (e.g., lat-lon as well as a room number). Each Thing can have one or more Datastream. The Datastream is an entity for grouping Observations of one Sensor that are observing the same phenomenon, called ObservedProperty. Each Observation has a FeatureOfInterest that it observes, as well as a Datastream.

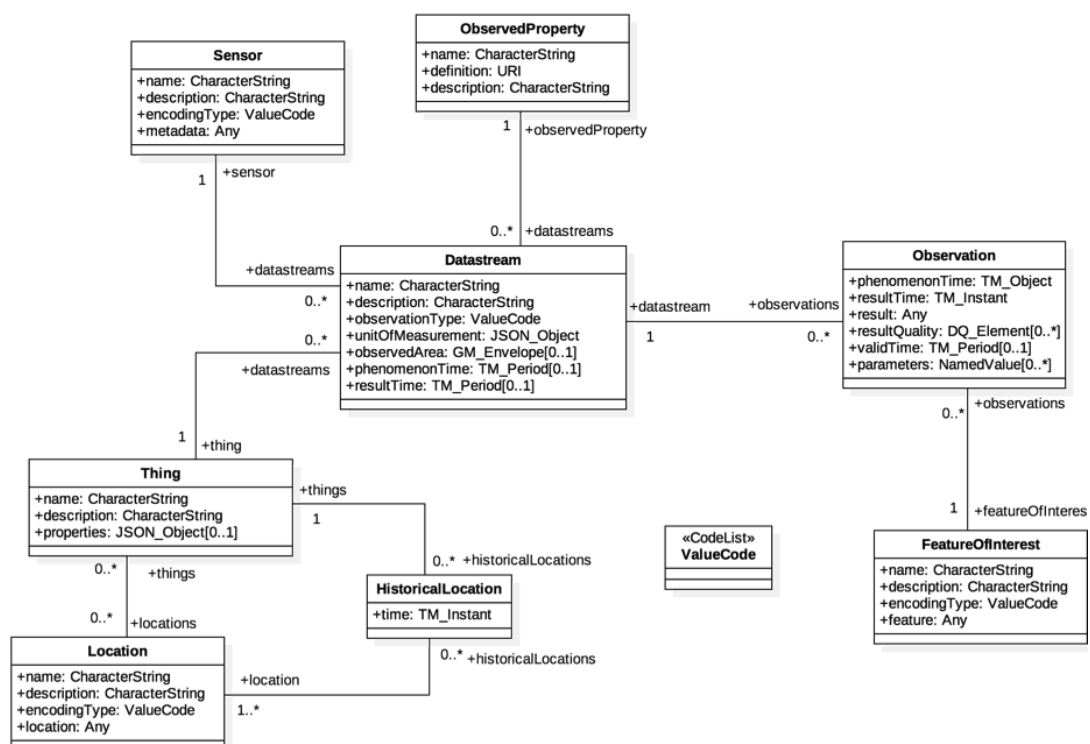


Figure 4. The SensorThings API data model. Source: [8].

2.2.2. RESTful Interface

The SensorThings rest API is loosely based on the OData API [22]. Each of the entity types described in Section 2.2.1 has an entity collection through which these entities can be accessed. Fetching the base URL of the API with a HTTP Get request would return an index document listing the URLs of each of the available entity collections.

A HTTP Get request on a collection returns a list of all entities in the collection. Each entity can also be fetched individually by appending the entity ID, in parentheses, to the entity collection. Besides its specific properties, each entity has an identifier, encoded in JSON as “@iot.id”, a self link, under “@iot.selfLink” and links to other entities as described in the data model. For example, a Thing can have relations to multiple Datastreams. Therefore, each Thing has a navigation link to a collection of Datastreams, listed under “Datastreams@iot.navigationLink” in its JSON, to /Things(<id>)/Datastreams. Likewise, each Datastream is linked to exactly one Thing. Therefore each Datastream has a navigation link to this Thing, listed under “Thing@iot.navigationLink”, to /Datastreams(<id>)/Thing.

A request to a collection is subject to pagination, based on the request parameters and server settings. A client can request the number of entities returned to be limited using the “\$top” query

parameter. The server will not return more than this number of entities, but it can return fewer if it is configured with a lower threshold. If there are more entities to be returned than allowed in a single request, the server adds a link to the result, named “@iot.nextLink”, that returns the next batch of entities. The client itself can also request that a number of entities are skipped, using the query parameter “\$skip”. For example, a client can request entities 11 to 15 by using \$top=5&\$skip=10.

Entities in a collection can be ordered using the “\$orderby” query parameter. The entities can be ordered by one or more of their properties, in ascending or descending order. If a client is not interested in all properties of the requested entities, it can limit the properties to be returned by using the “\$select” query parameter. For example, the query to /Observations?\$select=result,phenomenonTime only returns the result and phenomenonTime of all Observations.

