UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Centro de Investigación en Tecnoloxías da Información

Departamento de Electrónica e Computación

Thesis

SEMANTIC MEDIATION OF ENVIRONMENTAL OBSERVATION DATASETS THROUGH SENSOR OBSERVATION SERVICES

Author:

D. Manuel Antonio Regueiro Seoane

PhD Supervisors:

Dr. D. José Ramón Ríos Viqueira

Dr. D. José Ángel Taboada González

September 2016

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Dr. D. José Ramón Ríos Viqueira, Profesor Contratado Doutor da Área de Linguaxes e Sistemas da Universidade de Santiago de Compostela e investigador adscrito ao Centro Singular de Investigación en Tecnoloxías da Información (CITIUS)

Dr. D. José Ángel Taboada González, Profesor Titular de Universidade da Área de Linguaxes e Sistemas da Universidade de Santiago de Compostela e investigador adscrito ao Centro Singular de Investigación en Tecnoloxías da Información (CITIUS)

FAN CONSTAR:

Que a memoria titulada **SEMANTIC MEDIATION OF ENVIRONMENTAL OBSERVATION DATASETS THROUGH SENSOR OBSERVATION SERVICES** foi realizada por **D. Manuel Antonio Regueiro Seoane** baixo a nosa dirección no Centro Singular de Investigación en Tecnoloxías da Información da Universidade de Santiago de Compostela, e constitúe a Tese que presenta para obtar ao título de Doutor.

Setembro 2016

Dr. D. José Ramón Ríos Viqueira
Director da tese

Dr. D. José Ángel Taboada González
Codirector da tese

Manuel A. Regueiro Seoane
Autor da tese

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

A Mamá, Papá, María, Javier e Mónica

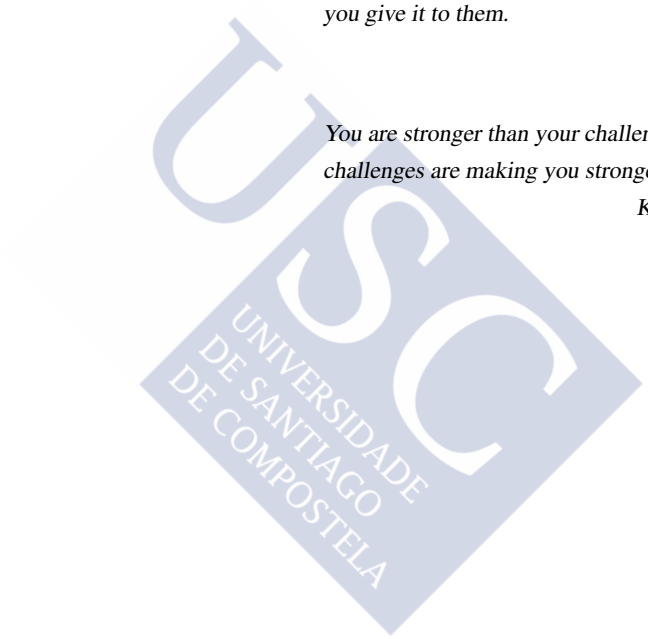
USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

No one has the power to shatter your dreams unless you give it to them.

Maeve Greyson

You are stronger than your challenges and your challenges are making you stronger.

Karen Salmansohn



USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Acknowledgments

The end of a stage in my life is coming, and although recently I saw that moment too distant, this almost has arrived. Many people and organizations made this journey doable, so to all of them I want to thank for their support and help during this walk.

I would like to thank my supervisor Dr. José Ramón Ríos Viqueira. Thank you for teaching me, helping me, for supporting me and guiding me towards my goals that I saw very far at the beginning. I would like to express my deepest gratitude not only for being my supervisor, but also for being a friend, a colleague, someone who always helped me in many ways, like a lighthouse during this stage of my life.

I would like to thank Dr. Christoph Stasch, who gave me a warm welcoming in Germany, offering me the opportunity to participate in his daily life and learn from him the way German people work. I appreciate his kindness and humility. I also appreciate the support of the Spatio-Temporal Modelling Lab (STML) of the Institute for Geoinformatics (IFGI) of the University of Munster.

I would like to thank my parents, my siblings and my girlfriend. Although they did not participate directly in this research, they contributed with their support and love.

There are many people I have to thank for fruitful communication, sharing their ideas, giving error reports, and so on. Thank you everyone.

Finally, I would like to thank all the organizations that collaborate with this work. This doctoral dissertation could not be possible without their financial support:

- Xunta de Galicia that sponsored my research through the grant "Apoio á etapa predoutoral do Plan Galego de Investigación, Innovación e Crecemento 2011-2015" (Plan I2C) to the year 2011.

- Deputación da Coruña that funded my research through the grant "Bolsa de investigación en enxeñería e arquitectura" (Programa BINV-EA/2015).

– Centro Singular de Investigación en Tecnoloxías da Información (CITIUS), that funded my predoctoral stay in Munster and hosted me these years.

– MeteoGalicia and INTECMAR for their collaboration through different projects during these years.

– My research group “Computer Graphics and Data Engineering Group” (COGRADE), where I was always warmly treated from the beginning. Thank you teachers and colleagues.

Agradecementos

O final dunha longa etapa aproxímase, e aínda que fai pouco tempo se vía ese intre moi lonxe, este case chegou. Moitas persoas e organizacións fixeron posible este percorrido, a todos eles lles quero agradecer o seu apoio e axuda durante este camiñar.

Primeiramente gustaríame darlle as grazas ó meu director de tese Dr. José Ramón Ríos Viqueira. Grazas por ensinarme, por axudarme en todo momento e por guiarme cara a metas que, con tempo, seguramente recordarei con moito máis valor. Grazas tamén a todos os profesores e compañeiros do grupo de investigación. Realmente agradécese poder traballar nunha gran familia como a que formamos.

Quero darlle as grazas tamén ao Dr. Christoph Stasch, que con gran amabilidade e afecto acolleu a un total descoñecido en Alemaña, brindándome a oportunidade de participar na súa vida cotiá e poder empaparme doutra cultura.

Grazas a meus pais, irmáns e moza, que ainda que non participaron directamente no traballo de investigación, contribuiron co seu apoio e cariño.

Por último, darlle as gracias a todas as organizacións que me apoiaron e que contribuirón para que este traballo fose posible:

- Xunta de Galicia a través da concesión da axuda de apoio á etapa predoutoral do Plan Galego de Investigación, Innovación e Crecemento 2011-2015 (Plan I2C) para o ano 2011.

- Deputación da Coruña a través da concesión da bolsa de investigación en enxeñería e arquitectura (Programa BINV-EA/2015).

- Centro Singular de Investigación en Tecnoloxías da Información (CITIUS) pola súa axuda, sobre todo en todos os temas burocráticos, e pola concesión dunha bolsa para realizar unha estadía predoutoral moi enriquecedora en Alemaña.

– MeteoGalicia e INTECMAR pola súa colaboración a través de diferentes proxectos ó longo destos anos.

– Grupo de investigación “Gráficos por Ordenador e Enxeñaría de Datos” (COGRADE), no cal me acolleron con moito afecto dende o comezo e sempre me brindaron toda a axuda e apoio necesario para realizar a tese.

Contents

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Abstract

A large volume of environmental observation data is being generated as a result of the observation of many properties at the Earth surface. It is estimated that this amount is going to increase considerably in the future due to the advances in the sensing devices hardware. In parallel, there exists a clear interest in accessing data from different data providers related to the same property, in order to solve concrete problems. For example, if we restrict to the area of Galicia, we find regional, national and international organizations that manage intersecting sets of meteorological stations. All of these organizations have huge amounts of available data, sometimes overlapped, but with a heterogeneous storage and access, hampered their integration in an automatic way. Based on such fact, there is also an increasing interest in publishing the above data through open interfaces in the scope of Spatial Data Infraestructures (SDIs).

There have been important advances in the definition of open standards of the Open Geospatial Consortium (OGC) that enable interoperable access to sensor data. Among the proposed interfaces, the Sensor Observation Service (SOS) is having an important impact in the development of current environmental information systems. This service enables standardized access to collections of observation data generated by different processes, which are in most cases physical sensors. Observations inside a SOS are organized in Offerings. Each observation of each offering has a value of an Observed Property for a given time instant. Besides, the observation references the domain specific entity to which the property applies, called Feature of Interest and the Process used to obtain the value, commonly a physical sensor. The Observations and Measurements (O&M) specification provides a data model for those observation collections.

We have realized that currently there is no available solution to provide integrated access to various data sources through a SOS interface. This problem shows up two main facets. On the one hand, the heterogeneity among different data sources has to be solved. On the other hand, semantic conflicts that arise during the integration process must also resolved with the help of relevant domain expert knowledge.

The most direct solution would be given by an ad-hoc implementation on the client side. The main drawbacks would be the lack of generality and the need to implement complex clients for specialized users. From the server side there are two clear alternatives for data integration. The first solution would be a data warehouse approach. This solution has to support both entity-based and array-based data models. Therefore, Extract Transform and Load (ETL) processes are needed for each data source, where the heterogeneity and semantic conflicts have to be solved. All the current SOS implementations follow this philosophy; however they are restricted to observations generated by in-situ devices in relational technologies. The second alternative would be a semantic mediation approach. In this case, the queries through SOS have to be transformed in suitable queries to each data source. Mediator/Wrapper architectures are used in this case, where an adapter for the model and format is developed for each data source, and a mediator is developed to resolve semantic conflicts. There is currently no SOS implementation of this type.

To solve the problems introduced in the preceding paragraphs, the main goal of this thesis is to design and develop a semantic data mediation framework to access any kind of environmental observation dataset, including both relational data sources and multidimensional arrays. The whole proposed solution will use a Mediator/Wrapper approach. The mediator will use semantic technologies to solve conflicts. Generic wrappers for spatial databases and for multidimensional array data sources accessed through NetCDF Subset interface will reduce the development cost.

Resumo

Diariamente estanse xerando enormes cantidades de datos medioambientais como resultado da observación de moitas propiedades sobre a superficie terrestre, e estímase que aumente de forma considerable no futuro. Isto é debido, en parte, á gran diversidade de sensores que se están instalando grazas ós importantes avances que se están producindo no desenvolvemento hardware. En concreto, a redución de costo dos sensores está propiciando o uso masivo deste tipo de sensores. A este feito hai que sumar a emerxencia de novas iniciativas de crowdsourcing para a adquisición de datos (crowdsensing). Á par que este escenario existe un claro interese polo acceso a varias fontes de datos medioambientais co fin de resolver problemas concretos ou realizar determinados estudos. A modo de exemplo, é vital analizar históricos de temperatura, salinidade, cantidade de chuvia, etc. nunha determinada zona para a predición das condicións en que se podería producir un brote de cólera. Outro exemplo significativo é o estudo do cambio climático. Moitas organizacións a nivel mundial tratan este tema analizando principalmente as temperaturas dende fai décadas co fin de estudar a evolución e atopar patróns de comportamento. Imaxinemos que para un estudio nos restrinximos a unha zona tan pequena como Galicia, pois atopámonos con que existen datos medioambientais xestionados por diferentes organizacións, entre elas:

– *Redes rexionais*: Dentro das rexionais están dispoñibles estacións oceanográficas do Instituto Tecnolóxico para o Control do Medio Mariño de Galicia (INTECMAR) e estacións meteorolóxicas da Axencia Galega de Meteoroloxía (MeteoGalicia).

– *Redes estatais*: A rede estatal ten todo tipo de estacións e sensores despregados por toda a xeografía española, entre os cales destacan a rede de estacións meteorolóxicas da Axencia Estatal de Meteoroloxía (AEMET).

– *Redes internacionais*: Aquí podemos atopar tamén redes de estacións da Global Historical Climatology Network (GHCN), as cales están accesible a través da National Oceanic and Atmospheric Administration (NOAA) do departamento de comercio dos Estados Unidos.

Estas organizacións teñen moitos datos dispoñibles, a veces solapados, pero sobre todo con un almacenamento e un acceso heteroxéneo, dificultando notablemente a súa integración de forma automática. Además existe unha tendencia para a súa publicación a través de interfaces estándar no ámbito das Infraestructuras de Datos Espaciais (IDEs). Este feito cobra máis importancia cando os provedores son entidades públicas. Por exemplo, en Europa, a directiva INSPIRE do 2007 lexisla sobre a creación de ditas IDEs e obriga a facer públicos os datos de observación a través de interfaces estandarizadas.

