

## Article

# Semantic Profiles for Easing SensorML Description: Review and Proposal

Paolo Tagliolato <sup>\*</sup>, Cristiano Fugazza , Alessandro Oggioni  and Paola Carrara 

Institute for Electromagnetic Sensing of the Environment, National Research Council (IREA-CNR), v. Bassini 15, 20133 Milan, Italy

<sup>\*</sup> Correspondence: tagliolato.p@irea.cnr.it

Received: 15 May 2019; Accepted: 25 July 2019; Published: 31 July 2019



**Abstract:** The adoption of Sensor Web Enablement (SWE) practices by sensor maintainers is hampered by the inherent complexity of the Sensor Model Language (SensorML), its high expressiveness, and the scarce availability of editing tools. To overcome these issues, the Earth Observation (EO) community often recurs to SensorML *profiles* narrowing the range of admitted metadata structures and value ranges. Unfortunately, profiles frequently fall short of providing usable editing tools and comprehensive validation criteria, particularly for the difficulty of checking value ranges in the multi-tenanted domain of the Web of Data. In this paper, we provide an updated review of current practices, techniques, and tools for editing SensorML in the perspective of profile support and introduce our solution for effective profile definition. Beside allowing for formalization of a broad range of constraints that concur in defining a metadata profile, our proposal closes the gap between profile definition and actual editing of the corresponding metadata by allowing for ex-ante validation of the metadata that is produced. On this basis, we suggest the notion of Semantic Web SensorML profiles, characterized by a new family of constraints involving Semantic Web sources. We also discuss implementation of SensorML profiles with our tool and pinpoint the benefits with respect to the existing ex-post validation facilities provided by schema definition languages.

**Keywords:** sensors; SensorML; profiles; editor; semantics

## 1. Introduction

Deployment of sensor resources is drastically increasing in recent years, addressing the call for continuous and widespread monitoring of the environment. Sensors collect heterogeneous observations from space and on Earth automatically, and their rapid technical evolution enables the constant refinement of information granularity. Consequently, sensor systems are expected to provide a data deluge that is likely to make the former unmanageable unless proper data handling facilities are put in place. For these reasons, to discover, integrate, and exploit sensor data (and also to preserve them for future needs), we need to accurately record sensor information; otherwise, usability of data coming from sensor networks is hampered. These criticalities impose accurate knowledge of sensors through their metadata.

The Sensor Model Language (SensorML) is the proposal of the Open Geospatial Consortium (OGC) for addressing these issues in the context of the Sensor Web Enablement (SWE) initiative [1]. The SensorML schemas [2,3] enable provision of standardized sensor metadata, hence constitute key components to foster interoperability. Nevertheless, SensorML does not grant correct and exhaustive sensor metadata, as it has not reached widespread adoption in the Earth Observation (EO) community. SensorML complexity, the scarcity of dedicated editing tools, and the flexible characteristics of the schema, which allows for multiple, equivalent descriptions of the same sensor [4], make provision of SWE sensor metadata a challenging task.

The EO community recurs to SensorML *profiles* to tackle these issues: By imposing additional constraints, profiles provide less flexibility with respect to the originating schema. Nevertheless, even with SensorML profiles, the creation of sensor metadata is problematic because, typically, profiles are accompanied by tools for *validating* documents but not for *composing* them easily. Furthermore, technologies for constraining XML documents to specific rules lack the ability to enforce requirements involving resources from the Semantic Web, such as in many profiles developed for SensorML. Hence, documents are typically created manually by XML experts, often via general purpose XML authoring software, and only then (partly) checked against the profile, thus preventing direct metadata authoring by domain or sensor experts.

In this paper, beside providing an updated review of the available practices, techniques, and tools for editing SensorML (Section 2), we introduce our customizable, template-based tool for editing any XML-based metadata. Both the review and the discussion of our tool are set in the context of the practices for definition of SensorML community profiles (Section 3); the rationale for this is that the template language that drives customization of our tool allows for formalizing many of the constraints that are required by profile definition. Since this template language exploits data sources made available as SPARQL endpoints [5], it is also possible to formalize and enforce a broad range of semantic constraints that, albeit informally required in many profiles, cannot be easily expressed by state of the art definition strategies. This leads to the introduction of a new category of *semantic* SensorML profiles, based on a new family of requirements involving data sources in the Web of Data (Section 4).

Moreover, our tool closes the gap between profile definition and actual editing of the corresponding metadata: In fact, accurate customization of our application allows for *ex ante* validation of the metadata that is produced. Specifically, exhaustive translation of a given profile into our template language guarantees compliance of the generated metadata documents with the former. Conclusions are then exposed in Section 5. Table 1 summarizes the issues faced in the present work.

**Table 1.** Summary of the issues addressed in this work.

| Issue   | Sub-Issue  |
|---|--|
| to ease authoring of constraints in profile definitions | specifying mandatory elements<br>narrowing acceptable values<br>specifying codelists                           |
| to exploit semantic resources                           | selecting attributes from semantic resources<br>checking property value consistency against semantic relations |

## 2. Related Work

SensorML [2,3] is a standardized metadata model defined by the associated XML Schemas [6] and partly based on the OGC Geographic Markup Language (GML) [7]. It provides a schema that supports all details of sensors and sensor-to-platform constellations.

Authoring SensorML descriptions is typically considered tedious and error-prone. As mentioned above, this is partially due to the scarcity of authoring tools expressively tailored to SensorML. The SWE Software webpage ([http://www.ogcnetwork.net/SWE\\_Software](http://www.ogcnetwork.net/SWE_Software)) on the OGC website provides some links to SensorML software (see Appendix B for the reference to an archived version of the page that is currently unavailable). Some of them are related to SensorML composition purposes, but just two of these are SensorML editors: The Pines SensorML Editor and the SensorML Process Editor. They are freely-available and developed in Java; unfortunately, both tools are currently not being developed and, anyway, are based on SensorML v1.0.1 (current version is 2.0). As a consequence of the scarce availability of SensorML editors, current practice often recurs to general purpose XML editing tools, which can be awkward to users unfamiliar with XML coding. Table A1 in Appendix B presents an update of the list of SensorML-composition-related software, considering more recent developments and comparing some characteristics of older and newer resources. The up-to-date census counts

four new editors: the OpenSensorHub SensorML editor, the SensorNanny draw my observatories [8], ISTSOS, and GET-IT EDI.

Beside the lack of dedicated software, other issues have been highlighted in the literature as possible limitations to widespread adoption of SensorML, even by experienced users. Firstly, SensorML is by design an extremely flexible schema, allowing for representation of a very broad range of sensors, but at the same time eliciting multiple descriptions for the same sensor [4,9,10]. This is due to the intrinsic characteristics of SensorML design. In fact, on the one hand, the schema is not prescriptive, it mandates a very narrow set of XML elements, and it does not constrain admissible XML structures sufficiently [11]: One could have the same information modeled in two distinct, both well-formed and valid, XML trees [12] (see an example in Figure 1).



Figure 1. Example of two different SensorML chunks modeling the same information.

On the other hand, flexibility in SensorML design is augmented by “soft-typed” and “linkable” properties. *Soft-typing* is a technique that lets properties to be defined by generic XML elements, and to further disambiguate their semantics by the value of a specific attribute (e.g., attributes *name* or *code*). Listing 1 shows an example of soft-typing, taken from the SensorML 1.0.1 specification [2]. Here, the semantics of the inner element *Quantity* is provided by the value of property *name* in the outer element *Component*, which is *focalLength*.

Listing 1. An example of soft-typing.

```

1 <component name="focalLength">
2   <Quantity id="fov" uom="urn:ogc:def:units:ogc:1.0:mm">0.1</Quantity>
3 </component>

```

*Linkable properties* allow specifying selected elements by referring to either XML code within the document itself or via an external link. As an example, Listing 2 shows a *swe:field* element specified inline (Lines 1–3), by reference to an anchor internal to the document (Line 5), or using a Uniform Resource Locator (URL) pointing to a fragment of an external document (Line 7).

Listing 2. An example of linkable property.

```

1 <swe:field>
2   <swe:Quantity id="TEMP" ... />
3 </swe:field>
4
5 <swe:field xlink:href="#TEMP"/>
6
7 <swe:field xlink:href="http://www.my.com/fields.xml#TEMP"/>

```

While these characteristics are key enablers of semantic interoperability and universal understanding of metadata, at the same time they allow for both misuses (e.g., when linking nonexistent or inappropriate sources) and metadata heterogeneity among communities.

### 2.1. SensorML Profiles

As previously said, in order to appropriately manage the aforementioned concerns, the EO community recurs to SensorML *profiles* as a solution for encoding-level interoperability [10,13–15].

Different interpretations for term “profile” exist. An insightful discussion on this topic can be found on the Library of Congress (LOC) website [16], according to which only some SensorML profiles mentioned in the literature should be considered International Standardized Profiles (ISP) (see also [17]), that is, “An Internationally agreed to, harmonized document which identifies a standard or group of standards, together with options and parameters, necessary to accomplish a function or set of functions”. Instead, other SensorML profiles could better correspond to the weaker notion of profile that the LOC puts forward, which comprises best practices that are recommended by research communities.

Profiles identify subsets of the valid instance documents for the schema they refer to by imposing additional constraints and thus providing less flexibility and a narrower set of compliant documents. Typical constraints require a given element:

1. to be mandatory (structure constraint) [15];
2. to feature a narrower set of acceptable values for the attributes amenable to soft-typing;
3. to exploit specific code-lists for parameter values (restricting linkable properties).

Profiles may be associated with tools, such as validators, typically implemented in Schematron [18] or in RelaxNG [19] as in [14], that let users check some of the defined constraints. Profile definition does not imply provision of this kind of tools for checking conformance of documents: When constraint validation is not supported, documents must be checked manually, following the textual profile documentation. SensorML profiles have been developed to favor sensor discovery [14] (mandating for instance the fields devoted to sensor identification) and to ease production of SensorML documents [15]. The documentation of these profiles expresses constraints in form of Schematron rules, thus providing a way for automatic validation of document compliance.