When requesting entities from a collection, a filter can be applied to the entities with the “\$filter” query parameter. This filter can act on any of the properties of the entities in the collection, or any of the properties of related entities. It is, for example, possible to request all Observations that have a phenomenonTime in a certain time range, or all Observations that have a Datastream that has a Thing with a certain name. The filtering options are extensive and include geospatial, mathematical and string functions. In addition, multiple filters can be combined with a Boolean operator (and, or, and not) and parenthesis.

Finally, when requesting entities, it is possible to have related entities be directly included in the response, by using the “\$expand” query parameter. The expanded items can be subjected to all query parameters, including the “\$expand” query parameter. This makes it possible to perform a single request for a Thing, including its Datastreams and their ObservedProperties, and the latest Observation of each of these Datastreams.

2.2.3. Implementations

- FROST-Server

FROST-Server (Fraunhofer OpenSource SensorThings Server) is an open-source implementation of the standard, developed by the German research institute Fraunhofer IOSB, to cover their need for a standards-based, easy to use sensor management and sensor-data storage platform, for use in various research programs. It is written in the Java programming language and can be deployed on servlet containers such as Apache Tomcat or Wildfly. For data persistence, it currently has backends for the PostgreSQL database management system, with either numeric, string or UUID entity identifiers.

Since it was developed to cover the wide range of use-cases that appear in research projects, the focus of development was on feature completeness and extendibility. The server implements the complete specification of the OGC SensorThings API Part 1: Sensing, including all extensions [8]. The open-source nature of the implementation means that users can tune and optimise the implementation for their specific use-case.

- 52°North

52°North provides another comprehensive open source implementation that enables sharing of observation data and corresponding metadata. The implementation is not only based on the OGC Sensor Observation Service (SOS) interface—but also through a range of complementary interfaces (e.g., a non-standardised REST API especially optimised for the development of lightweight client applications).

In 2017, the 52°North Sensor Web team completed a prototype for different ways to evaluate how to enhance the 52°North SOS implementation with support for the SensorThings API standard. Due to the strong similarities between the SOS and SensorThings API interfaces as well as data models (i.e., ISO/OGC Observations and Measurements is also a core foundation of the SensorThings API) this enhancement was rather easy to achieve. Currently, there is ongoing work at 52°North to advance

the prototypical developments into a stable module that will be published together with the next major SOS release expected for 2018.

- **SensorUp SensorThings**

SensorUp, a Calgary-based startup, developed the first compliant SensorThings implementation. It is considered to be an OGC reference implementation. The implementation is Java-based and the current persistence system is a PostgreSQL database. The implementation is tuned for scalability without loss of performance. In addition to server development, SensorUp provides multiple clients and client libraries to make SensorThings even easier to use for client developers. Moreover, a wide variety of documentations, interactive SDKs, tutorials, and examples are provided by SensorUp for developers.

- **GOST**

GOST is an open source implementation of SensorThings based on the GO programming language. This implementation was developed by Geodan, an Amsterdam based geo-ICT company. The implementation passed the SensorThings test suite and is OGC certified. It also has MQTT extension implemented. GOST is considered as alpha software and is not yet considered appropriate for customer use. GOST provides docker images as their easy deployment mechanism as well as binaries for users who do not prefer Docker.

- **Mozilla**

Mozilla has a Node implementation of SensorThings. This implementation is open source and almost passed all the OGC test suite tests. The implementation uses PostgreSQL for persistence layer. The development has not been active since February 2017.

- **CGI Kinota Big Data**

CGI developed a modular implementation of SensorThings called Kinota Big Data. Kinota is designed to support different persistence platforms ranging from RDBMS to NoSQL databases. The current implementation supports Apache Cassandra. Kinota implements only a subset of the SensorThings requirements. It is also a Java-based implementation and is provided under the GNU license.

3. SensorThings API for INSPIRE

3.1. Alignment of Data Specifications

To determine whether and how the SensorThings API can be utilised to fulfil the INSPIRE requirements pertaining to the data scope defined within the legislation, an alignment between these two data models must be specified. We approached this alignment on two hierarchical levels. Initially, the SensorThings API data model was analysed to determine which class or classes best correspond to the classes specified for the INSPIRE EF Theme [12], as well as for the specialised observations from the observation model of the INSPIRE Generic Conceptual Model (GCM) [23]. Table 2 provides an overview of the proposed alignment for EF as well as for the base O&M Observation type. Secondly, once corresponding classes were identified, the individual attributes were aligned. In addition, necessary extensions were specified for attributes not available within the core SensorThings API data model.

Table 2. Mapping between INSPIRE Feature Types and SensorThings API Entities.