Fixéronse importantes avances na definición de estándares abertos dentro da iniciativa Sensor Web Enablement (SWE) do Open Geospatial Consortium (OGC). De especial importancia é a interface Sensor Observation Service (SOS), que proporciona acceso web estándar a colección de datos de observación. Este estándar conta con tres operacións obrigatorias que todos os SOS deben implementar, i) *GetCapabilities*: esta operación permite obter os metadatos do servizo (identificación do servizo, provedor, operacións dispoñibles e filtros soportados) así como a lista de ofertas dispoñibles onde están agrupadas as observacións que se poden consultar, ii) *DescribeSensor*: esta operación proporciona unha descrición do proceso ou sensor empregado para tomar a medición dunha determinada propiedade (a observación) e iii) *GetObservation*: esta operación devolve todas aquelas observacións que concordan coa lista de filtros que o usuario especifica. Estes filtros poden ser por rango de tempo, area espacial, sensor, propiedade, estación ou valor.. Cada observación, que pode pertencer a unha ou varias ofertas, ten un valor (por exemplo 15^{o}C) sobre unha propiedade (a temperatura) nun instante de tempo. Ademais, a observación referencia unha entidade de interese onde se toma o valor (mídese a temperatura nunha estación meteorolóxica) e un proceso empregado para obtelo, normalmente un sensor (sensor de temperatura). Toda esta información represéntase utilizando o estándar Observations and Measurements (O&M), cuxa especificación prové un modelo de datos e a súa codificación XML. Finalmente o estándar Sensor Modelling Languaje (SensorML) proporciona unha linguaxe XML para describir os procesos de obtención de observacións. Existen outros estándares que poden utilizarse para acceder a datos de observación, como o Web Feature Service (WFS) ou o Web Coverage Service (WCS), pero solo

o estándar SOS captura a semántica dos datos que ten que ver coa observación.

O problema a resolver é por tanto prover dun acceso integrado a varias fontes de datos de observación a través da interface SOS. Este problema presenta dúas facetas principais. Nun primeiro lugar ten que resolverse a heteroxeneidade que existe entre as distintas fontes de datos. Esta heteroxeneidade débese a distintos factores.

– *Paradigma de modelado de datos*: En primeiro lugar, os datos xerados polos distintos tipos de sensores teñen natureza distinta e modélanse con paradigmas diferentes. En xeral, os datos xerados por sensores que miden in-situ encaixan nos modelos baseados en entidades e sóense xestionar polo tanto con tecnoloxías relacionais. Exemplos deste tipo de sensores son as estacións meteorolóxicas (plataforma estática) e os sensores da radio sonda (plataforma móbil). Por outro lado, os sensores de tipo remoto proporcionan en cada instante unha mostraxe espacial dunha determinada propiedade a unha determinada distancia do sensor, e polo tanto, os seus datos encaixan en modelos e tecnoloxías de xestión de arrays espacio-temporais. Exemplos deste tipo son os radares (plataformas estáticas) tales como o Radar de Alta Frecuencia (HFRadar) empregado para medir as correntes mariñas e o Radar Atmosférico (WSRadar) empregado para medir as precipitacións. Os sensores a bordo de satélite (plataforma móbil) son tamén exemplos deste tipo de sensores que miden en remoto.

– *Modelo de datos*: Incluso si nos restrinximos ó mesmo paradigma de modelado, por exemplo o relacional, distintas fontes de datos utilizarán distintos modelos para o almacenamento. Así, por exemplo, o modelo Entidade/Relación (ER) empregado pola axencia meteorolóxica rexional galega (MeteoGalicia) para o almacenamento das observacións xeradas polas súas redes de estacións meteorolóxicas é moi distinto do empregado polo Instituto Tecnolóxico para o Control do Medio Mariño (INTECMAR) para o almacenamento das súas observacións de perfiles CTD (Current, Temperature, Depth).

– *Formatos e interfaces de acceso a datos*: Finalmente, en calquera caso, tantos as interfaces de acceso como os formatos empregados para a codificación dos datos cambian moito dun provedor a outro, incluso sendo moi similares os datos. Varían dende formatos de texto binarios propietarios accesibles a través de protocolos de descarga como o File Transference Protocol (FTP) ata interfaces de servizo web baseadas en estándares internacionais como o NetCDF Subset.

Unha segunda faceta do problema da integración de datos é a que ten que ver cos conflitos semánticos. Estes conflitos pódense clasificar como segue:

– Conflitos derivados da asignación de distintos identificadores en distintas fontes de datos para o mesmo concepto do modelo de observación. Un exemplo disto é a asignación de distintos identificadores á mesma estación meteorolóxica cando é accedida a través dunha fonte ou outra, sexa MeteoGalicia ou AEMET. Tamén é habitual ter nas fontes anteriores a existencia de distintos identificadores para a mesma propiedade medioambiental, por exemplo *temperatura do aire*. Ante esta situación, o proceso de integración de datos ten que solucionar estes conflitos de tal xeito que se poidan combinar observacións de ambas fontes eliminando sempre os duplicados.

– Conflitos derivados da asignación do mesmo identificador a conceptos que son distintos en dúas fontes de datos distintas. Un exemplo típico é a interpretación distinta que se pode dar a un identificador de propiedade (por exemplo a *temperatura*) en fontes distintas: temperatura do chan, temperatura do aire a 1 metro, temperatura do aire a 10 metros, temperatura da superficie do mar, temperatura da auga dun río, etc. Neste caso o proceso de integración de datos ten que engadir o contexto de cada fonte de datos aos identificadores das observacións de tal xeito que se evite ambigüidade.

Poderíanse adoptar diferentes solucións dende o punto de vista da arquitectura do sotfware para dar resposta aos conflitos arriba mencionados. Neste senso no marco da arquitectura clásica Cliente/Servidor, a solución máis directa daríase mediante unha implementación ad-hoc no lado cliente. As principais desvantaxes serían, primeiro a falta de xeneralidade na solución e segundo a necesidade de implementar clientes complexos para usuarios especializados. Dende o lado do servidor existen dúas alternativas claras para a integración dos datos. A primeira sería unha solución de tipo almacén de datos (Data Warehouse – DW). Neste caso necesitaríase un almacén integrado que empregue un modelo de datos e unha tecnoloxía que de soporte por tanto a datos de entidades e tamén arrays espacio-temporais (os dous paradigmas anteriormente mencionados). Ademais, precísase a implementación de procesos de extracción, transformación e carga (Extract Transform and Load – ETL) para cada fonte, nos que deben resolverse os problemas das heteroxeneidades e dos conflitos semánticos. A totalidade de implementacións SOS actuais seguen esta filosofía. A maioría restrínxense a datos de sensores in-situ almacenados en tecnoloxía relacional. Ningunha da soporte integrado para os datos de entidades e arrays. Finalmente, os procesos ETL teñen que ser implementados de

forma ad-hoc para cada fonte de datos. A segunda alternativa sería empregar a mediación de datos. Neste caso, as consultas SOS transfórmanse en consultas adecuadas a cada fonte de datos, sen necesidade de dispoñer dun almacén común integrado. Utilízanse neste caso arquitecturas de tipo Mediador/Adaptador, nas que se implementa un adaptador para cada modelo, interface e formato de cada fonte de datos e un mediador que resolve os conflitos semánticos. Non existe na actualidade ningunha solución SOS deste tipo.

É importante destacar que unha solución tipo almacén de datos replica no almacén datos de cada fonte, co que pode ofrecer unha maior eficiencia. Sen embargo, unha solución de mediación de datos permite o acceso ás últimas observacións de cada fonte, sen ter que esperar a que se execute o proceso ETL, e reduce o custe de almacenamento. Na práctica, o máis común será empregar unha solución híbrida que combine almacéns con mediadores. Se nos restrinximos estritamente a integración de datos de observación na actualidade, as implementacións de SOS existentes seguen unha aproximación de almacén de datos e soportan soamente datos de entidades xerados por sensores in-situ que están gardados con tecnoloxías relacionais, é dicir, datos de sensores estáticos que miden en local e gardan os datos en base de datos (por exemplo unha estación meteorolóxica). Esta realidade fainos albiscar que non existe ningunha implementación de SOS que permita a integración semántica de datos de diferentes provedores, aparte da proposta feita nesta tese.

En base ó exposto nos parágrafos anteriores, o obxectivo da tese doutoral é o deseño e implementación dunha solución de mediación semántica de datos que permite o acceso a través de SOS a fontes de datos de observación medioambiental, que inclúan tanto fontes relacionais como de arrays multidimensionais. A solución proposta utilizará unha arquitectura de tipo Mediador/Adaptador, na que os adaptadores resolverán problemas de heteroxeneidade entre as fontes e na que o mediador resolverá os conflitos semánticos. Ademais, para reducir o custe de desenvolvemento, xeneralizarase a implementación de dous tipos de adaptadores. Un adaptador xenérico para acceder a calquera fonte de datos implementada mediante un Sistema Xestor de Base de Datos (SXBD) relacional con capacidades espaciais, e un adaptador xenérico para acceder a fontes de datos de arrays espacio-temporais publicadas mediante servizos de tipo NetCDF Subset. Máis polo miúdo as tres contribucións principais desta tese son:

1. Integración virtual para datos de observación de sensores de distintas fontes de datos. Con esta aproximación conseguimos un acceso integrado a datos de entidades e arrays espacio-temporais de forma conxunta, é dicir, resolvéronse os problemas de heteroxeneidade, aínda que está limitada na resolución de conflitos semánticos.

2. Mediación semántica de datos de observación. Coa incorporación de tecnoloxías semánticas á aproximación mencionada no punto anterior, conseguimos incorporar relacións complexas entre os conceptos das propias fontes de datos definidas polo experto do dominio, as cales son empregadas durante a integración de datos de tal xeito que se conseguimos resolver os problemas semánticos.

3. Desenvolvemento de adaptadores xenéricos para observacións de entidades e arrays multidimensionais. Desenvolveuse un adaptador xenérico para todas aquelas observacións que se gardan en bases de datos espaciais. Deste xeito calquera fonde de datos que empregue un modelo entidade-relación, poderá integrar as súas observacións. Por outra banda, desenvolveuse un adaptador xenérico para observacións tomadas en remoto, que son aquelas que usualmente se gardan en ficheiros NetCDF e que son accedidas a través de servidores TRHEDDS mediante o protocolo NecCDF Subset. Ambos adaptadores proporcionarán acceso xenérico pero tamén eficiente a calquera de ambas fontes de datos, que a o mesmo tempo son as tecnoloxías máis estendidas.

Nun primeiro prototipo da plataforma acadouse o primeiro dos obxectivos, a integración virtual, empregando a xa citada arquitectura Mediador/Adaptador, onde cada adaptador encárgase de adaptar o modelo local da fonte de datos ao modelo definido por O&M e onde o mediador permite a definición de ofertas globais sobre as definicións das ofertas locais. Para a definición destas ofertas globais deseñouse un modelo de datos de integración, Observation Data Integration Model (ODIM). O ODIM fai posible a definición de ofertas globais a partires de ofertas locais para conseguir a integración de observacións das diferentes fontes de datos. Ademais, ODIM tamén permite a posibilidade de definir asociacións entre conceptos globais da plataforma e conceptos relevantes de cada unha das fontes de datos dispoñibles. Neste senso o máis destacable é a posibilidade de definir propiedades globais que subsumen á propiedades locais de cada fonte de datos. Por exemplo, podemos definir a propiedade global "Temperatura", a cal podería subsumir diferentes "Temperaturas do aire" medidas a diferentes elevacións en fontes de datos de estacións meteorolóxicas ou oceanográficas. Sen embargo, ODIM móstrase limitado neste senso tanto para sensores como estacións, xa que

se asume que son diferentes en cada fonte de datos e non se poden definir de forma global. Esta simplificación foi asumida xa que non é común que se produza en escenarios reais. En consecuencia, os identificadores globais para os sensores e as estacións son obtidos mediante a concatenación do identificador local, un delimitador específico e o identificador da fonte de datos de onde provén. Un exemplo de sensor podería ser "Meteo_SantiagoEOASHMP155", onde se identifica un sensor de temperatura e humidade localizado en Santiago que pertence á rede de MeteoGalicia. Outro exemplo de estación podería ser "Oceano_FaroBorneira", onde se identifica unha estación oceanográfica (Boia mariña).