The OGC best practice for hydrological data is an extreme case of SWE profile by means of which it is possible to exemplify the extent to which SensorML complexity has been hampering the choice of SWE for sensor network development:

To foster adoption of the SWE Sensor Observation Service (SOS) standard [20] and to overcome other issues, the OGC Sensor Observation Service 2.0 Hydrology Profile [21] regards provision of SensorML descriptions as an optional feature. Even though SensorML is a key component in SOS, the Hydrology Profile requires expressing sensor *types* only, without mandating any information on the deployed physical sensor instances [21].

#### 2.1.1. SensorML Templates

A different, intermediate tool for facilitating profile adoption is constituted by SensorML examples, also named SensorML templates. This practice can be considered as a preliminary step to actual profile definition, but it is often the only tool a short-term community can afford.

Archives of SensorML templates can be the reference for making a certain SensorML encoding practice emerge. An example of this kind of archive is available online for what concerns the notable framework of the SWE Bridge described by Martínez et al. [22]: The Sensor Deployment Files repository archives SensorML documents from the EMSODEV (<http://www.emsodev.eu/>) , INTMARSIS (<https://www.sarti.webs.upc.edu/Intmarsis/>), and NeXoS (<http://www.nexosproject.eu/>) projects. These examples can be read by the SWE Bridge software so that the systems they refer to can be integrated in the proposed framework, enabling their automatic configuration in a “plug-in” fashion.

SensorML templates could be merely intended as examples to follow, but, in this paper, we use term “template” to indicate something more specific, i.e., pre-compiled structures of documents on the basis of which actual metadata can be produced. We report efforts to systematize such kind of technique via specific software editors (e.g., by using Smle by 52° North) and via RelaxNG documents defining rules to produce SensorML documents with an exact structure (see examples in the SeaDataNet project documentation [23]). In section 3, we describe our system for creation of such templates.

### 2.1.2. Usage of Semantic Resources within the SensorML and SWE Standards

References to Semantic Web concepts such as Resource Description Framework (RDF) vocabularies are, as highlighted in the Introduction, deeply woven into SWE standards. We can find examples in actual SensorML profiles. In [24], the electronic business Registry Information Model (eBRIM) SensorML profile is proposed, formulated as Schematron constraints (see Appendices A.1 and A.2 for some examples). In the “Issues” section of the document, when considering attribution of semantics by referencing external resources in SensorML (e.g., for soft-typing), it is recognized that “An important functionality would be an operation providing access to phenomenon definitions (e.g., resolving the URIs) but also for exploiting semantic relationships (e.g., finding equivalent or similar definitions)”. Then, it references the proposal in the Sensor Observable Registry Discussion Paper [25] where an ontology repository with reasoning and querying capabilities is sketched. The SeaDataNet SensorML and Observations and Measurements (O&M) [26,27] profiles for Research Vessels is described by the enlightening report in [23]. In the section devoted to “SWE Common”, it is highlighted that “the SWE Common Data Model is to define and package sensor related data in a self-describing and semantically enabled way. The main objective is to achieve interoperability, first at the syntactic and later at the semantic level (by using ontologies and probably semantic mediation)”. SeaDataNet defined its own vocabularies as Simple Knowledge Organization System (SKOS) thesauri [28] and the report indicates how their terms should be used within elements of SensorML and O&M documents. Note that O&M, as well as SensorML and other specifications, is part of the OGC Sensor Web Enablement initiative (see also <http://www.opengeospatial.org/ogc/markets-technologies/swe> for further references).

By the end of 2015, a discussion group had been started in the marine observation community for defining a novel profile for marine sensors [29] (the discussion group mailing list is available at <https://list.52north.org/mailman/listinfo/marine-swe-profiles> and a public repository is available at <https://odip.github.io/MarineProfilesForSWE/>). The initiative, promoted by Simon Jirka of 52° North, involves researchers and data managers from three continents (Europe, North America, and Oceania), several international and intercontinental projects, namely ODIP ([www.odip.eu](http://www.odip.eu)), SeaDataNet ([www.seadatanet.org](http://www.seadatanet.org)), FixO3 ([www.fixo3.eu](http://www.fixo3.eu)), and EuroFleets ([www.eurofleets.eu](http://www.eurofleets.eu)), and some research platforms, such as IOOS ([ioos.noaa.gov](http://ioos.noaa.gov)) and IMOS ([imos.org.au](http://imos.org.au)). A key topic in this discussion is the adoption of controlled vocabularies, thesauri, and ontologies for metadata harmonization. This follows the consolidated interoperability strategy developed by the British Oceanographic Data Centre (BODC) [30], whose semantic resources, accessible via the NERC Vocabulary Server (NVS) [31], have been exploited by the marine community for years [32] both as Linked Data and through the NVS SPARQL endpoint. Similarly, the Marine Metadata Interoperability (MMI) Ontology Registry and Repository [33] hosts several third-party marine ontologies and vocabularies that are exposed to the marine community through semantic services. With respect to SensorML, choosing this strategy will impact both on soft-typing and on linkable properties. What is emerging here (see [34]) as well as in similar proposals (e.g., in the Alfred Wegener Institute SensorML profile described in [35]) is a SensorML profile requiring definitions of soft-typed elements to be available as semantic resources by given authoritative endpoints. Apparently, this category of profiles is not completely definable by means of current technological practices (XML Schema, Schematron, and RelaxNG). In fact, these technologies are not able to check all the aforementioned constraints: They do not support, for instance, validating the restrictions on ranges of linkable properties when they are defined by third-party authoritative sources.



### 3. A Framework for Computer-Aided Management of Sensor Metadata

In this section, we introduce the solutions we developed to overcome the difficulties discussed in the previous sections. Specifically, we discuss a novel template-driven metadata editor, EDI, which supports profile definition in the perspective of metadata production.

#### 3.1. An Editor for Sensor Profiles

Our solution for metadata editing has been developed in the context of RITMARE (<http://www.ritmare.it>), a Flagship Project by the Italian Ministry of University and Research. The challenge we tackled was providing a web environment to easily and directly manage, publish, and share environmental data and metadata with no specific technical skills. We needed to ease provision of metadata for datasets according to the Italian profile to INSPIRE metadata (Infrastructure for Spatial Information in Europe, see <http://inspire.ec.europa.eu>) and for sensors (as SensorML descriptions). Thus, we developed an open-source, user-friendly editor, EDI [36–39], capable of abstracting from the specific metadata schema and, in particular, from a specific profile a given metadata format is complying with.

The architecture of the EDI editor, and its inputs, outputs, and relations with profile definitions are depicted in Figure 2. The technologies involved in each block are specified by vertical labels at its sides. The arrows represent communication among its distributed components via the HTTP protocol (solid lines), actions (dashed lines), and relations (dotted lines).

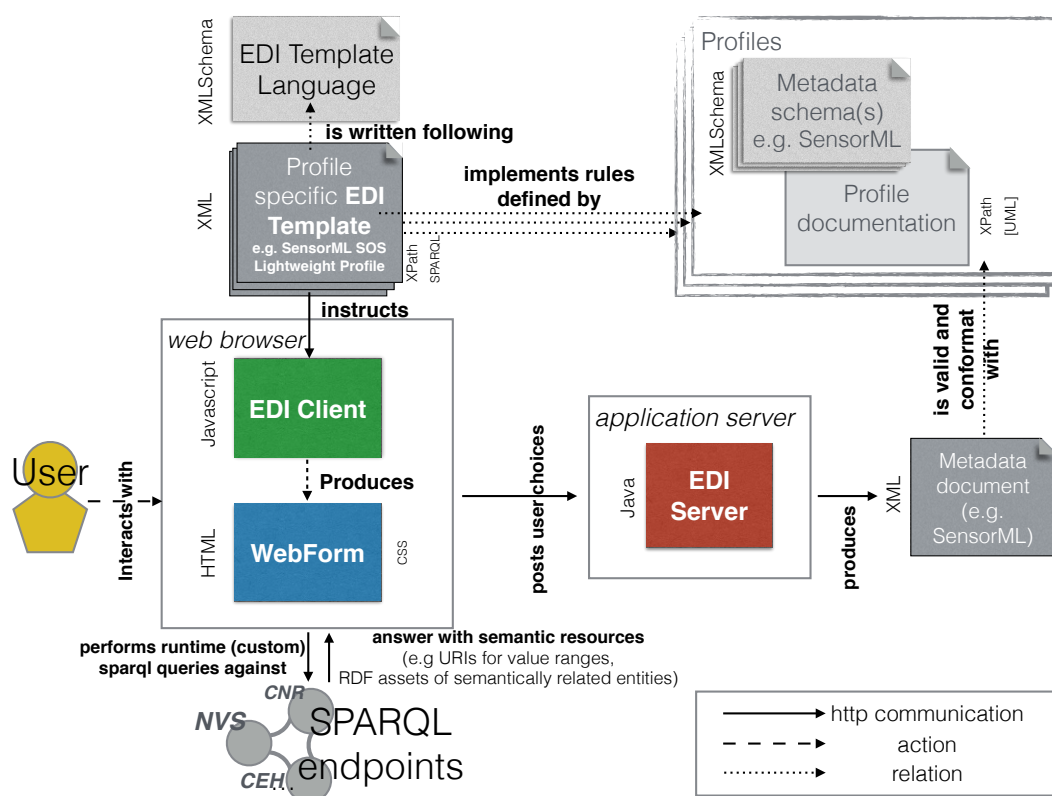


Figure 2. EDI architecture.

Profiles are represented in the top-right part of the figure. Albeit, strictly speaking, not being components of the system, the figure depicts their relations with EDI, clarifying the mutual roles of the profiles and the editor. Please note the multiplicity of arrows among the dark grey box set labeled “EDI templates” and the set of boxes representing profile definitions. In fact, a central role in the system is played by its variable, customizable part, i.e., the *EDI template* (the *template* in the following).