INSPIRE Feature Type	Definition	SensorThings API Entity	Comments
1. Environmental Monitoring Facility	A georeferenced object directly collecting or processing data about objects whose properties (e.g., physical, chemical, biological or other aspects of environmental conditions) are repeatedly observed or measured. Can also host other environmental monitoring facilities.	“Thing’ and “Location’	The ObservingCapability FeatureType associated to the Environmental Monitoring Facility via composition corresponds to a “Datastream’.
2. Feature Of Interest	Feature that is the subject of the observation, and carries the observed property being investigated.	“FeatureOfInterest’	
3. Observed Property	The Property Type for which the OM_Observation:result provides an estimate of its value.		
4. Process	Process used to generate the result.	“Sensor’	
5. OM_Observation	Links the domain and range of the observation being described, together with all relevant metadata required for the interpretation of this observation.	“Datastream’ and “Observation’	

Once corresponding classes were identified, the individual attributes were aligned to the requirements of the EF data specification. While all requirements of the SensorThings API data model could be mapped to mandatory data from EF, EF requirements were identified that could not be mapped to the SensorThings API data model. However, the addition of a properties section, allowing for the inclusion of a block of JSON encoded data, has already been recommended for inclusion in the upcoming 1.1 version of the SensorThings API standard. This properties section has been utilised within the alignment tables (Appendix A), allowing for full compliance with INSPIRE. Tables with additional details for each attribute are supplemented [online](#).

The specialised observations from the observation model of the GCM are defined by constraint. The ramifications of these constraints were analysed to determine a semantically equivalent usage of the SensorThings API data model. The following models for provision of the INSPIRE specialised observation types have been preliminarily sketched illustrating options for implementation. These should be further analysed before wide-spread use:

- *Point observation*: Datastream with one Observation pertaining to a single FeatureOfInterest.
- *Point Time Series Observation*: Datastream with multiple Observations over time pertaining to a single FeatureOfInterest.
- *Multi Point Observation*: Datastream with multiple Observations pertaining to different FeaturesOfInterest. Some additional grouping of the FeaturesOfInterest may be required.
- *Grid Observation*: Datastream with one Observation providing a complex result, FeatureOfInterest is a Grid. The Coverage Implementation Schema (CIS 1.1) should be explored for the provision of gridded coverages, as GeneralGridCoverage seems well suited for this purpose, and a JSON encoding is provided.
- *Grid Series Observation*: Datastream with multiple Observations providing complex results, FeatureOfInterest is a Grid. CIS 1.1 [24] should be explored for the provision of gridded coverages, as GeneralGridCoverage seems well suited for this purpose, and a JSON encoding is provided.
- *Profile Observation*: Depending on the type of Profile, various options exist for encoding via the SensorThings API. These range from the model sketched for Multi Point Observation, whereby all FeaturesOfInterest must share the same values for Lat/Long, over the use of a MultiDatastream providing a depth indicator (depth in m or pressure) as a second ObservedProperty to the utilization of grid types as specified within the constraints on the INSPIRE model.
- *Trajectory Observation*: Datastream with multiple Observations with a varying FeatureOfInterest. Some additional grouping of the FeaturesOfInterest may be required in order to expose the trajectory to client applications.

In addition, the list of observation types available from the OGC Naming Authority [25] does not include all observation types required for INSPIRE. These should either be appended to this list, or alternatively be made available from an alternative source.

3.2. Download Service Operations

As clarified in Section 2.1.2, INSPIRE distinguished between: (i) pre-defined; and (ii) direct access download services. The implementation of the former requires functionality for downloading whole datasets. They shall be documented through metadata which is exposed online through a discovery service, following the specifications of the OGC Catalogue Service for the Web—CSW [26]. A resolvable link (URL) shall be provided within each metadata record through which the dataset can be downloaded by sending a simple HTTP-protocol GET-request.

3.2.1. Pre-Defined Dataset

The INSPIRE Network Service Regulation [15] defines an abstract service model through a set of operations that each download service shall implement. No ability to query datasets or select user-defined subsets of datasets is foreseen for the pre-defined datasets. The required operations, together with the proposed equivalent in SensorThingsAPI are provided in Table 3.

Table 3. Operations for pre-defined dataset download services

Service operation	Definition	Mapping to SensorThings API
1. Get Download Service Metadata	Provides all necessary information about the service, the available Spatial Data Sets, and describes the service capabilities.	OpenAPI [27] document. This would require a mapping of the relevant (mostly static) contents of OGC GetCapabilities responses to OpenAPI.
2. Get Spatial Dataset	Retrieval a dataset as a whole, as defined by the data provider	The simplest option would be to map this operation to “Datastreams”, i.e., each datastream to deliver an INSPIRE dataset that is described separately through metadata. As a difference to the SOS offering concept [11], the datasets defined by the SensorThings API are restricted to one observed property. Thus, “MultiDatastreams” shall be used if the dataset contains more than one “ObservedProperty”. Alternatively, the powerful query mechanism of SensorThings API can be used to request information spread into multiple “Datastreams”, including “Observations”, “Observed Properties”, “Thing”, “Locations” and “Fol”, all filtered by time interval, area, or other criteria. Sample requests are provided in Appendix B.
3. Describe Spatial Dataset	Description of metadata about a dataset and all types of Spatial Objects contained in the Spatial Dataset.	This can be mapped to: (i) “Datastream” if the dataset consists of a single ObservedProperty; and (ii) “MultiDatastream” if the dataset contains more than one ObservedProperty.
4. Link Download Service	Allows the declaration of the availability of a Download Service while maintaining the downloading capability at the Public Authority or a Third Party location.	