No segundo prototipo da plataforma, coa mediación semántica (segundo obxectivo), conseguíronse mellorar as limitacións semánticas que presentaba ODIM mediante a incorporación de ontoloxías e tecnoloxía semántica. Recordemos que estes conflitos semánticos aparecen cando diferente terminoloxía se emprega en diferentes fontes de datos para referirse os mesmos conceptos ou a mesma terminoloxía é empregada en diferentes fontes de datos para denotar diferentes conceptos. Polo tanto, mellorouse a arquitectura anterior e para elo, agora substituíuse ODIM por ontoloxías para a especificación do coñecemento de integración necesario por parte do experto do dominio. Estas ontoloxías estenden "Semantic Sensor Network" (SSN) e "Semantic Web for Earth and Environmental Terminology" (SWEET), ontoloxías de alto nivel ben coñecidas no ámbito do modelado de sensores e datos medioambientais. Teremos polo tanto unha ontoloxía por cada adaptador que se engada á arquitectura, a cal definirá o experto desa fonte de datos, e outra ontoloxía no mediador que se encargará de integrar e resolver os conflitos semánticos que se poidan dar entre os conceptos provenientes de cada fonte de datos, e que constrúe un experto no dominio. Esta aproximación ten dúas vantaxes principais: i) permite engadir novos dominios de aplicación con novas fontes de datos implementando novos adaptadores e creando a súa ontoloxía asociada e ii) dótase a usuarios sen un coñecemento específico no dominio de aplicación da posibilidade de desenvolver novas aplicacións de propósito xeral que fagan uso do coñecemento que un experto do dominio especificou nas ontoloxías mediante relacións semánticas. Estas relacións semánticas entre conceptos globais e locais atópanse dentro do denominado "coñecemento de integración de datos", que ademais inclúe a definición de novas clases ou a definición de ofertas globais. Son tres os tipos de relacións semánticas que se poden especificar entre as clases e individuos das ontoloxías das fontes de datos e a do mediador.

– Relacións de subclase: son relacións que se representan pola propiedade "subClassOf" de RDFS (Resource Description Framework Schema) e permiten a integración de varias clases de propiedades, de sensores ou de estacións nunha soa clase. Por exemplo, poderíamos ter unha clase que represente unha estación meteorolóxica identificada como "meteo_EstacionAutomatica" (fonte de datos de estacións meteorolóxicas) e outra clase que represente unha estación oceanográfica identificada como "oceano_EstacionOceanografica" (fonte de datos de boias oceanográficas) que se integren na clase "med_Estacion" do mediador, que a súa vez estea relacionada coa clase "Station" de SWEET.

– Relacións de equivalencia de clase: son relacións que se representan polo predicado "equivalentClass" de OWL (Ontology Web Language) e permiten que varias clases de propiedades, sensores ou estacións representen o mesmo concepto, a pesares de ter diferentes identificadores en diferentes fontes de datos. Esta relación permite consultar ofertas dunha determinada fonte de datos empregando conceptos definidos noutra fonte de datos diferente. Por exemplo, poderíamos representar que a clase "meteo_Temperatura" da fonte de datos de estacións meteorolóxicas é equivalente á clase "oceano_TemperaturaAire" da fonte de datos de boias oceanográficas.

– Relacións de igualdade de individuos: son relacións que se representan polo predicado "sameAs" de OWL e permiten especificar que realmente dúas propiedades, dous sensores ou dúas estacións representan o mesmo individuo. É dicir, represéntase o feito de que un mesmo individuo poida estar en fontes de datos diferentes con diferentes identificadores. Por exemplo, a estación meteorolóxica "meteo_SantiagoEOAS" de MeteoGalicia é exactamente a mesma estación que "aemet_Santiago" que ten AEMET.

En resumo, esta mellora da plataforma de integración de datos de observación coa incorporación de tecnoloxías semánticas, resulta unha gran achega ao estado do arte xa que supón o primeiro intento de integrar semanticamente datos medioambientais a través de SOS.

Finalmente, na terceira evolución da plataforma (terceiro obxectivo), implementáronse dous adaptadores xenéricos que facilitan a incorporación de novas fontes de datos. Chegados a este punto da tese e de desenvolvemento da plataforma, detectouse que son dous os principais paradigmas de modelado que se empregan para representar a información medioambiental. Por unha banda, os sensores que miden in-situ xeran series temporais que encaixan

cos modelos clásicos entidade-relación de bases de datos. Deste xeito, implementouse un adaptador xenérico para Sistemas Xestores de Bases de Datos (SXBD), permitindo a fácil incorporación de novas fontes de datos en empreguen este formato. Esta adaptador funciona contra un modelo xenérico, ó cal se teñen que adaptar os diferentes modelos locais. O que nos primeiros prototipos da plataforma se facía implementando adaptadores ad-hoc en Java agora faise con vistas en SQL sobre os modelos locais. Por outra banda os sensores que miden en remoto xeran grandes arrays de datos espacio-temporais que se gardan en ficheiros. O adaptador xenérico que atende a este tipo de almacenamento implementouse de tal xeito que se pasou de ter que navegar por ficheiros de metadatos dun servidor Thredds onde se publicaban os datos, a ter que simplemente definir a Uniform Resource Locator (URL) do servidor e o catálogo de datos que se queren publicar. O novo adaptador xenérico crea automaticamente as ontoloxías necesarias para integrar os datos na plataforma.

En conclusión, a plataforma soluciona un problema real de integración virtual de datos de fontes heteroxéneas en dominios medioambientais. Ademais, dotouse da capacidade de mediación semántica necesaria para poder integrar propiedades, sensores e estacións meteorolóxicas de diferentes fontes de datos. Finalmente, coa implementación de adaptadores xenéricos que soportan os dous principais paradigmas de almacenamento, conséguese dotala plataforma dun sistema fácil e áxil de incorporación de novas fontes de datos.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

CHAPTER 1

INTRODUCTION

1.1 Background

A large volume of environmental observation data is being generated as a result of the observation of many distinct environmental properties at the Earth surface every day. In general, observation data may be manually provided by experts of a specific area or automatically or semi automatically generated with the support of sensing devices and processes. It is estimated that the amount automatically generated data is going to increase considerably in the future due to the advances in relevant hardware technologies. In particular, the reduction of the cost of sensors will cause the increasing in the number os such devices used in practice. Besides, the improvement of their processing power and energy consumption efficiency will result in an increment in the amount of data generated by each device. To this fact must be added the emergence of data acquisition crowd sourcing initiatives [27].

In parallel with this scenario, there exists a clear interest in accessing data from different data providers related to the same property for a specific area, in order to solve concrete problems or make certain investigations. For example, the access to different networks of meteorological stations to obtain temperature and humidity data to analyze the impact of those parameters in virus transmission is essential [45]. Another outstanding example is climate change and global warming research. Many organizations worldwide have issued this topic analyzing mainly temperatures during the last decades. If we restrict to the area of Galicia, various networks of meteorological stations are available.

- *Regional networks*: Oceanographic stations of the Technological Institute for the Control of Marine Environment in Galicia (INTECMAR) and meteorological stations of Galician regional meteorological agency (MeteoGalicia)
- *National networks*: Meteorological stations of the Spanish national meteorological agency (AEMET).
- *International networks*: Meteorological stations of the Global Historical Climatology Network (GHCN), which may be accessed through the National Oceanic and Atmospheric Administration (NOAA) of the U.S. Department of Commerce.

All of these organizations have huge amounts of available data, sometimes overlapped, but with a heterogeneous storage and access, hampered in a great manner their integrated access. There is also an increasing interest in publishing the above data through open interfaces in the scope of Spatial Data Infrastructures (SDIs). The above fact is especially relevant in public organizations. For example, the European directive INSPIRE, of May 2007, establishes the rules for the creation of a SDI in Europe. Such rules forces public administrations of EU countries to make their geospatial data accessible through well-known standard interfaces. Related to the above, and focusing on observations data, there have been important advances in the definition of open standards within the Sensor Web Enablement (SWE) initiative of the Open Geospatial Consortium (OGC). Those standards include services for discovering sensors, tasking of sensors, generating alarms and notifications derived from sensor data and, last but not least, for accessing the actual sensor data.

Among the proposed standard service interfaces, the Sensor Observation Service (SOS) [11] is gaining importance in the development of environmental information systems; as it is being considered as potential standard for INSPIRE observation data download service. A SOS enables standardized access to collections of observation data generated by different processes, which are in most cases physical sensors. Observations inside a SOS are organized in Offerings. Each observation of each offering has a value (such as 15ºC) of an Observed Property (such as Air Temperature) for a given time instant. Besides, the observation references the domain specific entity to which the property applies, called Feature of Interest (FOI) (for example a Meteorological Station) and the Process used to obtain the value, commonly a physical sensor (for example a temperature sensor). The Observations and Measurements (O&M) specification provides a data model [20] and relevant XML encoding [19] for those observation collections. Finally, SensorML [5] provides an XML language to describe Processes.

There exist other standards that may be used to access observation data like Web Feature Service (WFS) or Web Coverage Service (WCS), but only the SOS captures observation data semantics.

1.2 Problem description

The main problem to be solved in this Thesis is the integrated access to various observation data sources through a single SOS service interface. This general problem shows up two main facets, one derived from the heterogeneity of the data sources and another caused by semantic conflicts that arise during the integration process, as it is illustrated in Figure 1.1 and discussed in depth below.
Data sources are in general highly heterogeneous due to various reasons.

- *Data modeling paradigm*: Many different types of sensing devices are used. In-situ devices generate local observations, i.e. of a single FOI at the same location of the sensor. Examples of these are sensors of meteorological stations (static platform) and sensors of radio sounding (mobile platform). Generally, those sensors generate entity-based data that fits a classical Entity/Relationship (ER) modeling paradigm, and therefore, their data is generally managed with relevant relational data management technologies. On the other hand, remote devices generate observations of FOIs located at a certain distance from the sensor. Examples of these are the various types of surface radars (static platforms), such as the weather surveillance radars (WSR) used to locate precipitation and the High Frequency Radars (HF Radars) used to measure sea currents. Airbone and satellite (mobile platforms) sensors are also examples of remote devices. Generally, those sensors generate series of geospatial arrays, which cannot be efficiently managed with classical relational technologies.

- *Data model*: Even if we restrict to the ER data modeling paradigm, the ER model used by different data sources to represent their observations is generally different. An example of this are the three different data models shown in Figure 1.1 to represent meteorological station, radio sounding and CTD observation data. Notice for example, that property names (temperature, pressure, etc.) are represented by attribute names (relational meta-data) in the radio sounding data source, whereas they are represented by values of a text data type (relational data) in the meteorological station data source.