A template is an XML document that corresponds to a profile and that instructs the *EDI Client* on the web interface to be displayed. A template contains all the information needed by EDI in order to assist composition of metadata records according to the specific profile it was developed for. The *EDI template language* is defined by its own XML schema, which introduces the primitives that are necessary for the definition of metadata elements, intended as high-level abstractions of the information that shall be provided by the user. We acknowledge that there is a number of formalisms that can express the constraints defined in a profile, such as the Rule Interchange Format (RIF) [40] or the Semantic Web Rule Language (SWRL) [41]. However, our solution combines a gentle learning curve (XML, *simple paths* from XPath 1.0, and SPARQL as its only ingredients) and an attitude toward user experience in the metadata editing activity. The running modules of the editor are represented by the two boxes labeled “EDI Client” (green) and “EDI Server” (red): The first is a JavaScript software running in the web browser, the second is a remote web service developed in Java and communicating with the client via the HTTP protocol. Both these software packages are available in our github repository <https://github.com/SP7-RITMARE/>. The figure depicts a user interacting with the web browser. The interaction is, more specifically, with the web form (blue box) generated and controlled at runtime by the EDI Client. Once a template is provided, the EDI Client software is able to produce the web interface, an HTML5 web form the user can interact with. When the user submits the form, the client combines user and template information and posts the whole to the *EDI Server* that is responsible of producing the final XML metadata document. As shown in the bottom-left part of Figure 2, EDI is capable of issuing customized SPARQL queries to generic endpoints by defining appropriate data sources in the template. This way, EDI acts as a glue between XML metadata and the Web of Data.

Articulating EDI as a client–server application is functional to separating concerns. As an example, importing metadata records (a feature supported, for instance, by Smle and OpenSensorHub) is not supported by EDI because it is up to a second application, described in [42], to process pre-existing XML metadata, execute semantic lift according to an EDI template, use the EDI Server as a repository for the augmented metadata, and finally provide the user with the EDI Client interface for modifying the metadata. Similarly, it is not difficult to conceive a different application consuming the input of the EDI Client in place of (or simply before) invoking the EDI Server. For instance, we are considering to generate a native RDF [43] representation of metadata in order to provide enhanced discovery functionalities more easily.

### 3.2. The EDI Template Language

The strategy underlying EDI consists of providing ex-ante support to profiles: It contrasts the ex-post validation traditionally offered for profiles, typically through Schematron or RelaxNG. At the same time EDI templates are designed to represent the profile’s document requirements exploiting common syntaxes and technologies (such as XPath, XML, and SPARQL). Hence, the transition from traditional practices to an EDI template definition is facilitated.

It is significant to stress the peculiar characteristic of EDI shown in the bottom-left part of Figure 2, that is, the capability of exploiting semantic resources through the execution of runtime SPARQL queries whose logic is defined in the EDI template. This means that, by defining an EDI template, it is possible to *represent* and *enact* the constraints in a profile definition that traditional validators cannot check, such as limiting the value range of a metadata field by executing complex queries on third-party, authoritative semantic resources.

Thus far, we already authored EDI templates for several metadata profiles, defined both by international institutions, by Community of Practices (CoP), and by specific projects. Namely, our GitHub repository features the following templates: The INSPIRE profile to ISO 19115; RNDT (Repertorio Nazionale dei Dati Territoriali—national repository of territorial data: <http://www.rndt.gov.it>), the Italian profile to the former; SensorML v1.0.1; SensorML v2.0.0 SOS lightweight profile; the NcML format for annotating NetCDF deployments on THREDDS servers; the profile of SensorML

v1.0.1 in use in the RITMARE project; the previously introduced SeaDataNet II SensorML v2 Vessels profile that is in use in the Eurofleet project.

Figure 3 shows the essential components that are defined in the XML Schema underlying EDI templates. The *element* tag in Figure 3a (term “tag” is preferred over “element” to avoid ambiguity with *element* definitions in templates) specify the structure of individual metadata fields: Its attributes provide a unique *xml:id*, declare whether it is mandatory, and specifies the associated multiplicity. Moreover, the field can substitute (i.e., be *alternativeTo*) another. The *element* tags define the multilingual visual cues to be shown in the interface (the *label* and *help* tags) and the XPath location where instances of the metadata field shall be rooted (the *hasRoot* tag). The set of *items* contained in tag *produces* represent the individual XML nodes that shall be created for a given metadata field. Tag *rdfOut* has been introduced to drive creation of the RDF representation of the metadata field and, conversely, tag *rdfIn* drives extraction of metadata field values from a triple store. However, usage of these two fields is out of the scope of this paper.

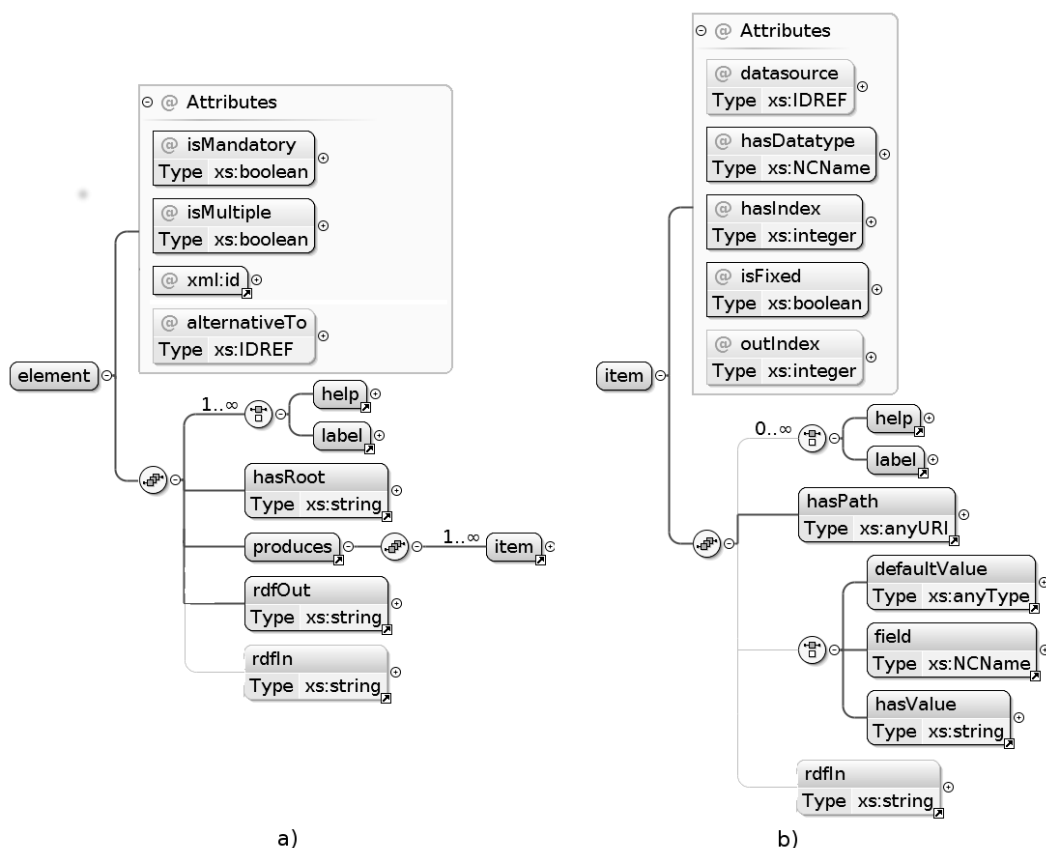


Figure 3. The *element* (a) and *item* (b) XML Schema components.

As mentioned above, *element* definitions can trigger the creation of multiple XML elements and attributes in the target metadata file: This is the purpose of the *item* tag (Figure 3b), whose semantics is the following: Attributes *hasIndex* and *outIndex* control the ordering of fields, respectively, in the interface and in the output document. Attribute *isFixed* determines whether a form widget shall be created in the editing interface (when set to “false”) or if the metadata field can be kept transparent to the end user (e.g., because its value is known in advance). Other key attributes are *hasDatatype*, specifying the range of valid values for the item, and *datasource*, specifying the link to the data source defining the admissible values for the metadata field. The tags included in individual *items* define, beside the visual cues previously found in the definition of the *element* tag, the fixed value for the field, a default one, or the corresponding variable (the *field* tag) that can be found in the projection of the SPARQL query associated with the corresponding *datasource* (more on this in the following).



The *hasPath* tag specifies the XPath of the XML node that shall be created. Finally, an optional *rdfln* tag complements the SPARQL query in the *rdfln* tag defined for the *element* as a whole. Please refer to the work of Fugazza et al. [44] for a comprehensive description of the interplay between element- and item-level *rdfln* tags.

We can now move on to the definition of data sources, one of the core feature for profile definition, depicted in Figure 4. All three different categories of data sources supported by templates share an *xml:id* attribute for unique identification and an *endpointType* attribute allowing administrators to specify additional parameters for specific endpoint types (endpoints are web addresses SPARQL queries can be posted to). The *url* tag allows for per-datasource definition of endpoints. The three categories are also described below.

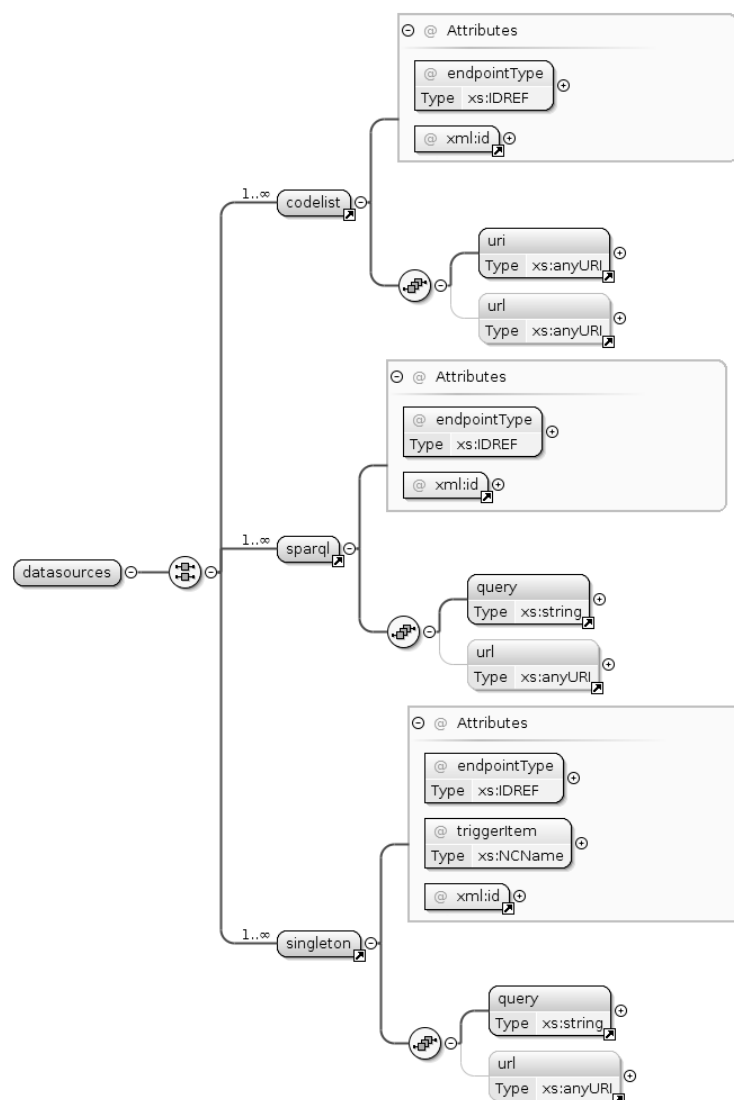


Figure 4. The *datasource* schema component.