This mapping shows that the SensorThings API is generally capable of providing the functionality needed for implementing the pre-defined dataset download functionality that is mandatory for INSPIRE Download Services. However, there are several open issues to be addressed with regard to the mapping proposed in Table 3:

- *Capabilities for SensorThings API services.* The standard provides a RESTful API for access to data. This is a convenient approach which allows easy uptake and immediate use by mainstream developers. The *GetCapabilities* operation included in the majority of OGC standards is not available. A solution for INSPIRE might be to use the OpenAPI standard (former Swagger) [27] to document the service, considering the required, mostly static, elements of the *Get Download Service Metadata* and *Describe Spatial Dataset* operations.
- *Dataset in SensorThings API.* The concept of a “dataset” does not exist in the SensorThings API standard. If following our proposal to equalise a dataset to a “Datastream”, it should represent a logically consistent grouping of individual observations.
- *Support for request parameters.* Request parameters are defined in [15] for the *Get Spatial Dataset* operation. Users should be able to request their datasets or spatial objects in any of the: (i) offered coordinate reference systems (CRS); and (ii) natural languages; and (iii) through their spatial

data set identifier. The REST-like interface of SensorThings API does not support the first two parameters.

3.2.2. Direct Access

In addition to the mandatory operations described in Section 3.2.1, non-mandatory “direct access” operations may be implemented to ensure that users can acquire the desired data in a flexible manner (i.e., sub-setting the datasets by properties such as time periods and spatial extends). From our perspective they would be easy to implement through SensorThings API. Their mappings are provided in Table 4. Even though direct access operations are not mandatory, from our perspective, they are the ones that are, and would increasingly be, desired by users, as they really add value to the data provider infrastructure.

Table 4. Operations for direct access download services

Service operation	Definition	Mapping to SensorThings API
1. Get Spatial Object	Retrieval of Spatial Objects based upon a query.	Request that resolves to an Observation entity in a collection together with its id.
2. Describe Spatial Object Type	Description of the specified Spatial Objects types.	Metadata provided as part of the Datastreams or Observations downloaded from a SensorThings API endpoint.
3. Link Download Service	Allows the declaration of the availability of a Download Service while maintaining the downloading capability at the Public Authority or a Third Party location.	

Similar to the *Get Spatial Dataset* operation, the *Get Spatial Object* operation shall support requests for coordinate reference system and language that are not supported by the standard.

4. Discussion and Conclusions

Within this manuscript we: (i) outline the challenges associated with the provision of spatio-temporal data in spatial data infrastructures; (ii) describe the contemporary technological scenery and emerging standardisation initiatives, and, most importantly, (iii) propose a solution that would allow the new SensorThings API to be considered as an INSPIRE solution. In summary, we consider that **no major blocking factors exist for proposing SensorThings as an INSPIRE solution**.

As outlined in Section 2.1, INSPIRE provides a transparent and straightforward means for encoding the location of environmental monitoring facilities and observation data. Nonetheless, there are several issues to be addressed. They include:

- *Definition of a dataset.* Datasets in INSPIRE should include observations for a predefined period of time and for logically consistent geographical entities usually observed by the same procedure. Similar to SOS, the concept of a dataset is unknown to the SensorThings API. This we see as a broader issue that should be addressed within INSPIRE.
- *SensorThings API Extension Points.* The data model of the SensorThings API has been concisely tailored to the requirements of sensors within the IoT while the INSPIRE data specifications require the provision of additional contextual information as required within an SDI. While some attributes required by INSPIRE can be directly aligned with attributes from the SensorThings API, additional attributes are required. For this purpose, we utilized the currently proposed extension (Appendix A) to the SensorThings API 1.1, whereby a properties attribute of type JSON_Object, as presently provided by the class Thing, is appended to the classes Datastream and Sensor. All additional requirements stemming from INSPIRE can be supported by providing

this within the JSON properties structure; for FeatureOfInterest, the feature attribute serves the same purpose.