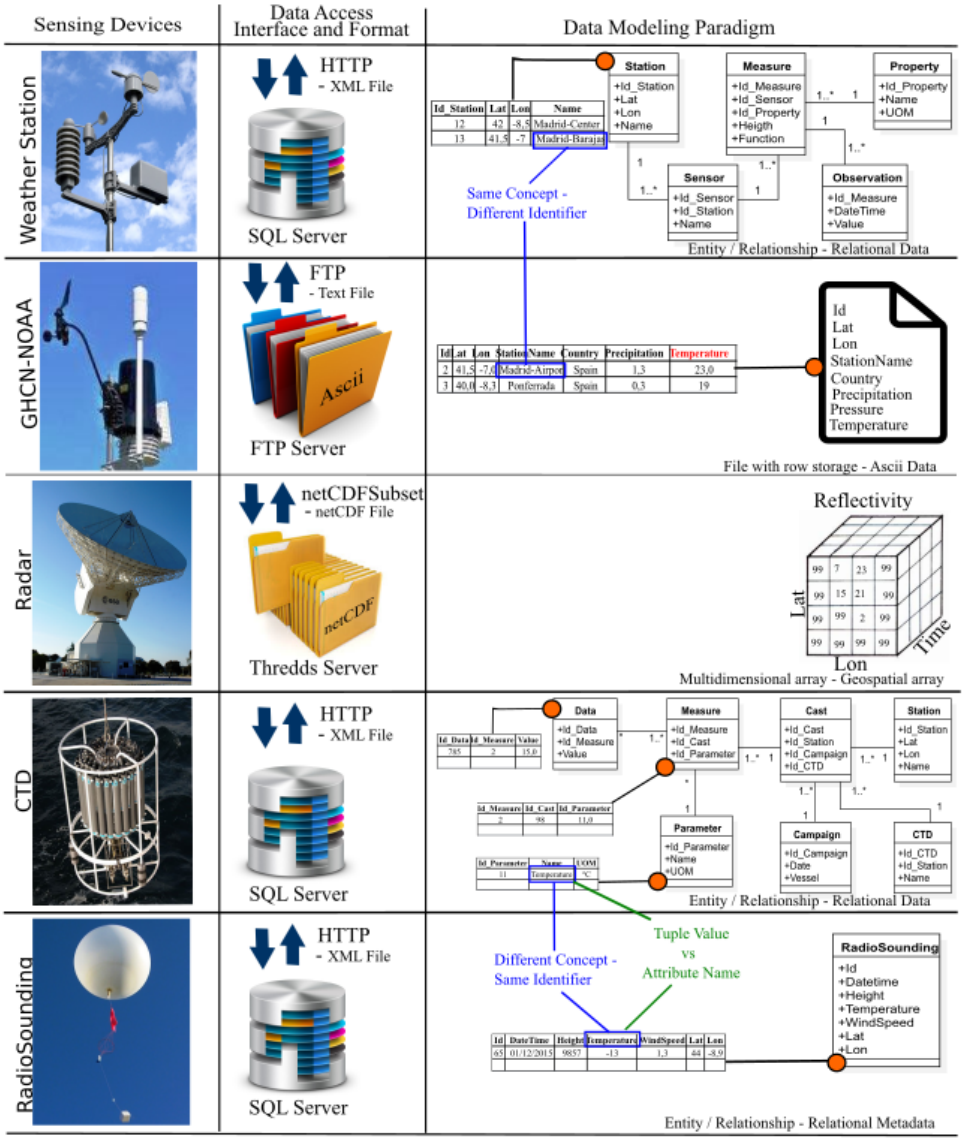


Figure 1.1: Heterogeneity and semantic conflicts

– *Data access interfaces and formats*: Different data format and data access interfaces may be used by different data sources, even if they record similar data. Thus, for example, currently, the data of the GHCN international meteorological station network may be downloaded from the NOAA data servers through a File Transfer Protocol (FTP) in a specific text format. On the other hand, data from the meteorological station networks of MeteoGalicia have to be downloaded through a web form in XML format.

Many different semantic conflicts may arise during the data integration process. In general, those conflicts may be caused by one of the two following reasons.

– Same concept appears in various data sources with different identifiers. A first example of this is the existence of the same meteorological station (FOI) with different identifiers in the networks of MeteoGalicia and AEMET. Another example is the existence of the property "Air Temperature" with different names in the two above data sources. Notice that the data integration process has to solve the above conflicts to enable the identification of redundant observations recorded in both data sources.

– Different concepts have the same identifier in different data sources. An example of this would be the use of the keyword "Temperature" to denote different properties in different data sources (soil temperature, air temperature at 10 cm, air temperature at 10 meters, sea surface temperature, water temperature of a river, etc.). In this case, the data integration process has to add the context of each data source to the identifiers to avoid ambiguity.

1.3 Motivation

Various different solutions, from the software architecture point of view, may be developed for the above problem, as it is shown in Figure 1.2. A straightforward solution would be given by an ad-hoc implementation on the client side (see Figure 1.2(a)). The main drawbacks would be, first the lack of generality and second the need to implement complex clients by specialized users. From the server side there are two clear broad alternatives for data integration. A first solution would be a data warehouse approach (Figure 1.2(b)). The data warehouse must support both entity-based and array-based data modeling paradigms. Extract Transform and Load (ETL) processes must be implemented for each data source, where the heterogeneity and semantic conflicts have to be solved. A second alternative would be a data mediation

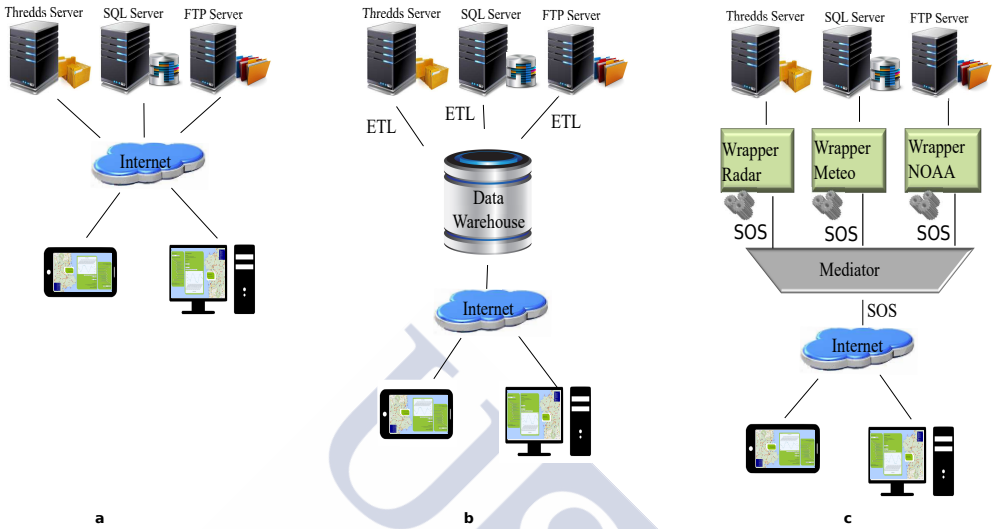


Figure 1.2: Different solutions for accessing to heterogeneous data

approach (see Figure 1.2(c)). In this case, the SOS requests are transformed, in query time, to suitable queries for each data source. Mediator/Wrapper architectures are used in this case. A wrapper is implemented for each data source to deal with all the heterogeneity issues. The mediator is in charge of the resolution of semantic conflicts.

A data warehouse is in general more efficient in terms of query response time and additionally it provides some kind of data replication. However, a data mediation approach can access latest generated data on the fly and it reduces the complexity of the whole system, reducing this way both the installation and administration costs. In general, complex hybrid solutions that combine data warehouses with data mediators should be used in real scenarios. If we restrict to observation data integration, current SOS implementations follow a data warehouse approach and they restrict to entity-based data generated by in-situ devices and recorded with relational technologies. Integrated entity-based and array-based data storage and analysis is only supported in [65], however, relevant ETL processes are not provided. Finally, to the best of my knowledge, apart from the one presented in this Thesis, there is not any semantic mediation SOS implementation.

1.4 Objective and contribution

The main objective of this Thesis is the design and implementation of a novel semantic data mediation framework that enables the access to any kind of environmental observation dataset, including both relational data sources and spatio-temporal arrays. The framework uses Mediator/Wrapper architecture. The mediator uses semantic technologies to take advantage of the knowledge provided by domain experts during the resolution of semantic conflicts. Generic wrappers for spatial databases and spatio-temporal array datasets published through the NetCDF Subset standard interface are also developed to ease the incorporation of these two dominant data source technologies.
The contributions of the Thesis are resumed next as follows:

- Virtual data integration solution for sensor observation data [62] (see chapter 3). It provides a data mediation approach to provide integrated access to entity-based and array-based sensor observation datasets. This initial solution is limited in the resolution of semantic conflicts.

- Semantic mediation of sensor observation services [61] (see chapter 4). The incorporation of semantic technologies to the above data mediation approach enables the incorporation of complex relationships between data source concepts defined by the domain expert, which are used during the data integration process to solve semantic conflicts.

- Development of generic wrappers for both entity-based and array-based datasets [47] (see chapter 5). A generic wrapper for in-situ observation data recorded in a spatial database is developed, which provides an SOS interface on top of any spatial database with any underlying ER model. A generic wrapper for remote observation data accessed through NetCDF Subset standard services is also developed. The above wrappers provide a generic yet efficient means to provide SOS access to those two broadly used technologies.

1.5 Outline of the thesis

The structure of the thesis is organized as follows. Chapter 2 presents a general state of the art together with previous research work and related projects. We start by presenting the Sensor Web Enablement (SWE) initiative and some of its specifications. Also, we present the relevant

standards that are defined in it, specially the standard Sensor Observation Service (SOS). In the following section of the chapter we present the current state of the art of data integration solutions, environmental and geographic semantic web and finally the overview of semantic mediation is briefly presented.

In Chapter 3 a virtual integration of heterogeneous observation data sources through a SOS standard interface is described. How global and local concepts are mapped by the virtual data integration approach undertaken in the frameworks mediator is presented at the beginning of the chapter. Then, the components of the software architecture are described in the second section. Finally, in the third section, the evaluation of the framework and its validation in real scenarios related to meteorological and oceanographic data is shown and the conclusions ended the chapter.

Chapter 4 provides a detailed description of the semantic data mediation between heterogeneous environmental observation datasets. The data mediation architecture is described in the first section. The second section illustrates the contents of data source ontologies. The definition of data integration knowledge is illustrated in the third section. The fourth section describes the implementation of the semantic data mediation process. Qualitative and performance evaluation results are discussed in the fifth section and the last section concludes the chapter with the conclusions.

Chapter 5 describes a solution for the development of generic semantic data access wrappers for observation datasets generated by in-situ and remote sensing devices. Those wrappers are key components of data mediation architecture designed for the semantic integrated publishing of observation data described in Chapter 4. The first and second section describes the design and implementation of the in-situ and remote sensor observation data wrapper, respectively. The evaluation of the performance of both wrappers and a required optimization strategy are discussed in third section. The last section of the chapter ends with the conclusions.

The findings of the research undertaken in this thesis and issues of future work are summarized in Chapter 6.

CHAPTER 2

BACKGROUND AND RELATED WORK

This chapter provides a description of the background and a previous research work related to the semantic integration of heterogeneous environmental data. To achieve this, technologies arising from the OGC SWE initiative and the W3C Semantic Web must be combined, the former to provide standardized web interfaces and encoding and the latter to tackle the resolution of semantic conflicts. The above combination of technologies is called Semantic Sensor Web (SSW) and was first discussed in [66]. Research challenges for the SSW are identified in [9, 18]. Related to the present work, semantic sensor data integration is identified as a challenge in both papers. Additionally, the solution proposed in the present Thesis also contributes to the rapid development of applications, identified as a challenge in [18], as it enables the rapid development of wrappers for relational and array-based datasets, which provide a common SOS interface and O&M based format.

The remainder of this chapter is outlined as follows. Section 2.1 describes the SWE initiative and relevant standards. Data integration solutions are described in section 2.2 from a general point of view. The basis of semantic technologies are described in Section 2.3. Sematic web technologies and pieces of work related to their application in the environmental and geospatial domain are described in Section 2.4. Section 2.5 describes semantic sensor web challenges, technologies and efforts made in this scope. In Section 2.6 a overall description of the state of art in semantic mediation of environmental datasets is given. Based on the state of the art described in previous sections, the motivation of the Thesis is outlined in the last section.

2.1 Sensor Web Enablement (SWE) initiative

This initiative proposes various standard web service interfaces and data formats that enable interoperable access to sensor data in environmental data infrastructures. These standards are well known means to acquire, catalog and integrate environmental observation data from various sources, as it has already been reported [17]. As OGC's Sensor Web Enablement group defined literally "SWE standards enable developers to make all types of networked sensors, transducers and sensor data repositories discoverable, accessible and useable via the Web or other networks". Thus, the initiative aims to integrate observations provided by either in-situ or remote sensors. Such observations have a location, and the location of both types of sensors is highly significant for many applications of use.

Among the proposed standards, and specially relevant in the domain of geospatial observation data, are the Sensor Observation Service (SOS) [11], that is internationally assumed as the interface to access observation data sources, the Observations and Measurements (O&M) that defines both a data model [20] and an XML encoding [19] for environmental observation data, and Sensor Model Language (SensorML) [5] that provides an XML language to describe *Process*. Various pieces of work have been reported in the literature that make use of SWE standards ([17, 76, 14, 30, 70, 41]), which is a good indication of the widespread use of these technologies. Besides, a reference Sensor Service Architecture (SensorSA)([73]) was proposed in the scope of the SANY (Sensors Anywhere) project. More details are given below in following subsections.