**Codelist:** In this case, the datasource is a SKOS thesaurus (a controlled vocabulary encoded according to this specific ontology) and then the interface can issue standard queries for matching code values. Typically, this category of data sources results in a drop-down list for selecting among the codes defined by the thesaurus.

**Sparql:** This category allows for executing generic SPARQL queries, a handy functionality to interact with data sources defined according to different formalisms (such as RDFS or the Web Ontology

Language (OWL) [45,46]) and with different purposes.

**Singleton:** The queries defined for this data source are required to return a single record (typically, as a consequence of a previous call to a datasource of the preceding type).

Figure 5 shows, in the context of SensorML, the three flavors of assisted metadata editing that leverage external data sources: (a) dynamic populating of dropdown lists (typically, driven by data sources of type “codelist”); (b) autocompletion functionalities (fed by data sources of type “sparql”); and (c) conditional automatic compiling of fields based on runtime queries (the data sources of type “singleton”) that selects property values on the basis of choices made by the user for other metadata items. In the example, once the user has selected a specific person in the Friend Of A Friend (FOAF) [47] registry populating the autocompletion options provided by EDI (Figure 5b), all the related details are extracted from the RDF graph according to the logics expressed by the SPARQL query (see Figure 5c). The template code at the basis of Figure 5 is detailed in Appendix A.2, Listing A6.

Position Properties

Reference system

wgs84

2D TM projection in ETRS89 on GRS80, z...

2D TM projection in ETRS89 on GRS80, z...

2D LAEA projection in ETRS89 on GRS80

2D LCC projection in ETRS89 on GRS80

2D geodetic in ETRS89 on GRS80

Monte Mario

Monte Mario (Rome)

Monte Mario / Italy zone 1

Monte Mario / Italy zone 2

ed50

ed50/utm 32n

ed50/utm 33n

etrs89/utm-zone32N

etrs89/utm-zone33N

igm95

igm95/utm 32n

igm95/utm 33n

✓ wgs84

wgs84/3d

wgs84/utm 32n

Operator

Operator e-mail address

fotr

mailto:aragornl.elassar@fotr.org

mailto:bilbo.baggins@fotr.org

mailto:frodo.baggins@fotr.org

mailto:gandalf.thegrey@fotr.org

mailto:meriadoc.brandybuck@fotr.org

mailto:peregrin.took@fotr.org

mailto:samwise.gamgee@fotr.org

Operator Postal code

Operator country

Operator web site

Operator

Operator e-mail address

mailto:bilbo.baggins@fotr.org

Operator name

Thorin and Company

Operator phone number

+00111111111

Operator address

Bagshot Row

Operator city

Hobbiton

Operator Postal code

000000

Operator country

The Shire

Operator web site

www.get-it.it

(a)

(b)

(c)

Figure 5. Items connected to datasources of type: codelist (a); sparql (b); and singleton (c).

Thus far, the EDI software has been successfully exploited by several communities In fact, EDI enabled publication of previously inexistent SensorML metadata documents, thus fostering effective sensor data sharing through Sensor Observation Services. The metadata catalogs (nodes) activated thanks to EDI are reported in Table 2. When possible, in the table, we show the comparison between the number of sensors deployed during the project’s life (as reported in [48,49]) and the number of sensor currently hosted by the same nodes, attesting that several adopters continue to successfully deploy sensors with the help of EDI (see also [50,51]), also outside the project’s framework and commitment.

**Table 2.** Nodes enabled by the EDI tool

| Node  | URL   | CoP/Domain  | Project  | Number of SensorML | Number of SensorML Created during Project RITMARE according to Census Report (2018) | Status                              |
|---|---|---|--|--------------------|---|-------------------------------------|
| CNR-ISMAR<br>(Institute of Marine Science)  | <a href="http://vesk.ve.ismar.cnr.it/sensors/">http://vesk.ve.ismar.cnr.it/sensors/</a>       | marine science,<br>lagoon ecosystem<br>research                   | RITMARE  | 35                 | 25  | active                              |
| IRSA CNR<br>(Institute for research on waters)  | <a href="http://sk.ise.cnr.it/sensors/">http://sk.ise.cnr.it/sensors/</a>                     | water and land<br>ecosystems,<br>aquatic ecology,<br>inner waters |  | 23                 | -   | active                              |
| NextData  | <a href="http://nextdata.get-it.it/sensors/">http://nextdata.get-it.it/sensors/</a>           | mountain and water<br>ecology,<br>inland waters                   | NextData project<br>( <a href="http://www.nextdatapoint.it/">www.nextdatapoint.it/</a> ) | 27                 | -   | active                              |
| LTER Italia<br>(Long Term Ecosystem Research)   | <a href="http://getit.lteritalia.it/sensors/">http://getit.lteritalia.it/sensors/</a>         | Long Term<br>Ecosystem Research                                   |  | 16                 | -   | active                              |
| eLTER project   | <a href="http://cdn.lter-europe.net/">http://cdn.lter-europe.net/</a>                         | Long Term<br>Ecosystem Research                                   | eLTER<br>( <a href="http://www.lter-europe.net">www.lter-europe.net</a> )                | 10                 | -   | offline                             |
| ICPSM<br>(centro previsioni e segnalazioni maree—<br>Comune di Venezia)                                       | <a href="http://icpsm.get-it.it/observations/sos">http://icpsm.get-it.it/observations/sos</a> | civil protection -<br>flood forecast                              | RITMARE  | 50 (approximately) | 13  | pilot project,<br>currently offline |
| CNR-IAS<br>(Institute for studies on<br>anthropic impacts<br>and sustainability<br>in the marine environment) | <a href="http://sk.oristano.iamc.cnr.it/sensors/">http://sk.oristano.iamc.cnr.it/sensors/</a> | Anthropic impact<br>in the marine<br>environment                  | RITMARE  | 3                  | 0   | active                              |

#### 4. Discussion: Towards Definition of Semantically Consistent Profiles

In Section 2.1, we discuss profiles defining the category of constraints in the second point of the item list, that is, “to feature a narrower set of acceptable values for attributes amenable to soft-typing”. Existing profiles already sport a refined notion of this category that is strongly linked to semantics, which could be formalized and generalized as:

- R1. To feature a narrower set of acceptable values for attributes amenable to soft-typing, and *selecting them from a specific semantic provider*.

Of the metadata editing tools reviewed in this paper, only EDI and the Open Sensor Hub SensorML editor (OSH) [52] can deal with this category of requirements. OSH features limited functionality because it allows to lookup semantic resources only according to a predefined SPARQL query, being the query hard-coded in the java software (see <https://github.com/opensensorhub/sensorml>). Moreover, in OSH there are only two fixed predefined SPARQL endpoints to choose from: The MMI Ontology Registry, and the XDOMES project endpoint (<https://xdomes.org>). Instead, EDI allows for formalization both of the endpoint against which the run-time query is performed and of the SPARQL query itself. This is achieved by profile developers customizing EDI templates. In Appendix A.3, a working example of the “sensor manufacturer information consistency” use case is provided as EDI template structures.

In addition, in [53] the authors described a possible scenario for semantics in the sensor web. In their work, they announced the development not only of semantic definitions within SWE technologies, but an entire technological infrastructure enabling reasoning and reconciliation of terminologies. They also sketched the case of a semantically-driven, intelligent sensor plug-and-play based on consistency of sensor information. As an example, they considered the issue of matching a sensor output with the property of a Feature of Interest (FoI). This reprises an unsolved aspect in the O&M Standard, whose abstract specification [27] states that FoI types must carry the observed property, while the same standard implementation (in XML Schema) does not model this constraint [26].

We think that the argument should be further stressed, stating that a future kind of profiles should be able to mandate a novel set of semantically driven constraints:

- R2. To require consistency among the property values of sets of elements with respect to specific semantic relations.

Our proposal in this direction, as described in Section 3, suggests that SPARQL queries could be the semantic glue to enforce consistency among metadata elements.

In the aforementioned case (that is, the issue “FoI types must carry the observed property”), such constraint could require that, in a given ontology, the observed Feature of Interest is of a type compatible with the declared observed property. For instance: If the FoI were the superficial part of a given glacier, whose type is “glacier top”, then the ontology should check if the observed property “temperature” is applicable to this kind of feature or not. Listing 3 shows one of the possible implementations of this constraint: Specifically, a query is performed against the RDF version of the Community Surface Dynamics Modeling Systems (CSDMS) [54] (a semantic resource, hosted in the MMI repository, containing the cross-domain naming conventions for describing process models, datasets, and their associated variables). The CSDMS standard names associate objects and quantities from different domains, and could be one of the possible interesting sources to take into consideration for this kind of semantic consistency checks.

The meaning of the query result is, in our scenario, that the object of type “glacier top” carries the quantity “temperature”. Unfortunately, currently EDI does not fully support R2: For enacting this constraint, EDI should propose the user the semantically consistent quantities on the basis of what is selected elsewhere in the metadata.

**Listing 3.** An example of semantic consistency check via a SPARQL query against [mmisw.org/ont/sparql](http://mmisw.org/ont/sparql).

```

1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX ts: <http://www.isi.edu/ikcap/geosoft/ontology/csdms/#>
4 PREFIX tso: <http://www.isi.edu/ikcap/geosoft/ontology/software.owl#>
5 SELECT ?s ?label_of_o ?label_of_q WHERE {
6   BIND(ts:object_glacier_top as ?o)
7   ?s tso:hasObject ?o .
8   ?o a tso:Object;
9     rdfs:label ?label_of_o .
10  ?s tso:hasQuantity ?q .
11  ?q a tso:Quantity;
12     rdfs:label ?label_of_q
13 }
14
15 --> returns: ts:glacier_top_temperature 'glacier_top' 'temperature'
```

Another constraint of type R2 is the following use case on “sensor manufacturer information consistency”, requiring that the attributes describing this entity are consistent with a given directory, such as a FOAF graph of sensor manufacturers (e.g., [55]). This use case is isomorphic to the one depicted in Figure 5b,c.