- *Metadata for services.* The SensorThings API provides a RESTful API for simultaneous access to both data and metadata, thus the API does not require a separate operation to request metadata. However, the INSPIRE Directive mandates such an operation. Considering the fact that the vast majority of required metadata elements are static, it would be fairly easy to document STA services through the OpenAPI specification [27]. Technical guidance should be released to describe such an approach.
- *Request for CRS and Language.* As outlined in Section 3.2, requesting datasets or individual observations in a particular natural language or coordinate reference system is not supported by SensorThings API. Satisfying this requirement would require a workaround that goes beyond the current version of the standard. When addressing the issue of CRS in particular, two aspects should be considered. Firstly, the provision of functionality for requesting CRS would add complexity to the service interface which might be conflicting with the overarching objective of the standard to be as developer friendly as possible. Secondly, the SensorThings API suggests GeoJSON for data encoding, and the conformance tests only test with it. This requirement poses a limitation, as all coordinates shall be provided in a geographic coordinate reference system, using the World Geodetic System 1984 datum [28].

In our future research, we would focus on the following:

- *Comparison with other download services.* Thus far, we analysed the SensorThings API from a legislative and technical perspective without going deeper into the analysis of how our proposed solution would fit with the rest of the implementations in the pan-European spatial data infrastructure. This deserves to be further investigated from at least two angles: (i) comparison with other possible solutions such as SOS and WFS; and (ii) approaches combining two or more solutions, e.g., where static data are handled through a WFS, and SensorThings API handles the spatio-temporal data.
- *Specialised observation types.* INSPIRE defines a set of specialised observation types (Section 3). For some, such as the *Point observation* and *Point Time Series Observation*, implementation options are quite straightforward. Others, foremost those pertaining to grids such as the *Grid Observation* and *Grid Series observation*, but also *Profile observation* and *Trajectory observation*, would require additional attention. For the provision of observations pertaining to grids (*Grid Observation* and *Grid Series observation*), available grid encoding standards must be analysed for suitability. CIS 1.1 [24] seems promising in this regard. For the encoding of *Multi Point Observation*, *Profile Observation* and *Trajectory Observation* methods for grouping a set of FeaturesOfInterest should be explored, as such a mechanism would provide easier handling for client applications. For the provision *Profile Observation*, alternatives utilising *MultiDatastreams* or *Grids* should also be taken into consideration.
- *Asynchronous transactions for INSPIRE download services.* The SensorThings API offers out-of-the-box functionality for publish–subscribe-based messaging through the use of MQTT [16]. At the same time, INSPIRE is, legislatively, bound to the request–response paradigm for data exchange. Given the rapid development of the IoT with its constraint devices where the latency of networks is challenging, the use of publish–subscribe services should be further investigated.
- *Sensor Tasking.* The possibility to assign tasks to environmental sensors in a standardised manner goes beyond the legal requirements of INSPIRE. However, considering the rapid growth of the number of connected IoT devices, this would be a very interesting, particularly for the planning and implementation of measurement campaigns. Within this context, the “SensorThings API, Part 2—Tasking Core” Candidate Standard [21] is to be investigated.
- *Other standards.* From our perspective, the approach, demonstrated in this paper, for analysing possible technological solutions for INSPIRE is applicable for other standards as well.

Most prominent from that perspective would be to investigate the emerging Web Feature Service 3.0 (WFS 3.0) [29] as a potential INSPIRE solution.

Supplementary Materials: The following is available online at <https://github.com/DataCoveEU/SensorThings/tree/master/INSPIRE>, Figure S1: Mapping between INSPIRE and SensorThings API attributes.

Author Contributions: The description of STA was prepared by S.L., T.K. and H.v.d.S. A.K. and M.L. provided an overview of INSPIRE. The alignment of data requirements between STA and INSPIRE was done by K.S., S.G. and M.B. S.J., H.v.d.S. and A.K. analysed the alignment of STA with INSPIRE download service specifications. A.K. and K.S. built the overall storyline, contributed to all sections of the paper, and coordinated the input by the rest of the co-authors.

Funding: The work on data model alignment between STA and INSPIRE was partially performed under the BRGM contract “SensorThings API Deployment within the French Groundwater Information Network”. The contribution of Simon Jirka (52 °North GmbH) was supported by the project COLABIS (German Federal Ministry of Education and Research, programme Geotechnologien, under grant agreement No. 03G0852A) as well as the European projects BRIDGES (European Union’s Horizon 2020 research and innovation programme under grant agreement No. 635359) and ODIP II (European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654310).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
CIS	Coverage Implementation Schema
CRS	Coordinate Reference System
CSW	Catalogue Service for the Web
EF	Environmental Monitoring Facilities
GNU	GNU General Public License
IoT	Internet of Things
ISO	International Organization for Standardization
MQTT	Message Queuing Telemetry Transport
OData	Open Data Protocol
OpenAPI	OpenAPI Specification
RDBMS	Relational Database Management System
SDI	Spatial Data Infrastructure
SDK	Software Development Kit
STA	SensorThings API
SWE	OGC Sensor Web Enablement Suite
WFS	Web Feature Service
XML	Extensible Markup Language

Appendix A. Alignment of INSPIRE EF Requirements to SensorThings API

Appendix A.1. EF SamplingPoint

Table A1. Alignment between the INSPIRE “Environmental Monitoring Facility” class and STA.