2.1.1 Observation and Measurements (O&M)

The OGC O&M standard specifies a general data model and XML encoding for the observations. An Observation provides a *Value* (such as 15^{o}C) of an *ObservedProperty* (such as air temperature) of a specific entity to which the property applies, called *Feature of Interest* (for example a meteorological station), which are generated by some observation *Process* (commonly a physical sensor like a temperature sensor), that is, an event with a result which has a value describing some phenomenon. Beyond the value and the references to Process, Property and FOI, the observation must register also the time when the value applies to the FOI (*PhenomenonTime*) and the time when the Process was executed (*ResultTime*). Optionally, a unit of measure (*UOM*), quality information and some other parameters might also be registered. One such parameter might be the specific location where the observation was

performed (*SamplingGeometry*) inside the Feature of Interest. As an example, to determine phytoplankton biomass (Property) at the Cantabrian Sea (FOI), first, a sampling of water is obtained at the specific point (samplingGeometry) at a specific time instant (phenomenonTime). Next (resultTime), the sampled water is analyzed in a laboratory to determine the phytoplankton biomass value (grams per liter). The depth of the sampling (additional parameter) is also recorded. The Process to analyze the water is typically a combination of a sensing device with some processing. For example, to determine the "sea surface temperature" (Property) of the "Gulf of Mexico" (FOI), the combination of a temperature sensor with aggregation and spatial interpolation operations is performed. More specifically, this standard defines XML schemas for observations, and for features involved in sampling when making observations. These provide document models for the exchange of information describing observation acts and their results, both within and between different scientific and technical communities.

2.1.2 Sensor Observation Service (SOS)

An OGC SOS [11] provides access to a collection of observations modeled with the O&M OGC specification [20] (described in the previous section 2.1.1). Such observations are organized in possibly overlapping collections called *Offerings*, which are defined as a logical grouping of observations offered by a service that are related in some way. The parameters that constrain the offering should be defined in such as a way that the *Offering* is "dense" in the sense that requests for observations that are within the specified parameters should be unlikely to result in an empty set. For instance, an offering could be "observations of humidity and temperature in the northern coast of Spain during the last three months". More specifically an *Offering* is constrained by a list of *FOIs*, a list of *process*, a list of *observed properties*, a *spatial extension* and a *temporal range*. Different versions of SOS standard interface have been released during this research work (v1.0 in 2007 and v2.0 afterwards in the middle of my research, 2013). As a consequence, this dissertation uses the version 1.0 as a basis. The standard has three mandatory operations (detailed below), together with various optional ones.

Operation GetCapabilities

This operation obtains metadata of both the service and each of its available *Offerings*. The response of this operation, also called "Capabilities document" provides metadata about

the service identification, service provider, operations metadata together with two sections that are specific for the SOS, namely the Filter_Capabilities and the Contents section. The FilterCapabilities section is used to indicate what types of query parameters are supported by the service. These capabilities refer to the parameters of the GetObservation operation that is described below. The *Contents* section is used to show which *offerings* can be queried. Each observation offering of each *process* is constrained by its temporal and spatial extension and the list of referenced *properties* and *FOIs*.

Operation DescribeSensor

This operation obtains a *SensorML* description of a given *Process*, that is, an XML codification to describe a sensor or process. This operation is specially designed to request detailed sensor metadata. In the standard, the observation process can be classified in different ways based on the type of sensor and on the characteristics of the platform. Such classification is outlined as follows:

– Physical vs Non-Physical: The process can be a physical sensor or a procedure to determine a value.

– In-Situ vs Remote: Based on the sensor characteristics, this may measure either in-situ or remotely.

– Fixed vs Mobile: Based on the dynamics of the platform, a sensor system may be fixed or mobile.

Thus, a remote sensing device installed in a satellite is classified as “mobile remote”. See Figure 2.1(a). However, if this sensing device measures attached to a radio sounding (see Figure 2.1(b)), then it is classified as “mobile in-situ”. Similarly, the Figure 2.1(c) shows both a vertical and horizontal *Acoustic Doppler Current Profiler* (ADCP). Such devices are examples of “fixed remote” sensors. Finally, the Figure 2.1(d) depicts a weather station with several sensors installed that measure “fixed in-situ”.

Operation GetObservation

This operation retrieves the observations of a given *Offering* that matches a set of specified criteria defined by the user in the request. Such criteria include one or more *Property* ids, zero

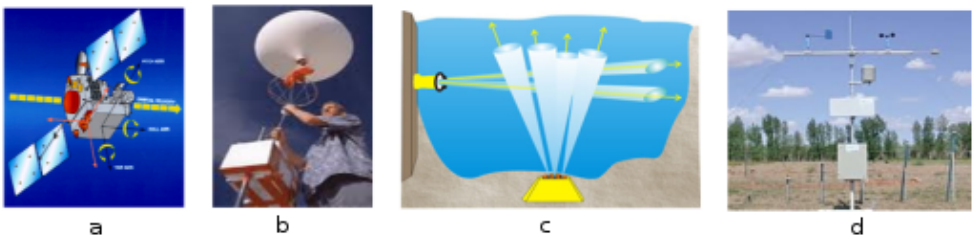


Figure 2.1: Sensors classification and examples

of various *Process* id, zero or various *FOI* ids, optional temporal filters and an optional spatial filter. These elements constrain the observations to be retrieved from a Sensor Observation Service. The response to a GetObservation request is in general a collection of observations encoded in O&M XML format.

2.1.3 SOS implementations

Many observation servers implementing SOS interfaces have been proposed in the literature [40, 44, 15, 29], all of them following a data warehouse approach. Most of the currently available implementations are tailored to in-situ observations recorded in relational DBMSs [55, 22, 48], and all of them following a data warehouse approach. Few support raster observation data sources [8]. Representative examples are provided by the well known open source initiatives described below:

- 52°North SOS [55]: It is a Java-based implementation, INSPIRE compliant that uses a relational storage. Therefore, this open source software adds some extra filters to access latest recorded data and support the whole set of operations. 52°North has implemented both published versions of SOS so far (it supports both SOS 1.0 and SOS 2.0 specifications). An interesting initiative is the project called "Sensor Data Access for Rasdaman", that aims to connect Rasdaman as an alternative data storage back-end to the 52°North SOS and explore storing array (raster) observations in the SOS.
- Deegree SOS [22]: It is a implementation of the SOS 1.0 specification. Is is OGC compliant and only supports the core profile operations plus *GetFeatureOfInterest*. Deegree SOS supports both SQL and binary data backends unlike 52°North SOS.

– MapServer SOS [48]: It is an open source SOS written in C that is part of the Mapserver platform. This SOS is quite different compared with the previous two since it follows the Mapserver philosophy and a regular MapServer mapfile has to be expressly defined with mandatory parameters and metadata entries. This implementation only supports the core profile operations (GetCapabilities, DescriberSensor and GetObservation).
– Oostethys SOS: It is an open source implementation developed in collaboration among SCOOP, the Southeastern Universities Research Association Coastal Ocean Observing and Prediction Program, and the Marine Metadata Initiative (MMI), gathering as "OOSTethys". This implementation support mainly NetCDF files (very common in this scope in USA), or a database or even CSV text files. The initiative is open and everyone can contribute with code in Java, Perl, Python, etc.

The first three solutions do follow a data warehouse approach (see Section 2.2) using relational storage. Nevertheless, the fourth also follows a data warehouse approach, but using NetCDF files as storage. A performance evaluation of these implementations has been reported in [59].

2.2 Data integration solutions

The simplest solutions provide gateways to access various Database Management Systems (DBMS). Such solutions face transparency problems derived from the fact that users must be aware of which pieces of data are located in each data source. Some solutions are described below.

2.2.1 Data warehouse

In this first solution, denoted as data warehouse ([34, 39]), all the required data is recorded under a common multidimensional data model, which is fed by Extract, Transform and Load (ETL) processes. Data warehouses are widely used as Business Intelligence (BI) solutions for decision support in organizations. They require additional storage infrastructure and if near-real time requirements are present then ETL may have to be performed too frequently. Event-driven architectures have been proved to be successful to alleviate these problems ([52]). Spatio-temporal data warehouse solutions are surveyed in [25] and a brand-new data warehouse approach for sensor observation data analytics is proposed in [75].

2.2.2 Data mediation

This second approach is connected with federated databases and virtual integration. McLeod and Heimbigner [31] were among the first to define a federated database system in the mid 1980s. We might say that a federated database system is a type of meta-database management system (DBMS), which transparently maps multiple autonomous database systems into a single federated database or virtual database. Such virtual database is a composite of all constituent databases in a federated database system. There is no actual data integration in the constituent disparate databases as a result of data federation. The three important components of a *Federated Database System* (FDBS) are autonomy, heterogeneity and distribution [68].

Virtual data integration arises in the area of FDBS [68]. Subsequent solutions attempted the integration of structured data sources through the web [43, 13], using Mediator/Wrapper architectures [79]. Such architecture was already proposed in [28] for the integration of spatial entities and coverages and in [6] for the integration of OGC WFS compliant data sources. Moreover, general issues of spatial virtual integration are discussed in [23]. Key challenges in virtual data integration were identified in [42]:

- The definition of a global virtual data model in which all possible local data models may be integrated.
- The resolution of syntax, semantic and system conflicts during the integration process.
- The definition of query reformulation algorithms that generate execution plans for global queries that efficiently access the data sources.
- The definition of relationships between global and local model elements.

Virtual data integration does not require ETL, however, constructing the virtual global data model on query time may lead to poor response times. An overview of the advantages and disadvantages of different decision support architectures is given in [2], where virtual integration is referred as federated architectures. In practice, however, complex architectures that combine data warehouses with virtual integration are usually developed to adapt to the specific needs of each organization.

2.3 Semantic knowledge representation and management

Technologies related to *Ontologies*, their representation and management are briefly described in the following subsections.

2.3.1 Ontologies

The ultimate purpose of the ontologies is to enable machines to exchange information effectively and efficiently. They can provide formalisms and structure information allowing a degree of automated reasoning. Gruber [54] created one of the most cited definitions of ontology according to Genesereth and Nilsson [69] in the field of computing, "an explicit and formal specification on a shared conceptualization". In [4, 58] Borst defined an ontology as a "formal specification of a shared conceptualization". This definition additionally required that the conceptualization should express a shared view between several parties, a consensus rather than an individual view. Also, such conceptualization should be expressed in a machine readable format. In 1998, Studer et al. [71] merged these two definitions stating that: "An Ontology is a formal, explicit specification of a shared conceptualization." An ontology provides a common vocabulary of a domain of knowledge and defines the meaning of the terms and the relations between terms in different levels of formality [71]. The components of ontologies are classes (concepts), relations, axioms and individuals. The classes or concepts in the ontology represent any entity that provides some information and contains properties. Relations represent interactions between classes, such as inheritance (usually called taxonomic relation), and individuals are concrete instances of a particular class. Ontologies are formally encoded using knowledge representation languages. The Web Ontology Language (OWL) [50] is a broadly used one that is defined by the W3C upon the Resource Description Framework (RDF) [63].

2.3.2 Resource Description Framework (RDF)

RDF is a format to encode data of *Resources* available in the web. Broadly, RDF enables the definition of statements of the form

```
(subject predicate object),
```

where *subject* is a *Resource*, *predicate* is a *Property* of the *subject* and *object* is either another *Resource* or a data literal. Each *Resource* and *Property* is identified by a Internationalized Resource Identifier (IRI). An example of RDF statement is

```
(john hasName "John"),
```

which states that the literal "John" is the value of the property identified by the IRI hasName of the *Resource* identified by the IRI john. An RDF dataset is modelled with a graph, where *Resources* and literals are the nodes and *Predicates* are the edges.

RDF Schema (RDFS) [21] is a semantic extension of RDF that provides a data modelling vocabulary. Such vocabulary is a collection of RDF *Resources* and *Properties* that enable the definition of classes of resources (individuals) and class hierarchies. As an example, the following RDF statements define that resource http://myserver/john as and individual of class http://myserver/employee, which is a subclass of http://myserver/person.

```
(person rdf:type rdfs:Class)
(employee rdf:type rdfs:Class)
(employee rdfs:subClassOf person)
(john rdf:type person).
```

2.3.3 Web Ontology Language (OWL)

OWL increases the expressive power of RDFS with additional constructors, which include the following knowledge representation capabilities. The definition of *Object Properties*, *Data Properties*, transitive, symmetric and functional *Properties*. The definition of a *Property* as the inverse of another. The definition of new classes by specifying restrictions over properties. The definition of complex classes as unions, intersections and complements of other. The definition of classes by the enumeration of their instances. The definition of mappings between classes and individuals.