The EDI design, as mentioned, enables ex-ante validation because only consistent values are proposed for metadata field autocompletion; moreover, upon choosing one of the options, dependent fields can be automatically filled in. In the use case presented, as soon as the user chooses manufacturer X while editing a SensorML description, the SPARQL result containing all pertinent information (e.g., the country) is used to automatically fill in the appropriate SensorML elements. In the example, the EDI template specifies the endpoint of project RITMARE, and the query is directed towards the FOAF graph of sensor manufacturers developed for the project. This second example relates to point R2, and it is an example of R2 cases supported by EDI. It moreover shows how the definition of constraints of type R1 and R2 could ease the work of metadata providers, once appropriate tools would be offered to a profile’s community: on the one hand, suggesting to them the appropriate lists of acceptable values in order to foster homogeneous but eventually evolving terminologies in a given community and, on the other hand, guiding them through semantically consistent choices.

## 5. Conclusions

In this paper, we propose our solution to the issues hampering widespread adoption of SensorML as a sensor description metadata language in the EO community. Specifically, we developed a template-driven, form-based editing tool, EDI, that is capable of easing editing of records in any XML-based schema. EDI is made available as free and open source software. By itself, such tool allows for a number of user-oriented functionalities that speed up metadata editing and assure compliance with the specific schema that is considered. The tool can be used to formalize specific profiles for a given schema, a feature that is advisable for SensorML descriptions of specific categories of sensors. Moreover, EDI can exploit semantic resources, to make them easily available to providers during metadata compilation. Consequently, we built a number of sensor metadata profile templates that produce profile-compliant and semantically consistent SensorML. This also relieves developers of the burden of providing redundant information about a specific sensor, thus allowing them to focus on the aspects concerning the specific deployment. More importantly, these templates are capable of formalizing the eventually evolving information characterizing a given CoP, such as the reference terminology, drawing it from RDF data sources in the Web of Data. In this way, EDI helps avoid as much as possible mistakes and favors the most appropriate usage of terminology within a specific domain. It is impossible for state-of-the-art practices for SensorML profile definition to achieve this. The solution presented produced an increase in quality and quantity of the available sensor descriptions in the RITMARE community, encompassing most of Italian marine research and representing most of



Italian observation systems in this domain. For this reason, the metadata templates created by now are mainly related to sensors in the marine environment. As any software, EDI, the tool proposed in this paper for encoding of metadata profiles, features its own shortcomings. Beside the issue described in the preceding section, the only two issues we identified thus far are related to articulation of XML nodes in the output metadata and the number of SPARQL endpoints that can be accessed by a given data source. The former can be summarized as follows: EDI cannot easily handle creation of nodes that can be arbitrarily nested into nodes of the same type. As an example of this, consider the SensorML *event* tag: This tag can be nested (e.g., think of dependent calibration events) but execution of EDI can only arrange them as siblings in the metadata document. The second issue is related to the univocal specification of the target SPARQL endpoint in data source definitions. Future work will be devoted to solving these limitations.

**Author Contributions:** Conceptualization, Paolo Tagliolato and Cristiano Fugazza; Data curation, Paolo Tagliolato, Cristiano Fugazza and Alessandro Oggioni. Funding acquisition, Alessandro Oggioni and Paola Carrara; Methodology, Paolo Tagliolato; Software, Paolo Tagliolato, Cristiano Fugazza and Alessandro Oggioni; Supervision, Paola Carrara; Writing—original draft, Paolo Tagliolato and Cristiano Fugazza; and Writing—review and editing, Paolo Tagliolato, Cristiano Fugazza, Alessandro Oggioni and Paola Carrara.

**Funding:** This research was funded by MIUR Italian flagship project RITMARE, by H2020 grant number 654359 (project eLTER). Publication costs were partly covered using residual CNR funds from project H2020 grant number 654310 (project ODIP II).

**Acknowledgments:** We would thank for the fruitful exchange of ideas about SWE and SensorML profiles Jordi Sorribas, Raquel Casas (CSIC) and all the ODIP II prototype 3 people. We gratefully acknowledge Martina Zilioli for providing us with documentation and datasets reporting the census of project RITMARE resources.

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

## Abbreviations

The following abbreviations are used in this manuscript:

|          |  |
|----------|--|
| CoP      | Community of Practices                         |
| EO       | Earth Observation                              |
| ebRIM    | Electronic Business Registry Information Model |
| FOAF     | Friend of A Friend                             |
| GML      | Geographic Markup Language                     |
| LOC      | Library of Congress                            |
| ISP      | International Standardized Profile             |
| OGC      | Open Geospatial Consortium                     |
| OWL      | Web Ontology Language                          |
| O&M      | Observations and Measurements                  |
| RDF      | Resource Description Framework                 |
| RDFS     | Resource Description Framework Schema          |
| RIF      | Rule Interchange Format                        |
| SOS      | Sensor Observation Service                     |
| SKOS     | Simple Knowledge Organization System           |
| SPARQL   | SPARQL Protocol and RDF Query Language         |
| SWE      | Sensor Web Enablement                          |
| SWRL     | Semantic Web Rule Language                     |
| SensorML | Sensor Model Language                          |
| URI      | Uniform Resource Identifier                    |
| URL      | Uniform Resource Locator                       |
| XML      | eXtensible Markup Language                     |
| XPath    | XML Path Language                              |

## Appendix A

Example of SensorML profiling (from [24]).

### Appendix A.1. Soft-Typing

**Listing A1.** An example of soft-typing (from the SensorML ebRIM profile).

```

1 <identifier name="longName">
2   <Term definition="urn:ogc:def:identifier:OGC:1.0:longName">
3     <value>OSIRIS Thermometer 123</value>
4   </Term>
5 </identifier>

```

**Listing A2.** Schematron rule example constraining soft-typing to specific URIs (from the SensorML ebRIM profile).

```

1 <rule context="//sml:identification">
2   <assert test="count(sml:IdentifierList/sml:identifier/sml:Term[@definition =
3     'urn:ogc:def:identifier:OGC:longName']) = 1" >Error: one identifier has to
4     be of the type 'urn:ogc:def:identifier:OGC:longName'.
5   </assert>
6 </rule>

```

**Listing A3.** Excerpt of EDI template enacting the constraint on soft-typing.

```

1 <element xml:id="long_name" isMandatory="true" isMultiple="false">
2   <label xml:lang="en">Long Name</label>
3   <help xml:lang="en">Long name of the system or platform.</help>
4   <hasRoot>/sml:PhysicalSystem/sml:identification/sml:IdentifierList</hasRoot>
5   <produces>
6     <item hasIndex="1" xml:id="long_name_1" isFixed="true" hasDatatype="string">
7       <hasPath>sml:identifier/sml:Term/@definition</hasPath>
8       <hasValue>urn:ogc:def:identifier:OGC:longName</hasValue>
9     </item>
10    <item hasIndex="2" xml:id="long_name_2" isFixed="false" hasDatatype="string">
11      <hasPath>sml:identifier/sml:Term/sml:value</hasPath>
12    </item>
13  </produces>
14 </element>

```

**Figure A1.** EDI web form rendering of the excerpt in Listing A3.

### Appendix A.2. Requiring the Presence of a Specific Structure in SensorML Documents

Example requiring specific fields for contacts (from the profile document in [24]). Specifically, The full constraint is: Ensure that, for every System and Component, a contact element is provided and it contains at least an organisationName element.

**Listing A4.** Example of required SensorML contact section.

```

1 <contact>
2   <ResponsibleParty gml:id="WWU.IfGI.thermometer.contact">
3     <individualName>Simon Jirka</individualName>
4     <organizationName>Institute for geoinformatics, Universitaet Muenster</organizationName>
5     <contactInfo>
6       <address>
7         <deliveryPoint>Weseler Strasse</deliveryPoint>
8         <city>Muenster</city>
9         <postalCode>48151</postalCode>
10        <country>Germany</country>
11        <electronicMailAddress>swslifgi(at)listserv.unimuenster.de</electronicMailAddress>
12      </address>
13    </contactInfo>
14  </ResponsibleParty>
15 </contact>

```

**Listing A5.** Schematron rules for making specific elements mandatory.

```

1 <rule context="//sml:System">
2   <assert test="sml:contact">
3     Error: 'sml:contact' element has to be present
4   </assert>
5 </rule>
6 <rule context="//sml:contact/sml:ResponsibleParty">
7   <assert test="sml:organizationName">
8     Error: 'sml:organizationName' element has to be present
9   </assert>
10 </rule>

```

Listing A6 presents an EDI template excerpt enacting production of the SensorML chunk in Listing A4 and following the constraint in Listing A5.

**Listing A6.** EDI template excerpt enacting production of a SensorML chunk consistent with that in Listing A4.

```

1 <element xml:id="operator" isMandatory="true" isMultiple="false">
2   <label xml:lang="en">Operator</label>
3   <help xml:lang="en">Operator contact information; autocompletion
4     is performed if the operator appears in the movie The Lord of the Ring.
5   </help>
6   <hasRoot>/sml:PhysicalSystem/sml:contacts/sml:ContactList</hasRoot>
7   <produces>
8     <item hasIndex="1" xml:id="operator_1" outIndex="8"
9       isFixed="false" hasDatatype="autoCompletion"
10      datasource="operators">
11       <label xml:lang="en">Operator e-mail address</label>
12       <hasPath>sml:contact/gmd:CI_ResponsibleParty/gmd:contactInfo/
13         gmd:CI_Contact/gmd:address/gmd:CI_Address/
14         gmd:electronicMailAddress/gco:CharacterString
15       </hasPath>
16     </item>
17     <item hasIndex="2" xml:id="operator_2" outIndex="1"
18       isFixed="true" hasDatatype="string">
19       <hasPath>sml:contact/@xlink:arcrole</hasPath>
20       <hasValue>http://inspire.ec.europa.eu/metadata-codelist/
21         ResponsiblePartyRole/pointOfContact
22       </hasValue>
23     </item>
24     <item hasIndex="3" xml:id="operator_3" outIndex="2"
25       field="1" isFixed="false" hasDatatype="select"
26      datasource="info_operator">
27       <label xml:lang="en">Operator name</label>
28       <hasPath>sml:contact/gmd:CI_ResponsibleParty/gmd:organisationName/
29         gco:CharacterString
30       </hasPath>
31     </item>
32     ...
33   </produces>
34 </element>

```

### Appendix A.3. Enforce and Enact Semantic Consistency

In the next listings, we report a more complete example where a SensorML structure similar to that in Listing A4 (expressing a contact) is enforced (Listing A7) and whose semantic consistency is enacted by the SPARQL queries in Listing A8. The required structure concerns the information about the the Manufacturer of a Physical System.