Environmental Monitoring Facility	Class	Attribute	CR on ST Part 1 for Properties Extension
gml:id	THINGS	ID	
gml:identifier	THINGS	ID	
inspire Id			
localId	THINGS	ID	
namespace	THINGS	PROPERTIES/namespace	X
name	THINGS	NAME	
media Monitored	THINGS	PROPERTIES/mediaMonitored	X
geometry	LOCATIONS		
gml:id	LOCATIONS	ID	
srsDimension	LOCATIONS	GEOM	
srsName	LOCATIONS	GEOM	
gml:Point	LOCATIONS	GEOM	
observing Capability	DATASTREAMS		
gml:id	DATASTREAMS	ID	
observing Time	DATASTREAMS		
gml:id	DATASTREAMS	ID	
beginPosition	DATASTREAMS	PHENOMENON_TIME_START	
endPosition	DATASTREAMS	PHENOMENON_TIME_END	
process Type	DATASTREAMS	PROPERTIES/processType	X
result Nature	DATASTREAMS	PROPERTIES/resultNature	X
online Resource	DATASTREAMS		
featureOfInterest	OBSERVATIONS	FEATURE_ID	
observedProperty	DATASTREAMS	OBS_PROPERTY_ID	
procedure	DATASTREAMS	SENSOR_ID	
measurement Regime	THINGS	PROPERTIES/measurementRegime	X
mobile	THINGS	PROPERTIES/mobile	X
operational Activity	THINGS	PROPERTIES	X
Period			
gml:id	THINGS	ID	
beginPosition	THINGS	PROPERTIES/beginTime	X
endPosition	THINGS	PROPERTIES/endTime	X
comment	THINGS	DESCRIPTION	

Appendix A.2. Process

Table A2. Alignment between the INSPIRE “Process” class and STA.

Type		Class	Attribute	CR on ST Part 1 for Properties Extension
OM_Process	gml:id	SENSORS	ID	
	gml:identifier	SENSORS	ID	
	inspireId	SENSORS		
	localId	SENSORS	ID	
	namespace	SENSORS	PROPERTIES/namespace	X
	name	SENSORS	NAME	
	type	SENSORS	PROPERTIES/type	
	documentation	SENSORS	DESCRIPTION	
	responsible Party	SENSORS	PROPERTIES/responsibleParty	X
RelatedParty	individualName	SENSORS	PROPERTIES/responsibleParty/individualName	X
	organisationName	SENSORS	PROPERTIES/responsibleParty/organisationName	X
	positionName	SENSORS	PROPERTIES/responsibleParty/positionName	X
	contact//address//language	SENSORS	PROPERTIES/responsibleParty/language	X
	contact//address//adminUnit//SpellingOfName	SENSORS		
	text	SENSORS	PROPERTIES/responsibleParty/adminUnit	X
	script	SENSORS	PROPERTIES/responsibleParty/script	X
	address//locatorDesignator	SENSORS	PROPERTIES/responsibleParty/locatorDesignator	X
	address//postCode	SENSORS	PROPERTIES/responsibleParty/postCode	X
	electronicMailAddress	SENSORS	PROPERTIES/responsibleParty/electronicMailAddress	X
	telephoneVoice	SENSORS	PROPERTIES/responsibleParty/telephoneVoice	X
	website	SENSORS	PROPERTIES/responsibleParty/website	X
	role	SENSORS	PROPERTIES/responsibleParty/role	X

Appendix A.3. Feature of Interest

Table A3. Alignment between the INSPIRE “Feature of Interest” class and STA.

Type	Class	Attribute	CR on ST Part 1 for Properties Extension
gml:id	FEATURES	ID	
gml:identifier	FEATURES	ID	
sf:type	FEATURES	PROPERTIES/type	X
sf:sampledFeature	FEATURES	PROPERTIES/sampledFeature	X
sams:shape	FEATURES	FEATURE	
gml:id	FEATURES	GEOM	
srsName	FEATURES	GEOM	
gml:geometry	FEATURES	GEOM	

Appendix A.4. Observation

Table A4. Alignment between the INSPIRE “Observation” class and STA.