Various different syntaxes have been formalized for OWL. RDF/XML [1] is used by the present framework to record OWL files, however Manchester Syntax [2], which is more compact, will be used in the remainder of this paper.

2.3.4 SPARQL

Regarding semantic knowledge management, various technologies are currently available in the semantic web area. SPARQL Protocol and RDF Query Language enables the declarative query of RDF graphs, using graph pattern expressions. This language is used in the present thesis to access and extract information from the required OWL ontologies.

[1] https://www.w3.org/TR/rdf-syntax-grammar/
[2] http://www.w3.org/TR/owl2-manchester-syntax/

2.4 Environmental and geospatial semantic approaches

Semantic web technologies have already widely been applied in the areas of geospatial and environmental data management. Thus, the semantic enablement of Spatial Data Infrastructures (SDIs) is discussed in [38]. State of art and research perspectives related to geospatial semantic data management are provided in [56]. More specifically, [49] proposes an ontology design pattern to model the quantification over types. A new architecture for semantic gazetteers is presented in [12]. A semantically enabled environmental monitoring framework is described in [77], which uses foundational ontologies to support environmental regulation violations and relevant human health effects. A new extensible architecture for the above framework, which is based on the use of semantic technologies and that eases the incorporation new data sources and domains is proposed in [57]. Finally, a plug-in that extends the ontology framework Protégé with a semantic similarity measure is described in [51].

A semantically enabled environmental monitoring framework is described in [77], which uses foundational ontologies to support environmental regulation violations and relevant human health effects. A new extensible architecture for the above framework, which is based on the use of semantic technologies and that eases the incorporation new data sources and domains is proposed in [57].

A well-known ontology in the scope of environmental data modeling is the NASA Semantic Web for Earth and Environmental Terminology (SWEET) [60]. It is a pack of ontologies written in the OWL ontology language and are publicly available. Such ontologies consist of nine top-level concepts/ontologies (Some of the next-level concepts are shown in the Figure 2.2). SWEET is a middle-level ontology, most users add a domain-specific ontology using the components defined here to satisfy end user needs. It contains over 6000 concepts organized in 200 ontologies. Top-level concepts include *Representation* (math, space, science, time, data), *Realm* (Ocean, Land Surface, Terrestrial Hydroshere, Atmosphere, etc.), *Phenomena* (macro-scale ecological and physical), *Processes* (micro-scale physical, biological, chemical, and mathematical), *Human Activities* (Decision, Commerce, Jurisdiction, Environmental, Research) and *Property* (Binary Property, Categorical Property, Quantity , Ordinal Property).

2.5 Semantic Sensor Web (SSW)

The linking and combination of elements from *Semantic Web* technologies with sensor networks and technologies from the OGC SWE initiative has been called *Semantic Sensor Web*

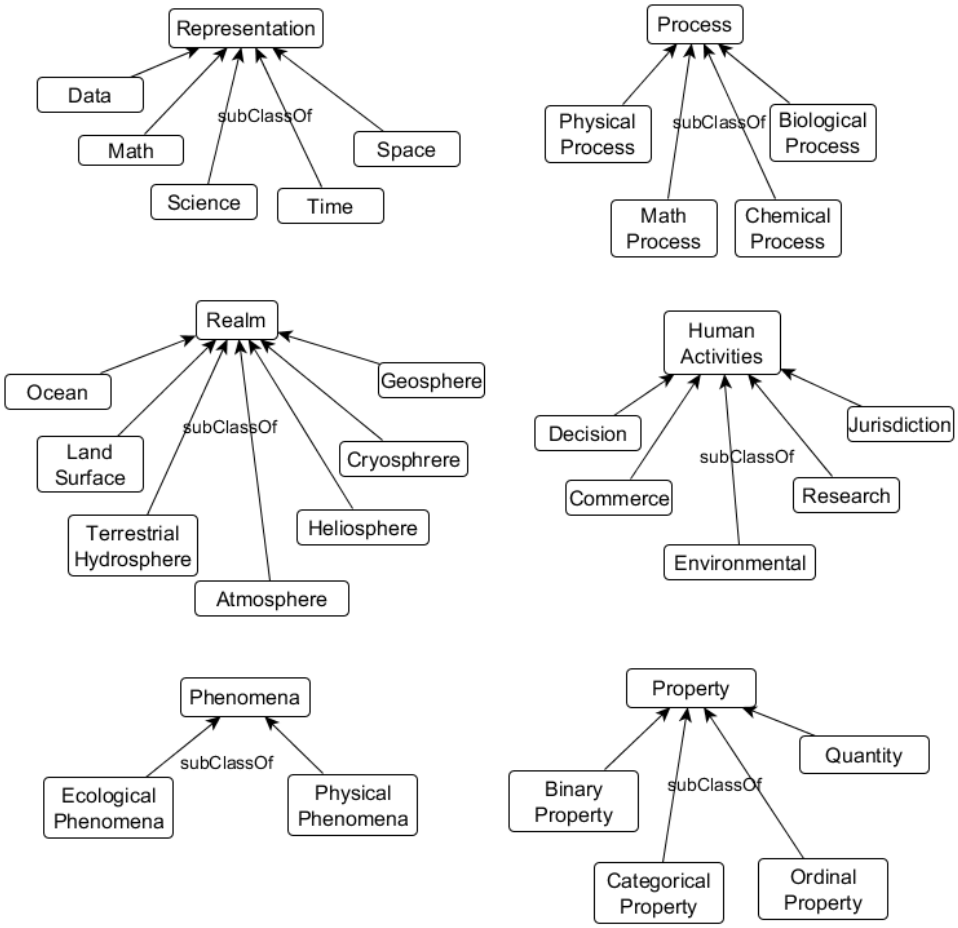


Figure 2.2: The SWEET main ontologies graph

(SSW), and was first discussed in [66]. More recently, the SSW and the Linked Sensor Data were identified as future work topics towards a new generation of SWE standards [9]. In particular, semantic sensor data discovery and integration were identified as major challenges to be overcome. According to [18] there are five challenges for the Semantic Sensor Web: 1) the first is about the abstraction level of the data extraction, process and management; 2) Quality of Service of sensor data; 3) integration and fusion of sensor data; 4) identification and location of relevant sensor-based data sources; and 5) rapid development of applications. Some efforts [33, 24] were made on the alignment of different ontologies with the objective to fulfill some of these challenges. There are several ontologies to organize the concepts and relations of the domain. In 2008 a framework named Semantic Sensor Web [67] was proposed and one of the main features of this framework is the use of ontologies. A well-known ontology in the scope of semantic sensor web is the W3C Semantic Sensor Network (SSN) [16]. Such ontology describes sensors and observations defined OWL2. It can describes sensors in terms of capabilities, measurement processes, observations and deployments. Note that everything turns around the concept *Sensor*. The SSN ontology has begun to achieve broad adoption and application within the sensors community. The main classes of the SSN ontology have been aligned with classes in the DOLCE Ultra Lite (DUL) [36] foundational ontology to facilitate reuse and interoperability. However the rest of existing ontologies in sensor networks domain do not propose any kind of alignment with any other, making interoperability between them difficult.

Some more efforts are cited below. An integrated water resource decision support application was proposed in [80], where some of its contributions include i) re-using and matching the W3C Semantic Sensor Network (SSN) ontology and other popular ontologies for heterogeneous data modeling in the water resources application domain. The research platform called Sense2Web [3, 72] describes a platform to publish Semantic Sensor Network data and to link the data to existing resources on the Web. The linked Semantic Sensor Web data platform supports the publication of extensible and interoperable resource (i.e. sensor network and service resources) descriptions and observation and measurement data in the form of linked data. Sense2Web also supports the association of different sensor data to resources described on the Web of Data. The paper focuses on publishing linked data to describe sensor data and sensor network resources descriptions and link them to other existing resources on the Web.

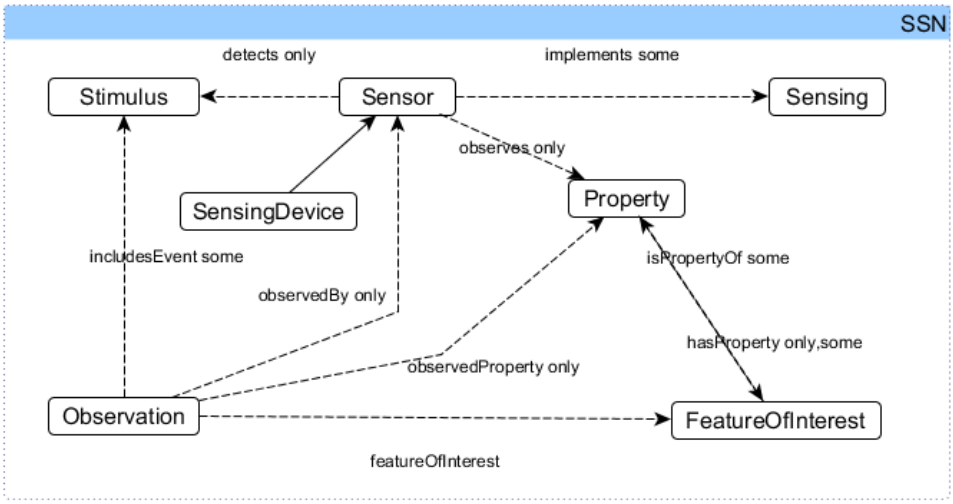


Figure 2.3: The SSN ontology, key concepts and relations

2.6 Semantic mediation

Various pieces of work have dealt with the problem of semantic mediation among scientific, geospatial and environmental data sources. An approach called *Model Based Mediation* is proposed in [46] for the integration of scientific data sources. Each data source exports raw data and conceptual models with explicit semantics. The mediator combines the data source conceptual models with auxiliary domain knowledge sources, called *glue knowledge*, which includes relationships between concepts and unions and intersections of concepts.

Semantic Mediation between geospatial data sources is a piece of functionality that should be provided by services of the brokering approach introduced by the EuroGEOSS project and adopted by the GEOSS Common Infrastructure [53].

The mediation approach for environmental knowledge representation is discussed in the review reported in [74]. Inputs and outputs of processes in scientific workflows have to be enriched with knowledge representation, i.e., they must be semantically annotated with concepts from relevant ontologies. Reasoning may be next applied to check for compatibility between inputs and outputs. Semantic mediation between various oceanographic vocabularies is discussed in [26].

If we restrict to observation data, in [7], the authors define an extension of standard conceptual modeling approaches with new constructs for the incorporation of observation semantics. The result model can be used to annotate data sets with observation semantics, enabling them to be semantically integrated. Semantic annotation of SensorML [5] documents is performed in [10] to enable the semantic registration of sensing systems in SOS services. The annotation process establishes relationships between concepts in SensorML and O&M. In particular, entities, stimuli and properties are mapped respectively to FOIs, sensor inputs and sensor outputs. Another example is OntoSensor ontology[64, 1]. It was intended as a general knowledge base of sensors for query and inference. OntoSensor covers a wide range of concepts, class definitions and individuals and can describe the platform a sensor is attached to, but can only describe generic part-of relations. This ontology provides high expressiveness for data and the ability to organize sensors into a hierarchy of sensing concepts.

A semantic SOS (SemSOS) has been designed and implemented [32] as an extension of a well known SOS open source tool [55]. Raw sensor data is first semantically annotated and next transformed to RDF to be recorded in a knowledge base. SOS requests are next transformed to SPARQL queries over the stored RDF. SOS responses are encoded in semantically annotated O&M and SensorML documents.

The importance of semantic mediation has already been stated from a general purpose point of view [78] and specifically from the geospatial community [53].

2.7 Motivation of the Thesis

Despite of the above and although virtual data integration and semantic web technologies have widely been applied in the areas of geospatial and environmental data management, there are no SOS implementations which overcome the limitations of the relevant approaches that were identified throughout this chapter. None of the above implementations deal with the integration of various observations data sources, both relational data and multidimensional arrays. Moreover, none of the reported approaches performs semantic integration of O&M concepts, so this is the main motivation of this dissertation.