**Listing A7.** Constraining a SensorML element content model and enacting semantic consistency via SPARQL: SensorML structure.

```

1 <element xml:id="manufacturer" isMandatory="false" isMultiple="false">
2   <label xml:lang="en">Manufacturer</label>
3   <help xml:lang="en">Manufacturer contact information;
4     autocompletion is performed if the manufacturer of the system
5     (sensor or platform) is present in our repository
6     (in this case has been already recognized in
7     the field "Manufacturer Name" in the
8     "Identification of the system" section).
9 </help>
10 <hasRoot>/sml:PhysicalSystem/sml:contacts/sml:ContactList/sml:contact</hasRoot>
11 <produces>
12   <item hasIndex="1" xml:id="manufacturer_1" outIndex="1"
13     isFixed="true" hasDatatype="string">
14     <hasPath>@xlink:arcrole</hasPath>
15     <hasValue>http://mmisw.org/ont/ioos/definition/manufacturerName</hasValue>
16   </item>
17   <item hasIndex="2" xml:id="manufacturer_2" outIndex="2"
18     field="1" isFixed="false" hasDatatype="select"
19     datasource="manufacturers">
20     <label xml:lang="en">Manufacturer name</label>
21     <hasPath>gmd:CI_ResponsibleParty/gmd:organisationName/
22       gco:CharacterString
23     </hasPath>
24   </item>
25   <item hasIndex="3" xml:id="manufacturer_3" outIndex="3"
26     field="phone" isFixed="false" hasDatatype="select"
27     datasource="info_manufacturers">
28     <label xml:lang="en">Manufacturer phone number</label>
29     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
30       gmd:CI_Contact/gmd:phone/gmd:CI_Telephone/
31       gmd:voice/gco:CharacterString
32     </hasPath>
33   </item>
34   <item hasIndex="4" xml:id="manufacturer_4" outIndex="4"
35     field="addr" isFixed="false" hasDatatype="select"
36     datasource="info_manufacturers">
37     <label xml:lang="en">Manufacturer address</label>
38     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
39       gmd:CI_Contact/gmd:address/gmd:CI_Address/
40       gmd:deliveryPoint/gco:CharacterString
41     </hasPath>
42   </item>
43   <item hasIndex="5" xml:id="manufacturer_5" outIndex="5"
44     field="town" isFixed="false" hasDatatype="select"
45     datasource="info_manufacturers">
46     <label xml:lang="en">Manufacturer city</label>
47     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
48       gmd:CI_Contact/gmd:address/gmd:CI_Address/
49       gmd:city/gco:CharacterString
50     </hasPath>
51   </item>
52   <item hasIndex="6" xml:id="manufacturer_6" outIndex="6"
53     field="region" isFixed="false" hasDatatype="select"
54     datasource="info_manufacturers">
55     <label xml:lang="en">Manufacturer administrative area</label>
56     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
57       gmd:CI_Contact/gmd:address/gmd:CI_Address/
58       gmd:administrativeArea/gco:CharacterString
59     </hasPath>
60   </item>

```

## Listing A7. Cont.

```

61 <item hasIndex="7" xml:id="manufacturer_7" outIndex="7"
62     field="postcode" isFixed="false" hasDatatype="select"
63     datasource="info_manufacturers">
64     <label xml:lang="en">Manufacturer Postal code</label>
65     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
66         gmd:CI_Contact/gmd:address/gmd:CI_Address/
67         gmd:postalCode/gco:CharacterString
68     </hasPath>
69 </item>
70 <item hasIndex="8" xml:id="manufacturer_8" outIndex="8"
71     field="country" isFixed="false" hasDatatype="select"
72     datasource="info_manufacturers">
73     <label xml:lang="en">Manufacturer country</label>
74     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
75         gmd:CI_Contact/gmd:address/gmd:CI_Address/gmd:country/
76         gco:CharacterString
77     </hasPath>
78 </item>
79 <item hasIndex="9" xml:id="manufacturer_9" outIndex="9"
80     field="email" isFixed="false" hasDatatype="select"
81     datasource="info_manufacturers">
82     <label xml:lang="en">Manufacturer e-mail address</label>
83     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/
84         gmd:CI_Contact/gmd:address/gmd:CI_Address/
85         gmd:electronicMailAddress/gco:CharacterString
86     </hasPath>
87 </item>
88 <item hasIndex="10" xml:id="manufacturer_10" outIndex="10"
89     field="homepage" isFixed="false" hasDatatype="select"
90     datasource="info_manufacturers">
91     <label xml:lang="en">Manufacturer web site</label>
92     <hasPath>gmd:CI_ResponsibleParty/gmd:contactInfo/gmd:CI_Contact/
93         gmd:onlineResource/gmd:CI_OnlineResource/
94         gmd:linkage/gmd:URL
95     </hasPath>
96 </item>
97 <item hasIndex="11" xml:id="manufacturer_11" outIndex="11"
98     isFixed="true" hasDatatype="string">
99     <hasPath>gmd:CI_ResponsibleParty/gmd:role/@gco:nilReason</hasPath>
100     <hasValue>inapplicable</hasValue>
101 </item>
102 </produces>
103 </element>

```

**Identification of the system**

Manufacturer name (if you don't find the name please add new following this [link](#))

Sea

- Cetacean Research Technology
- Chelsea Technologies Group Ltd.
- DeepSea Power & Light
- IXSEA SAS
- McLane Research Laboratories, Inc.
- PREVCO Subsea Housings
- SEACON BRANTNER & ASSOCIATES, INC.
- SEATOOLS B.V.
- Sea & Sun Technology GmbH
- Sea-Bird Electronics, Inc.
- Seapoint Sensors Inc.
- Wilcoxon Research, Inc.

**Figure A2.** EDI web form rendering of autocompletion options retrieved by the first SPARQL query in Listing A8.




| Relevant Contacts   |                                     |
|---|-------------------------------------|
| <b>Manufacturer</b>  |                                     |
| <b>Manufacturer phone number</b>  | tel:+1-206-297-1310                 |
| <b>Manufacturer address</b>   | 12th Ave. NE                        |
| <b>Manufacturer city</b>  | Seattle                             |
| <b>Manufacturer administrative area</b>   | WA                                  |
| <b>Manufacturer Postal code</b>   | 98105                               |
| <b>Manufacturer country</b>   | US                                  |
| <b>Manufacturer e-mail address</b>  | mailto:crtinfo@cetaceanresearch.com |
| <b>Manufacturer web site</b>  | http://www.cetaceanresearch.com     |

Figure A3. EDI web form rendering of the constraints in Listings A7 and A8.

**Listing A8.** Constraining a SensorML element content model and enacting semantic consistency via SPARQL: Queries for value ranges.

```

1 <singleton xml:id="manufacturers" endpointType="virtuoso">
2   <query><![CDATA[
3     PREFIX ns: <http://www.w3.org/2006/vcard/ns#>
4     SELECT ?c ?l
5     FROM <http://ritmare.it/rdfdata/manufacturers>
6     WHERE {
7       ?c rdf:type foaf:Organization.
8       ?c foaf:name ?l.
9       FILTER( REGEX( STR(?l), "$search_param", "i") )
10    }
11    ORDER BY ASC(?l)
12  ]]></query>
13 </singleton>
14 <singleton xml:id="info_manufacturers" endpointType="virtuoso" triggerItem="manuf_name_3_uri">
15   <query><![CDATA[
16     PREFIX addr: <http://wymiwyg.org/ontologies/foaf/postaddress#>
17     PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
18     SELECT ?phone fn:concat(?l1, ' ', ' ', ?l2) AS ?addr ?town ?region ?postcode ?country ?email ?
19     homepage
20     FROM <http://ritmare.it/rdfdata/manufacturers>
21     WHERE {
22       OPTIONAL {<$search_param> foaf:phone ?phone .}
23       OPTIONAL {<$search_param> vcard:email ?email .}
24       OPTIONAL {<$search_param> foaf:homepage ?homepage .}
25       OPTIONAL {<$search_param> addr:address ?b1 .}
26       OPTIONAL {?b1 addr:deliveryPoint ?b2 .}
27       OPTIONAL {?b2 addr:location ?b3 .}
28       OPTIONAL {?b3 addr:thoroughfareName ?l1 .}
29       OPTIONAL {?b3 addr:streetNr ?l2 .}
30       OPTIONAL {?b3 addr:town ?town .}
31       OPTIONAL {?b3 addr:region ?region .}
32       OPTIONAL {?b3 addr:postcode ?postcode .}
33       OPTIONAL {?b3 addr:country ?country .}
34     }
35   ]]></query>
36 </singleton>

```

## Appendix B

### Appendix B.1. SensorML Editors

Table A1 compares several characteristics of the available SensorML open-source editors and can be intended as an extension to the OGC web pages on SensorML-related software ([http://www.ogcnetwork.net/SWE\\_Software](http://www.ogcnetwork.net/SWE_Software) – currently unavailable: see [https://web.archive.org/web/20180306125329/http://www.ogcnetwork.net/SWE\\_Software](https://web.archive.org/web/20180306125329/http://www.ogcnetwork.net/SWE_Software)). The table updates the one in the ODIP II project deliverable D3.3 (<http://www.odip.eu/media/odip/org/documents/d3-3-progress-prototypes-1-submitted.pdf>).