Type	Class	Attribute
gml:id	DATASTREAMS	ID
phenomenonTime		
gml:id	DATASTREAMS	ID
beginPosition	DATASTREAMS	PHENOMENON_TIME_START
endPosition	DATASTREAMS	PHENOMENON_TIME_END
resultTime		
gml:id	DATASTREAMS	ID
beginPosition	DATASTREAMS	PHENOMENON_TIME_END
procedure	DATASTREAMS	SENSOR_ID
observedProperty	DATASTREAMS	OBS_PROPERTY_ID
featureOfInterest	OBSERVATIONS	FEATURE_ID
result	OBSERVATIONS	RESULT_NUMBER

Appendix B. Sample requests to SensorThings API Services*Appendix B.1. Request of an Air Temperature and Humidity Dataset*

The query below returns, for a given “Thing”, its metadata, the “Location”, and “Datastreams” that measure Air Temperature and Humidity. For each “Datastream”, the “Sensor” and “ObservedProperty” metadata, all observations for a given period, and for each “Observation” the id of the “FeatureOfInterest”. The example covers a static station. That is why the FoI is always the same, coinciding with the “Location” of the “Thing”, so these data could be left out.

```

https://sensorthingsserver.eu/SensorThingsService/v1.0/Things(10)
?$select=name,id
&$expand=Locations,Datastreams(
  $filter=ObservedProperty/id eq 2 or ObservedProperty/id eq 5
  ;$select=name,id,unitOfMeasurement
  ;$expand=
    Sensor($select=id,name,description)
    ,ObservedProperty($select=id,name,definition,description)
    ,Observations(
      $filter=phenomenonTime ge 2018-02-01T00:00:00Z and phenomenonTime lt 2018-03-01T00:00:00Z
      ;$orderby=phenomenonTime asc
      ;$select=phenomenonTime,result
      ;$expand=
        FeatureOfInterest(
          $select=id
        )
      )
    )
  )
)

```

Figure A1. Sample request to a SensorThings API server for retrieving an air temperature and humidity dataset.

Appendix B.2. Request of a Water Level Dataset

This query below returns all “Datastreams” in a given area that measure a “Water Level” including the “Datastreams” marked as “forecast”, including their “Thing” and the Thing’s “Location”, “Sensor”, “ObservedProperty” and “Observations”.

```

https://sensorthingsserver.eu/SensorThingsService/v1.0/Datastreams?
$filter=ObservedProperty/definition eq 'http://dbpedia.org/page/Water_level'
and properties/type eq 'measurement'
and st_within(Thing/Locations/location,geography'POLYGON ((11.4 45.7,11.4 45.4,11.8 45.4,11.8))')
&$select=id,name,properties,unitOfMeasurement
&$expand=
  Thing(
    $select=id,name,description
    ;$expand=Locations
  )
  ,Sensor($select=id,name,description)
  ,ObservedProperty($select=id,name,definition,description)
  ,Observations(
    $filter=phenomenonTime ge 2018-02-01T00:00:00Z and phenomenonTime lt 2018-03-01T00:00:00Z
    ;$orderby=phenomenonTime asc
    ;$select=phenomenonTime,result
  )
)

```

Figure A2. Sample request to a SensorThings API server for retrieving a water level dataset.

References

1. European Commission. *Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 Establishing an Infrastructure for Spatial Information in the European Community (INSPIRE)*; European Parliament and of the Council of the European Union: Brussels, Belgium, 2007.
2. European Commission. Building a European Data Economy. Communication from the Commission 2017. Available online: https://eur-lex.europa.eu/content/news/building_EU_data_economy.html (accessed on 22 February 2018).
3. Botts, M.; Percivall, G.; Reed, C.; Davidson, J. OGC® sensor web enablement: Overview and high level architecture. In *GeoSensor Networks*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 175–190.
4. Bröring, A.; Jirka, S.; Kotsev, A.; Spinsanti, L. Making the sensor observation service INSPIRE compliant. In *Proceedings of the INSPIRE Conference 2013, Florence, Italy, 23–27 June 2013*.
5. Technical Guidance for Implementing INSPIRE Download Services Using the OGC Sensor Observation Service and ISO 19143 Filter Encoding. Available online: <https://inspire.ec.europa.eu/id/document/tg/download-sos> (accessed on 1 June 2018).
6. Eric Schmidt: Every 2 Days We Create As Much Information As We Did Up To 2003. Available online: <https://techcrunch.com/2010/08/04/schmidt-data/?guccounter=1> (accessed on 1 June 2018).