CHAPTER 3

VIRTUAL INTEGRATION OF SENSOR OBSERVATION DATA

As it was already stated at Chapter 1, the main challenges to be solved in order to achieve integrated access to environmental observation data sources are related to both heterogeneity and semantic conflicts. This chapter focuses on heterogeneity related problems. In particular, with those related to use of different data modeling paradigms and specific data models and consequently with the use of different data access interfaces and formats. To tackle those problems, client-side, data warehouse and data mediation solutions may be undertaken, as it was described at Chapter 1. Besides, the OGC SOS standard interface is the appropriate one to solve interoperability issues. Based on the above, in this chapter the design, implementation and evaluation of an initial virtual data integration solution for spatial observation data sources is proposed. The framework provides an integrated SOS-based view of entity-based and array-based data sources. To achieve this, it uses a Mediator/Wrapper architecture, where wrappers adapt the data sources to the O&M data model and SOS interface, and the mediator enables the definition of global *Offerings* in terms of local ones.

The remainder of the chapter is organized as follows. Section 3.1 defines the data integration model used by the mediator. The Mediator/Wrapper architecture is described in Section 3.2. The evaluation of the framework and its validation in two real scenarios is shown in Section 3.3. Finally, Section 3.4 concludes the chapter.

3.1 Observation data integration model (ODIM)

As it was already explained in chapter 2, the data that is published through a SOS 1.0 must be organized into possible overlapping collections of observations called *Offerings* , in order to minimize the probability of issuing queries with empty results. The *Observation Data Integration Model* (ODIM), described in this section, enables the definition of the global *Offerings* of the framework in terms of queries to the *Offerings* defined in the available SOS compliant data sources. In other words, ODIM makes possible the definition of global Offerings over local ones to achieve the integration of observations from different data sources. Additionally, the model provides also mappings between global concepts of the framework and relevant concepts in each of the available data sources. One of the most striking aspects of ODIM is the possibility to define new global *Observed Properties* whose semantics subsume one or various *Observed Properties* of one or various data sources. For instance, a global *Property* "Temperature" could be defined to subsume different air temperature properties measured at different elevations in meteorological and oceanographic station data sources. Contrary to the above, the approach is more limited for *Processes* and *FOIs*, since they are assumed to be distinct in each data source and global ones cannot be defined in the ODIM. This simplification was assumed due to the fact that it is not common in real scenarios to have various data sources generated by the same *Processes* for the same *FOIs*. Figure 3.1 shows the conceptual design of the ODIM in the form of a UML class diagram and Figure 3.2 provides a relational representation of the entities and relationships of a specific reduced instantiation of the model. A *ODIMManager* component of the frameworks architecture (see Section 3.2) provides access to a persistent copy of the ODIM , which was recorded in a relational DBMS in the first prototype implementation of the framework. It is finally noticed that the functionality of the ODIM regarding semantic integration was improved with the incorporation of ontologies in a next prototype, as it is described in Chapter 4.

3.1.1 Global offering

As it is shown in Figure 3.1, each global *Offering* is represented by an instance of the class *GlobalOffering*. A *Universal Resource Identifier* (URI) is used for identification, which may be complemented by a meaningful name and description. The maximum spatial extension of the *Offering* (a rectangle) is defined by property *boundedBy* of type *Envelope* of *Geography Markup Language* (GML). Regarding the maximum temporal extent of the *Offering*, it may

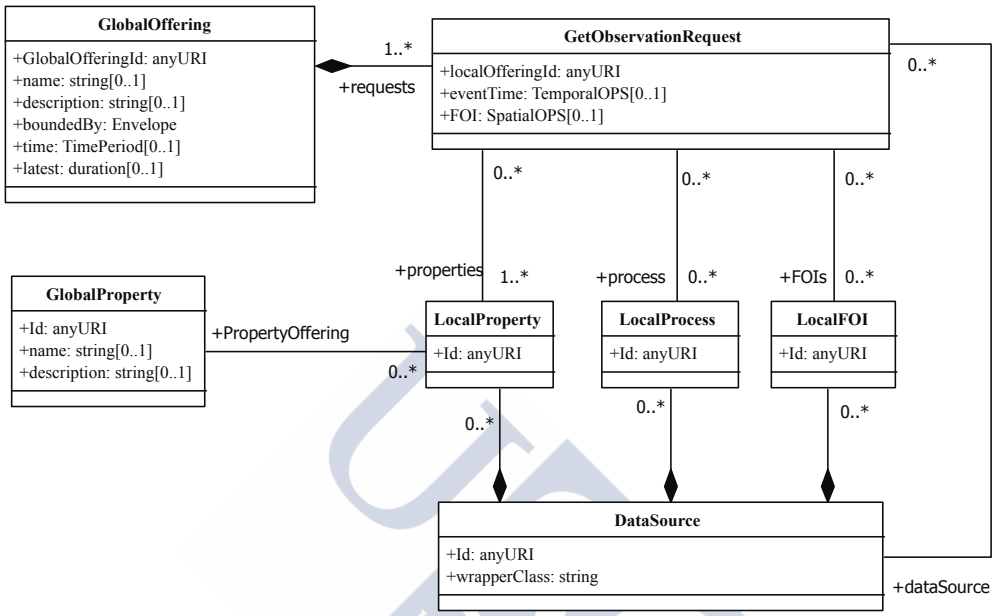


Figure 3.1: Observation Data Integration Model (ODIM) that enables both the definition of global Offerings and mappings between global and local concepts (Observed Properties, Processes and FOIs

be specified by either of properties time or latest. The former defines a time period from a start time instant to an end time instant, as it is defined by type *TimePeriod* of GML. An example of this is "Offering1" in Figure 3.2, which contains observations of temperature for a specific area obtained between 2012-02-01 and 2012-10-31 . The latter defines a dynamic time period whose end time is defined by the current system time and whose start time instant is computed by subtracting the value of the property latest of XML Schema type duration from the current system time. An example of this is "Offering2" in Figure 3.2, which contains observations of temperature for a specific area obtained during the last year, specified in days (365 days).

The observation data of each global *Offering* is obtained by the union of a set of queries over local *Offerings* of the available data sources. Each such query is represented by an instance of class *GetObservationRequest*. The identifier of the local *Offering* of the query is recorded in property *LocalOfferingId* and the data source is specified with an association with an instance of class *DataSource*. Notice that each such instance is identified by a URI and has a property *wrapperClass* of type string that records the name of the wrapper Java class that is used by the framework to access the data source (see frameworks architecture in next Section

3.2). Optional property *eventTime* of class *GetObservationRequest* enables the specification of a temporal filter over the local *Offering* for each query. The format of such a filter is the one specified by the *TemporalOPS* type of SOS 1.0.0 standard. The temporal operators that may be used in such filter are those supported by the capabilities of the framework. Similarly, a spatial filter over the local *Offering* may also be defined with attribute *FOI* of OGC type *SpatialOPS*, in this case an *Envelope*. Notice therefore, that the software wrapper (see Section 3.2) of each data source must implement the temporal and spatial operators supported by the service capabilities.

As an example, the observation data of global offering "Offering1" is obtained by the union of the result of three *GetObservation* requests on three different local offerings, as it is shown in Figure 3.2 ("MeteoStatios_1","MeteoStations_2","OceanBuoys"). The two first requests are performed on two different local offerings of the same data source "es.usc.citius.cograde.sosvdi.meteo", whose *wrapperClass* is "SOSmeteoImp". The last request is performed on a local offering of data source "es.usc.citius.cograde.sosvdi.ocean", whose *wrapperClass* is "SOSoceanImp". Notice that the real spatial and temporal range of each global offering is obtained by the intersection of various spatial envelopes and time periods, respectively. In particular, the real temporal range of global "Offering1" is defined by the intersection of the temporal range defined in class *GlobalOffering* with the result of the union of the real temporal range of the relevant *GetObservation* instances. Each such temporal range is again obtained as the intersection of the temporal range of the request and the temporal range of the relevant local offering.

In addition to temporal and spatial filters, a *GetObservation* request must specify at least one *Observed Property* (LocalProperty) and may specify one or various *Processes* (LocalProcess) and *FOIs* (LocalFOI). Due to a restriction in the SOS 1.0 standard interface, specifying both a spatial filter and a list of FOIs is not possible. Notice that aggregating *GetObservation* requests is more general than just aggregating data source *Offerings*. In fact, spatio-temporal unions of data source *Offerings* may be easily expressed as spatio-temporal unions of *GetObservation* requests; however, the inverse is not true, due to the fact that *GetObservation* requests may include additional restrictions of various types. As a final remark, global offerings must be defined as dense in the ODIM by a domain expert, who must have complete knowledge of the contents and relevant semantics of each data source.

GlobalOffering

GlobalOfferingId	Name	Description	BoundedBy	Time	Latest
Offering1	O_Temperatures	Temperatures from the Rias Baixas estuaries	41, -9.5 43, -8.2	2012/02/01 2012/10/31	null
Offering2	O_Temperatures	Temperatures from the Rias Altas estuaries	42, -10 44, -8	null	365

GetObservation

GlobalOfferingId	LocalOfferingId	EventTime	FOI	DataSource
Offering1	MeteoStations_1	2012/01/01 2012/09/01	42, -10 43, -9	meteo
Offering1	MeteoStations_2	2012/04/01 2013/12/01	40.5, -8.5 42, -7.5	meteo
Offering1	OceanBuoys	2016/01/01 2016/07/01	42, -9,5 43, -8	ocean
Offering2	OceanBuoys	2015/04/01 2016/08/01	43, -9 44, -7	ocean

GlobalProperty

GlobalPropertyID	Name	Description
AirTemperature	Temperature	The temp…
WindSpeed	WindSpeed	The winds…
Precipitation	Precipitation	The preci…

LocalGlobalProperty

LocalPropertyId	GlobalPropertyId
meteo#Temperature155	AirTemperature
ocean#AirTemperature	AirTemperature
meteo#Wind4	WindSpeed
ocean#Precipitation	Precipitation

LocalPropertyOffering

LocalPropertyId	LocalOfferingId	GlobalOfferingId
meteo#Temperature155	MeteoStations_1	Offering1
meteo#Temperature155	MeteoStations_2	Offering1
ocean#AirTemperature	OceanBuoys	Offering1

LocalProcessOffering

LocalProcessId	LocalOfferingId	GlobalOfferingId
meteo#SantiagoEOAS-HMP155	MeteoStations_1	Offering1
meteo#VigoAirport-T3	MeteoStations_2	Offering1
ocean#Borneira-WXT520	OceanBuoys	Offering1

DataSource

DataSourceId	WrapperClass
meteo	es.usc.citius.cograde.sosvdi.SOSmeteoImp
ocean	es.usc.citius.cograde.sosvdi.SOSoceanImp

LocalFOIOffering

LocalFOIId	LocalOfferingId	GlobalOfferingId
meteo#Santiago-EOAS	MeteoStations_1	Offering1
meteo#VigoAirport	MeteoStations_1	Offering1
meteo#VigoAirport	MeteoStations_2	Offering1
ocean#FaroBorneria	OceanBuoys	Offering1

Figure 3.2: Example of an instantiation of the ODIM

3.1.2 Global-Local concept mappings

The ODIM must also define the global counterpart for each of the local *Observed Properties*, *Processes* and *FOIs*. With regard to such global-local mapping of concepts the following initial assumption has been made in the current version of the framework: A *Process* or *FOI* of any data source is assumed to be different from any other *Process* or *FOI* of any other data source. Therefore, the global identification of a *Process* or *FOI* is automatically obtained as the concatenation of the relevant local identification (stored in the table *DataSource*) with a specific delimiter and with the identification of the data source. Thus, domain experts do not have to worry about mappings between global and local *Processes* and *FOIs*. As an example, as it is shown in Figure 3.3, the global identifier of the local *Process* "SantiagoEOAS-HMP155" of data source "Meteo" is "Meteo_SantiagoEOAS-HMP155". Similarly, the global identifier of the local FOI "FaroBorneria" of data source "Ocean" is "Ocean_FaroBorneria".

The above assumption simplifies a lot the definition of the ODIM by the domain expert and it was based on the fact the in most cases *Processes* belong to one single data source. Similarly, *FOIs* use to be dependent of the domain behind a specific data source. Of course, exceptions to the above rules also exist. For example, expensive devices on board of satellites might be shared by different data sources. In such cases, in spite of the fact that observations of different data sources will reference the same *Process* with different identifiers, the domain expert can still define *Offerings* that merge all the observations produced by the same *Process* in all data sources. Overcoming this initial limitation using semantic web technologies is the objective of the semantic integration of SOS data sources approach described in Chapter 4.