We considered the information currently available on the web for the following software:

- GET-IT EDI: <http://www.get-it.it/> (website), [https://github.com/SP7-Ritmare/EDI-NG\\_client](https://github.com/SP7-Ritmare/EDI-NG_client) and [https://github.com/SP7-Ritmare/EDI-NG\\_templates](https://github.com/SP7-Ritmare/EDI-NG_templates) (software);
- the SensorML Process Editor (recently named SensorML Library): <http://code.google.com/p/sensorml-data-processing/> (software—deprecated), <https://github.com/sensiasoft/lib-sensorml> (software—current);
- the SensorML Schema Browser: [http://www.botts-inc.net/SensorMLv1.0\\_Browser.html](http://www.botts-inc.net/SensorMLv1.0_Browser.html) (website);
- the SensorML Profile Library : <http://code.google.com/p/sensorml-profile-library/> (software);
- the Pines SensorML Editor: <http://lxspine.googlepages.com/pine%27ssensormleditor> (website and software);
- the SensorML Editor: <https://code.google.com/archive/p/sensorml-tools/downloads> (software);
- the OpenSensorHub SensorML editor: <https://github.com/opensensorhub/sensorml-editor> (software), <http://opensensorhub.github.io/sensorml-editor/SensorMLEditor.html> (online demo) (see also [52]);
- SensorNanny—drawmyobservatories: <https://github.com/ifremer/snanny-drawmyobservatory> (software);
- ISTSOS: <http://istsos.org/index.html> (website), <https://sourceforge.net/projects/istsos/> (software);
- 52North—Smle: <http://52north.github.io/smle> (website), <https://github.com/52North/smle> (software), <http://52north.github.io/smle/master/#/editor> (online demo), <http://www.fixo3.eu/download/Deliverables/FixO3-D2.10-FINAL.pdf> (tech report).

**Table A1.** SensorML editors: tentative enumeration and comparison of available software characteristics. Legend: “-” = non applicable; “?” = not enough information within documentation or other sources (code, examples, executable if any).

|                                   | GET-IT EDI   | SensorML Process Editor   | SensorML Schema Browser   | SensorML Profile Library   | Pines SensorML Editor   | SensorML Editor   | OpenSensorHub SensorML editor   | SensorNanny-drawmyobservatories   | ISTSOS  | 52North-smle   |
|-----------------------------------|--|---|---|--|---|---|---|---|---|--|
| reported in OGC page              | y (within comment)   | y   | y   | y  | y   | y   | n   | n   | n   | n  |
| software type                     | editor   | editor and software library   | webpages with graphical representation of a selection of SensorML   | file repository (see descr.)   | editor  | editor  | editor  | editor  | editor (embedded in SOS)  | editor   |
| dev. group                        | GET-IT, CNR IREA, CNR-ISMAR  | UAH—SensiaSoft, Botts Innovative Research ( <a href="http://www.botts-inc.com/software.html">http://www.botts-inc.com/software.html</a> ) | UAH—SensiaSoft, Botts Innovative Research inc.  | UAH - SensiaSoft, Botts Innovative Research inc.   | Xuesong Liu (Civil and Environmental Engineering, Carnegie Mellon University) | Botts innovative Research   | SensiaSoft  | IFREMER   | SUPSI   | 52North initiative   |
| description                       | template-driven metadata authoring tool that can be easily customized to any XML-based metadata format and to a specific workgroup, institute, or project. | Library for the execution of processes represented in SensorML. It is a process chain execution engine (not an editor of SensorML).       | Webpages pointing to views of SensorML schema (similar to XML representation utilities like in generic XML editors). Currently no software seems to be available. | Repository for executable SensorML process model instances, as well as RelaxNG profiles of the core SensorML schema (not an editor). | Program to explore and modify SensorML models                                 | This product is used to create and edit SensorML instances. It may be downloaded as a standalone eclipse application. | A web based viewer/editor to create your SensorML document. This SensorML viewer/editor is used by OSH but can also be used as a standalone web editor. This editor allows to view any SensorML documents (V2.0) and edit the current content. The project has been designed using GWT. | Graphical composition of predefined SensorML of specific Sensors. App for OwnCloud. | SOS server with SensorML editor embedded in the management interface of the SOS | SensorML editor which enables web-based editing of SensorML descriptions |
| last updated                      | 2018   | 2017  | unavailable   | 2009   | 2011  | 2009  | 2018  | 2017  | 2017  | 2018   |
| status                            | stable   | stable  | -   | -  | stable  | stable  | stable  | beta  | stable  | stable   |
| licence                           | GPL  | Mozilla Public License, version 2.0   | undefined   | Mozilla Public License 1.1   | undefined   | Mozilla Public License 1.1  | Mozilla Public License 2  | GNU AFFERO GENERAL PUBLIC LICENSE   | GPL v.2   | Apache License 2.0   |
| SensorML 1                        | Y  | unclear: Documentation does not provide such information. Apparently the source code has only references to SensorML 2.0.0                | Y   | Y  | Y   | Y   | N   | N   | Y   | unclear  |
| SensorML 2                        | Y  | Y   | N   | N  | N   | N   | Y   | Y   | N   | Y  |
| extensibility to other MD schemas | Y  | -   | -   | -  | N   | N   | N   | N   | N   | N  |
| profiles support                  | Y (by EDI template Language)   | -   | -   | -  | Y (validation only by RelaxNG and schematron)                                 | ?   | Y (RelaxNG)   | ?   | ?   | ?  |
| UI type                           | web form   | -   | -   | GUI-visualization  | Java desktop application (with GUI)   | ?   | web form  | GUI, web app for owncloud   | web form  | web form   |

Table A1. Cont.

|   | GET-IT EDI             | SensorML Process Editor | SensorML Schema Browser | SensorML Profile Library | Pines SensorML Editor | SensorML Editor | OpenSensorHub SensorML editor    | SensorNanny-drawmyobservatories | ISTSOS   | 52North-smle                                 |
|---|------------------------|-------------------------|-------------------------|--------------------------|-----------------------|-----------------|----------------------------------|---------------------------------|--|--|
| standalone module   | Y                      | -                       | -                       | -                        | Y                     | Y               | Y                                | Y                               | N  | ?  |
| integrated in other systems   | Y (e.g., GET-IT)       | Y                       | -                       | -                        | N                     | ?               | Y (OpenSensorHub)                | Y (SensorNanny)                 | Y (part of the management interface of IstSOS software)  | ?  |
| type of support for soft typing (e.g., manual insertion of URIs, code-lists stored in local db, remote queries) | runtime SPARQL queries | -                       | -                       | -                        | manual(?)             | ?               | runtime SPARQL queries           | local                           | local db, dynamic (the user can add new values, URIs manually, which will be available for new insertions) | manual insertion of URIs                     |
| SPARQL endpoints and queries customizable   | Y                      | -                       | -                       | -                        | -                     | ?               | need modification of source code | -                               | -  | -  |
| progr. language   | JAVA, javascript       | JAVA                    | -                       | -                        | JAVA                  | JAVA            | JAVA, javascript                 | php, javascript                 | python, javascript   | javascript (AngularJS framework), TypeScript |
| persistence   | JBO, SOS               | -                       | -                       | -                        | ?                     | ?               | XML                              | JSON                            | SOS db   | SOS  |

## References

1. Reed, C.; Botts, M.; Davidson, J. OGC<sup>®</sup> sensor web enablement: overview and high level architecture. In Proceedings of the 2007 IEEE Autotestcon, Baltimore, MD, USA, 17–20 September 2007; pp. 372–380. [CrossRef]
2. OGC. *OpenGIS Sensor Model Language (SensorML) Implementation Specification*; OpenGIS<sup>®</sup> Publicly Available Standard OGC 07-000; Open Geospatial Consortium: Wayland, MA, USA, 2007.
3. OGC. *OGC<sup>®</sup> SensorML: Model and XML Encoding Standard*; Encoding Standard OGC-12-000; Open Geospatial Consortium: Wayland, MA, USA, 2014.
4. Pearlman, J.; Graves, H.; Schaap, D. Ocean Data Interoperability Platform (ODIP): Addressing key challenges for marine data management on a global scale. In Proceedings of the MTS/IEEE Oceans Conference, Monterey, CA, USA, 19–23 September 2016.
5. SPARQL Working Group. *SPARQL 1.1 Query Language*; W3C Recommendation 21 March 2013; World Wide Web Consortium: Cambridge, MA, USA, 2013.
6. W3C XML Schema Working Group. *XML Schema Part 1: Structures*, 2nd ed.; W3C Recommendation 28 October 2004; World Wide Web Consortium: Cambridge, MA, USA, 2004.
7. ISO/TC 211. *Geographic information – Geography Markup Language (GML)*; Standard ISO 19136:2007; International Organization for Standardization: Geneva, Switzerland, 2007.
8. Loubrieu, T.; Détoc, J.; Thorel, A.; Azelmat, H. Sensor Nanny, data management services for marine observation operators. In Proceedings of the EGU General Assembly 2016, Vienna, Austria, 17–22 April 2016; EGU General Assembly Conference Abstracts, Volume 18, p. 12332.
9. Simonis, I.; Echterhoff, J. *GEOSS and the Sensor Web—GEOSS Sensor Web Workshop Report*; GEOSS Task DA 07-04; Geneva, Switzerland, 2008.
10. Malewski, C.; Simonis, I.; Terhorst, A.; Bröring, A. StarFL—A modularised metadata language for sensor descriptions. *Int. J. Digit. Earth* **2012**, *7*, 450–469. [CrossRef]
11. Bröring, A.; Echterhoff, J.; Jirka, S.; Simonis, I.; Everding, T.; Stasch, C.; Liang, S.; Lemmens, R. New generation Sensor Web Enablement. *Sensors* **2011**, *11*, 2652–99. [CrossRef] [PubMed]
12. Hu, C.; Guan, Q.; Chen, N.; Li, J.; Zhong, X.; Han, Y. An Observation Capability Metadata Model for EO Sensor Discovery in Sensor Web Enablement Environments. *Remote Sens.* **2014**, *6*, 10546–10570. [CrossRef]
13. Botts, M.; Reed, C.; Percivall, G.; Davidson, J. OGC Sensor Web Enablement : Overview And High Level Architecture. In *International Conference on GeoSensor Networks*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 4540, pp. 713–723. [CrossRef]
14. Jirka, S.; Broering, A. *OGC<sup>®</sup> SensorML Profile for Discovery*; Discussion Paper OGC 09-033; Open Geospatial Consortium: Wayland, MA, USA, 2009.
15. OGC. *OGC<sup>®</sup> Best Practice for Sensor Web Enablement Lightweight SOS Profile for Stationary In-Situ Sensors*; Best Practice OGC 11-169r1; Open Geospatial Consortium: Wayland, MA, USA, 2014.
16. Library of Congress. About Profiles. 1998. Available online: <http://www.loc.gov/z3950/agency/profiles/about.html> (accessed on 1 July 2016).
17. Ledrick, D.P.; Spring, M.B. International standardized profiles. *Comput. Stand. Interfaces* **1990**, *11*, 95–103. [CrossRef]
18. JTC1/SC34. *Information technology—Document Schema Definition Language (DSDL) —Part 3: Rule-Based Validation—Schematron*; Standard ISO/IEC 19757-3:2006; International Organization for Standardization: Geneva, Switzerland, 2006.
19. JTC1/SC34. *Information Technology—Document Schema Definition Language (DSDL)—Part 2: Regular-Grammar-Based Validation—RELAX NG*; Standard ISO/IEC 19757-2:2008; International Organization for Standardization: Geneva, Switzerland, 2006.
20. OGC. *OGC<sup>®</sup> Sensor Observation Service Interface Standard, Version 2.0*; Standard OGC 12-006; Open Geospatial Consortium: Wayland, MA, USA, 2012.
21. OGC. *OGC<sup>®</sup> Sensor Observation Service 2.0 Hydrology Profile*; Best Practice 14-004r1; Open Geospatial Consortium: Wayland, MA, USA, 2014.
22. Martínez, E.; Toma, D.M.; Jirka, S.; Del Río, J. Middleware for Plug and Play Integration of Heterogeneous Sensor Resources into the Sensor Web. *Sensors* **2017**, *17*, 2923. [CrossRef] [PubMed]