7. Lutz, M.; Bernard, L.; Portele, C.; Hansen, T.; Tiainen, E.; Lucchi, R. INSPIRE – What if? Summary report from the What if...? sessions at the 2017 INSPIRE Conference. Available online: <https://ec.europa.eu/jrc/en/publication/inspire-what-if-summary-report-what-if-sessions-2017-inspire-conference> (accessed on 30 April 2018).
8. Liang, S.; Huang, C.Y.; Khalafbeigi, T. OGC SensorThings API-Part 1: Sensing. OGC® Implementation Standard. 2016. Available online: <http://docs.openegeospatial.org/is/15-078r6/15-078r6.html> (accessed on 14 June 2018).
9. Cetl, V.; Tomas, R.; Kotsev, A.; de Lima, V.N.; Smith, R.S.; Jobst, M. *Establishing Common Ground through INSPIRE: The Legally-Driven European Spatial Data Infra-Structure*; Springer Lecture Notes in Geoinformation and Cartography; Springer: Cham, Germany, 2018.
10. Guidelines for the use of Observations and Measurements and Sensor Web Enablement-related standards in INSPIRE. Available online: inspire.ec.europa.eu/id/document/tg/d2.9-o&m-swe/3.0 (accessed on 14 June 2018).
11. Open Geospatial Consortium (OGC). *Sensor Observation Service Interface Standard*; Open Geospatial Consortium Interface Standard; Open Geospatial Consortium (OGC): Wayland, MA, USA, 2012; 12–006.
12. INSPIRE Data Specification on Environmental Monitoring Facilities—Technical Guidelines. Available online: <https://inspire.ec.europa.eu/id/document/tg/ef> (accessed on 10 April 2018).
13. ISO/TC211. *ISO 19156: 2011-Geographic Information—Observations and Measurements-International Standard*; International Organization for Standardization: Geneva, Switzerland, 2011.
14. Botts, M.; Robin, A. *OpenGIS Sensor Model Language (SensorML) Implementation Specification*; OpenGIS Implementation Specification OGC: Wayland, MA, USA, 2007.
15. European Commission. *COMMISSION REGULATION (EU) No 1088/2010 of 23 November 2010 Amending Regulation (EC) No 976/2009 as Regards Download Services and Transformation*; European Parliament and of the Council of the European Union: Brussels, Belgium, 2010.
16. ISO/IEC 20922:2016, Information Technology—Message Queuing Telemetry Transport (MQTT) v3.1.1. Available online: <https://www.iso.org/standard/69466.html> (accessed on 10 February 2018).
17. Grothe, M.; Broecke, J.V.; Carton, L.; Volten, H.; Kieboom, R. Smart Emission: Building A Spatial Data Infrastructure for an Environmental Citizen Sensor Network. In Proceedings of the Geospatial Sensor Webs Conference 2016 (GSW 2016), Münster, Germany, 29–31 August 2016.
18. Soto, J.A.; Werner-Kytölä, O.; Jahn, M.; Pullmann, J.; Bonino, D.; Pastrone, C.; Spirito, M. Towards a federation of smart city services. In Proceedings of the International Conference on Recent Advances in Computer Systems (RACS 2015), Hail, Saudi Arabia, 30 November–1 December 2015; pp. 163–168.
19. Trilles, S.; Luján, A.; Belmonte, Ó.; Montoliu, R.; Torres-Sospedra, J.; Huerta, J. SEnviro: A sensorized platform proposal using open hardware and open standards. *Sensors* **2015**, *15*, 5555–5582. [[CrossRef](#)] [[PubMed](#)]
20. Huang, C.Y.; Wu, C.H. A Web Service Protocol Realizing Interoperable Internet of Things Tasking Capability. *Sensors* **2016**, *16*, 1395. [[CrossRef](#)] [[PubMed](#)]
21. SensorThings API Part 2—Tasking. Available online: <http://www.openegeospatial.org/pressroom/pressreleases/2739> (accessed on 24 April 2018).
22. ISO/IEC 20802-1:2016. Open Data Protocol (OData). Available online: <https://www.iso.org/standard/69208.html> (accessed on 24 April 2018).
23. INSPIRE Generic Conceptual Model. Available online: https://inspire.ec.europa.eu/documents/Data_Specifications/D2.5_v3.4rc3.pdf (accessed on 24 April 2018).
24. OGC Coverage Implementation Schema (CIS) v1.1. Available online: <http://docs.openegeospatial.org/is/09-146r6/09-146r6.html> (accessed on 24 April 2018).
25. OGC Naming Authority. Available online: <http://www.opengis.net/def/observationType/OGC-OM/2.0/> (accessed on 24 April 2018).
26. Nogueras-Iso, J.; Zarazaga-Soria, F.J.; Béjar, R.; Álvarez, P.; Muro-Medrano, P.R. OGC Catalog Services: A key element for the development of Spatial Data Infrastructures. *Computers & Geosciences* **2005**, *31*, 199–209.
27. OpenAPI Specification. Available online: <https://github.com/OAI/OpenAPI-Specification> (accessed on 10 March 2018).

28. Butler, H.; Daly, M.; Doyle, A.; Gillies, S.; Hagen, S.; Schaub, T. *The Geojson Format*; Technical report; Internet Engineering Task Force (IETF): Fremont, CA, USA, 2016.
29. Web Feature Service 3.0 and Filter Encoding Standards. Available online: https://github.com/opengeospatial/WFS_FES (accessed on 10 March 2018).



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).