23. Sorribas, J. *SensorML Profiles and O&M Data Models Adapted to Specific Marine Observations Data*; Project Deliverable, Sea Data Net II Pan-European Infrastructure for Ocean and Marine Data Management—FP7 283607; Ifremer: Plouzané, France, 2014. [[CrossRef](#)]
24. OGC. *Extension Package for ebRIM Application Profile: SensorML*; OGC Discussion Paper OGC 09-163r2; Open Geospatial Consortium: Wayland, MA, USA, 2010.
25. OGC. *OGC Sensor Observable Registry (SOR) Discussion Paper*; OGC Discussion Paper OGC 09-112r1; Open Geospatial Consortium: Wayland, MA, USA, 2010.
26. OGC. *Observations and Measurements—XML Implementation*; OGC Implementation OGC 10-025r1; Open Geospatial Consortium: Wayland, MA, USA, 2011.
27. OGC. *Geographic Information—Observations and Measurements*; OGC Standard: Abstract Specification OGC 10-004r3; Open Geospatial Consortium: Wayland, MA, USA, 2013.
28. Semantic Web Deployment Working Group. *SKOS Simple Knowledge Organization System Reference*; W3C Recommendation 18 August 2009; World Wide Web Consortium: Cambridge, MA, USA, 2009.
29. Jirka, S. The Marine Profiles for OGC Sensor Web Enablement Standards Team. Marine Profiles for OGC Sensor Web Enablement Standards. In Proceedings of the EGU General Assembly 2016, Vienna, Austria, 17–22 April 2016; EGU General Assembly Conference Abstracts, Volume 18, p. 14690.
30. Kokkinaki, A.; Buck, J.; Darroch, L. A semantically rich and standardised approach enhancing discovery of sensor data and metadata. In Proceedings of the EGU General Assembly 2016, Vienna, Austria, 17–22 April 2016; EGU General Assembly Conference Abstracts, Volume 18, p. 12970.
31. Leadbetter, A.; Lowry, R.; Clements, D. The NERC Vocabulary Server: Version 2.0. In Proceedings of the EGU General Assembly 2012, Vienna, Austria, 22–27 April 2012.
32. Lowry, R.K. 25 Years of Controlled Vocabularies in Oceanographic Data Management. In Proceedings of the 2008 Fall Meeting, San Francisco, CA, USA, 15–19 December 2008.
33. Rueda, C.; Bermudez, L.; Fredericks, J. The MMI Ontology Registry and Repository: A portal for Marine Metadata Interoperability. In Proceedings of the OCEANS 2009, Biloxi, MS, USA, 26–29 October 2009; pp. 1–6. [[CrossRef](#)]
34. Kokkinaki, A.; Darroch, L.J.; Buck, J.J.H.; Jirka, S.; the “Marine Profiles for OGC Sensor Web Enablement Standards” Team. Semantically enhancing SensorML with Controlled Vocabularies in the Marine Domain. In Proceedings of the Geospatial Sensor Webs Conference 2016 (GSW 2016), Muenster, Germany, 29–31 August 2016; CEUR-WS.org., Jirka, S., Stasch, C., Hitchcock, A., Eds.; Volume 1762.
35. Koppe, R.; Gerchow, P.; Macario, A.; Haas, A.; Schäfer-Neth, C.; Pfeiffenberger, H. O2A: A generic framework for enabling the flow of sensor observations to archives and publications. In Proceedings of the OCEANS 2015, Genova, Italy, 18–21 May 2015; pp. 1–6. [[CrossRef](#)]
36. Basoni, A.; Bastianini, M.; Fugazza, C.; Menegon, S.; Minuzzo, T.; Oggioni, A.; Pavesi, F.; Pepe, M.; Sarretta, A.; Tagliolato, P.; Carrara, P. Fostering bottom-up capacity in managing and sharing marine observations: the RITMARE StarterKit. In Proceedings of the EuroGOOS 2014, Sopot, Poland, 4–6 October 2014.
37. Pavesi, F.; Basoni, A.; Fugazza, C.; Menegon, S.; Oggioni, A.; Pepe, M.; Tagliolato, P.; Carrara, P. EDI—A template-driven metadata editor for research data. *J. Open Res. Softw.* **2016**, *4*, e40. [[CrossRef](#)]
38. Oggioni, A.; Tagliolato, P.; Fugazza, C.; Pepe, M.; Menegon, S.; Pavesi, F.; Carrara, P. Interoperability in marine sensor networks through SWE services. In *Oceanographic and Marine Cross-Domain Data Management for Sustainable Development*; Diviacco, P., Leadbetter, A., Graves, H., Eds.; IGI Global: Hershey, PA, USA, 2017.
39. Fugazza, C.; Pepe, M.; Oggioni, A.; Tagliolato, P.; Carrara, P. Raising Semantics-Awareness in Geospatial Metadata Management. *ISPRS Int. J. Geo-Inf.* **2018**, *7*. [[CrossRef](#)]
40. The Rule Interchange Format (RIF) Working Group. *RIF Overview*, 2nd ed.; W3C Working Group Note 05 February 2013; World Wide Web Consortium: Cambridge, MA, USA, 2013.
41. Horrocks, I.; Patel Schneider, P.F.; Boley, H.; Tabet, S.; Grosz, B.; Dean, M. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*; W3C Member Submission 21 May 2014; World Wide Web Consortium: Cambridge, MA, USA, 2004.
42. Fugazza, C.; Tagliolato, P.; Frigerio, L.; Carrara, P. Web-Scale Normalization of Geospatial Metadata Based on Semantics-Aware Data Sources. *ISPRS Int. J. Geo-Inf.* **2017**, *6*. [[CrossRef](#)]
43. RDF Working Group. *RDF 1.1 Concepts and Abstract Syntax*; W3C Recommendation 25 February 2014; World Wide Web Consortium: Cambridge, MA, USA, 2014.

44. Fugazza, C.; Pepe, M.; Oggioni, A.; Tagliolato, P.; Pavesi, F.; Carrara, P. Describing Geospatial Assets in the Web of Data: A Metadata Management Scenario. *ISPRS Int. J. Geo-Inf.* **2016**, *5*, 229. [CrossRef]
45. RDF Working Group. *RDF Schema 1.1*; W3C Recommendation 25 Feb 2014; World Wide Web Consortium: Cambridge, MA, USA, 2014.
46. OWL Working Group. *OWL 2 Web Ontology Language Document Overview*, 2nd ed.; W3C Recommendation 11 December 2012; World Wide Web Consortium: Cambridge, MA, USA, 2012.
47. Members of the FOAF Mailing List. FOAF Vocabulary Specification 0.99. Technical Report, 2014. Available online: <http://xmlns.com/foaf/spec/> (accessed on 2 April 2019).
48. Zilioli, M.; Oggioni, A. *LTER-Italy EBVs and EI Indicators Inventory*; [Data Set]; Zenodo: Genève, Switzerland, 2018. [CrossRef]
49. Zilioli, M.; Oggioni, A. *RITMARE Data Portal (v.0.0) Research Data Catalogue*; [Data Set]; Zenodo: Genève, Switzerland, 2018. [CrossRef]
50. Zilioli, M.; Lanucara, S.; Oggioni, A.F.C.; Carrara, P. Fostering Data Sharing in Multidisciplinary Research Communities: A Case Study in the Geospatial Domain. *Data Sci. J.* **2019**, *18*, 15. [CrossRef]
51. Zilioli, M.; Oggioni, A.; Tagliolato, P.; Pugnetti, A.; Carrara, P. Feeding Essential Biodiversity Variables (EBVs): actual and potential contributions from LTER-Italy. *Nat. Conserv.* **2019**, *34*, 477–503. [CrossRef]
52. Fredericks, J.; Botts, M. Promoting the capture of sensor data provenance: A role-based approach to enable data quality assessment, sensor management and interoperability. *Open Geospat. Data Softw. Stand.* **2018**, *3*, 3. [CrossRef]
53. Bröring, A.; Janowicz, K.; Stasch, C.; Kuhn, W. Semantic Challenges for Sensor Plug and Play. In *Web and Wireless Geographical Information Systems*; Carswell, J.D., Fotheringham, A.S., McArdle, G., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 72–86.
54. Peckham, S.D. The CSDMS standard names: cross-domain naming conventions for describing process models, data sets and their associated variables. In *Proceedings of the 7th International Congress on Environmental Modelling and Software*, San Diego, CA, USA, 15–19 June 2014.
55. Oggioni, A. *oggioniale/RDF-FOAF-Manufacturer-List: First Release of RDF-FOAF Manufacturers List*; [Data Set]; Zenodo: Genève, Switzerland, 2019. [CrossRef]



© 2019 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/>).