On the other hand, it is common that the same *Observed Property* is observed by different *Processes* in different data sources. See Figure 3.2 and Figure 3.3 that show examples of the global-local mapping of concepts during the integration of two data sources, one of meteorological stations and another of oceanographic stations. Due to the above, the global *Observed Property* that corresponds to each local *Observed Property* of each data source must be specified by the expert in the ODIM. Global *Observed Properties* of the framework are represented by instances of class *GlobalProperty* and the corresponding local *Observed Properties* are specified through the association with class *LocalProperty*. Notice that this approach is still limited compared with the one described in Chapter 4, since the semantic relationship between global and local properties (equivalent, subsume, etc.) is not declared in the present approach, in fact, it is assumed that the global property semantically subsumes all the related local properties. As an example (see Figure 3.3, it is assumed that global property "AirTem-

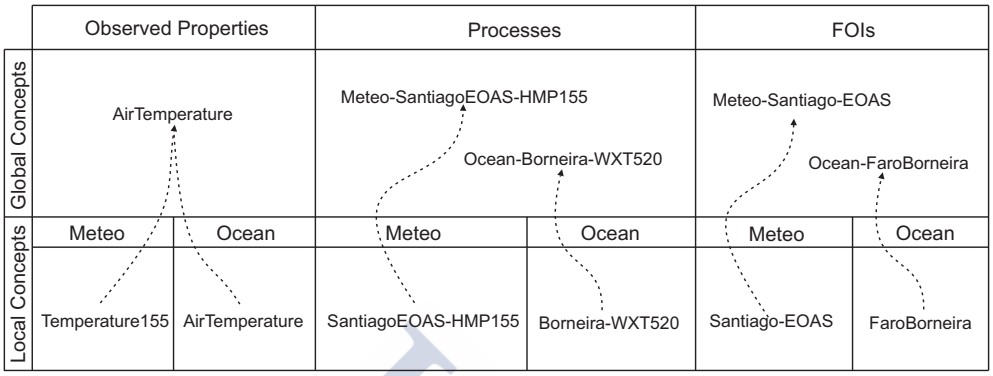


Figure 3.3: Example of global-local mapping of concepts

perature" subsumes both property "Temperature155" of data source "Meteo" and property "AirTemperature" of data source "Ocean". However, looking at the names of the properties, it is likely that global "AirTemperature" property is equivalent (the same property) to local property "AirTemperature" of data source "Ocean". Semantic relationships between ontology concepts will be used in the approach described in Chapter 4 to solve these issues.

3.2 Mediator/Wrapper frameworks architecture

The architecture of the framework (see Figure 3.4) is based on the well-known Mediator/Wrapper paradigm. Each of the components is next briefly described .

SOSDIService: It provides the SOS web service interface. It receives the HTTP either GET or POST requests from the client side, which are next passed to the SOSDIMeditorCore component through the ISOS interface. The responses generated by the SOSDIMediatorCore are next transmitted back to the client through HTTP. Its current implementation is based on the 52º North SOS 1.0 implementation (version 3.1.1).

SOSRequestParser: It provides functionality to parse the XML encodings of the SOS requests. Its current implementation is based on the 52º North SOS 1.0 implementation.

SOSXMLEncoder: It enables the generation of standard SOS XML responses. Its current implementation is based on the 52º North SOS 1.0 implementation.

ODIMManager: It provides access to the stored ODIM instantiation used by the framework, where both global offerings are defined in terms of queries to local ones and global-local

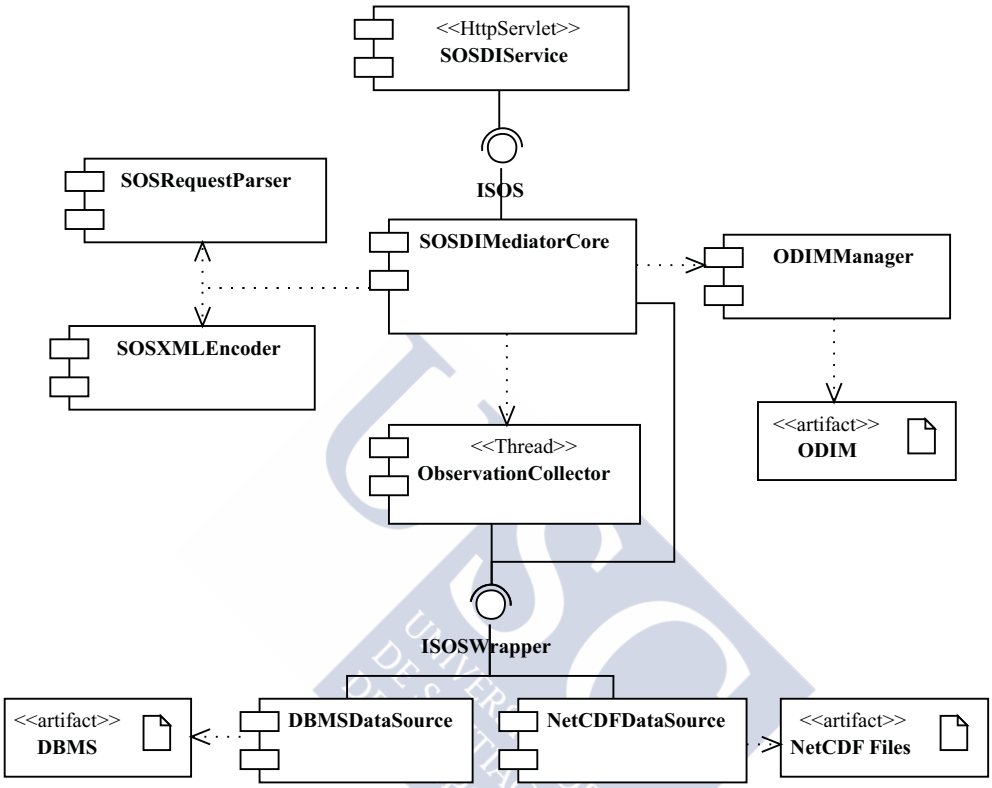


Figure 3.4: Mediator/Wrapper frameworks architecture. SOSDIMediatorCore distributes each request among the available data source wrappers, using the defitions of global fferings recorded in the ODIM.

mapping of *Properties* are recorded. In the current implementation, the ODIM instantiation is recorded in a relational database.

ISOSWrapper: This is the interface that each data source wrapper must implement. Figure 3.4 illustrates an example of a wrapper that access a DBMS with vector data and another example of a wrapper that access a NetCDF file with raster data. Additional details regarding the implementation of raster wrappers are given in Subsection 3.2.2. The interface contains methods for each of the implemented SOS operations. It is reminded that the list of data sources is recorded in the ODIM, together with the name of the wrapper class that implements this ISOSWrapper interface (see Figure 3.2).

ObservationCollector: A new instance of this class is created in a different Thread to invoke the *GetObservation* method of each data source wrapper. This way, each data source is queried in an asynchronous manner, improving the overall performance of the framework.

SOSDIMediatorCore: This is the main component of the proposed framework, which implements the data mediation algorithm that is described in more detail in the following subsection. SOS requests are received from the *SOSDIService* through the ISOS interface. Next, the *SOSRequestParser* is invoked to parse the requests. The *ODIMManager* is used to obtain the ODIM instantiation from persistent storage. The request is next distributed according to the requested global offering definition. To achieve this in a general and efficient manner, for each involved data source referenced in the ODIM instantiation, an instance of its wrapper class, which implements the *ISOSWrapper* interface, is created using the Java Reflection API. For each data source a new *ObservationCollector* is created in a different Thread, which enables the asynchronous execution of the different *GetObservation* requests of the different involved data sources. The observations retrieved from each data source are finally merged, encoded, using the *SOSXMLEncoder*, and delivered to the client side through the *SOSDIService*.

Further details related to the sensor observation data mediation and to the construction of raster wrappers are given in the following subsections.

3.2.1 SOS mediation

A brief description of the most important interactions between the software components of the architecture is given below for the implementation of each of the three mandatory SOS operations.

GetCapabilities. Yields descriptive metadata of both the service and its contents.

1. Based on the ODIM data (obtained from the *ODIMManager*) and on the capabilities of the data sources, the *SOSDIMediatorCore* composes the contents of each global *Offering*. In particular, temporal and spatial maximum extents are obtained as the intersection of those defined in the *GlobalOffering* class (properties time and boundedBy, respectively) with the union of the extents of all *GetObservationRequest* elements. The extension of each *GetObservationRequest* is obtained by applying temporal or spatial filters (properties *eventTime* or *FOI*) to the extents of relevant data source local *Offerings*. Examples of the generation of these real temporal and spatial extents of each

global offering were already given in Section 3.1. The lists of *Properties*, *Processes* and *FOIs* of each global *Offering* are obtained by filtering the lists of the relevant local offerings with the lists provided in the ODIM for each *GetObservationRequest* of the global offering. It is reminded that global identifiers of *Processes* and *FOIs* are automatically generated by appending the local identifiers to the identifier of the data source. Regarding global *Properties*, the relevant mappings recorded in the ODIM will be used.

2. The *GetCapabilities* response is constructed by adding to the above information of global *Offerings* the service capabilities, including descriptive information of the service interface and a list of the temporal, spatial and conventional operators supported by service query filters.

DescribeSensor. Yields the SensorML document that describes a specific data acquisition Process (commonly a physical sensor).

1. Data source and local *Process* identifiers are obtained from global *Process* identifiers. Thus for example, the global Process identifier "Meteo_SantiagoEOAS-HMP155" is composed of a data source identifier "Meteo" and a local Process identifier "SantiagoEOAS-HMP155".

2. The DescribeSensor operation of the relevant wrapper is invoked with the local *Process* identifier. The local *Process* identifier of the resposne is automatically replaced by the relevant global identifier of the same *Process*.

GetObservation. It is the most important operation since it enables clients to perform queries over the recorded collection of observations.

1. For each *GetObservationRequest* of the requested global *Offering*, the *SOSDIMediatorCore* instantiates a new *ObservationCollector* thread that invokes the *GetObservation* method of the *ISOSWrapper* interface for the relevant data source wrapper facade class. Instantiating the wrapper facade class by its name enables dynamic linking of the framework with new wrappers. Notice that the use of different threads enables data sources to evaluate their queries in parallel and this way the overall performance of the service is improved.

2. The state of each *ObservationCollector* thread is monitored and as soon as they finish their results are appended to the global result observation collection. The global-local mappings of *Observed Properties*, *Processes* and *FOIs* defined in Section 3.1 are applied at this stage.

It is noticed that at service start-up, the ODIM is loaded from persistent storage to main memory, avoiding too many future disk access, and an instance of each of the facade classes of the data sources is created. Therefore, to incorporate a new data source, it suffices to implement a new wrapper with a relevant facade class that implements the *ISOSWrapper* interface, register the data source with the facade class name in the ODIM, incorporate local *Offerings* of the data source in the definition of the global *Offerings* of the service and restart the *ODIMManager*. Finally, it has to be said that, in the current implementation, components *SOSDIService*, *SOSRequestParser* and *SOSXMLEncoder* are reusing code from version 3.1.1 of the SOS implementation of the open source software initiative 52° North [1].

3.2.2 Raster wrappers

Implicit or explicit vector to raster and raster to vector transformations are required to implement wrappers over raster data sources such as *NetCDFDataSource* of Figure 3.4.

More precisely, spatial filters of *GetObservation* requests must be rasterized to determine the pixels to be queried and result pixels must be vectorized to generate relevant *FOI* geometries. Current implementations of radar data raster wrappers simulate the behaviour of Bresenham line algorithm for rasterization and interpret result pixels as sampling points (located at the center of the pixel) for vectorization. Such an interpretation is in general applicable to most raster sources generated by remote sensors, which perform regularly spaced samplings of some surface. Examples of rasterizations of filter geometries are given in Figure 3.5 for a point, a line and a surface. In addition, Figure 3.5 also shows the central point of each pixel which is used as the relevant FOI geometry in the result O&M response.

Depending on the request, such an approach might lead to very large responses. However, it is also true that it is not the aim of the proposed approach to replace the functionality of raster services such as *Web Coverage Service* (WCS) and *NetCDFSubset*. Clients must have this in mind to use either SOS or WCS or *NetCDFSubset*, depending on the required functionality. Notice that WCS interfaces do not support the whole functionality of SOS. As an example,

[1] http://52north.org/communities/sensorweb/sos/index.html

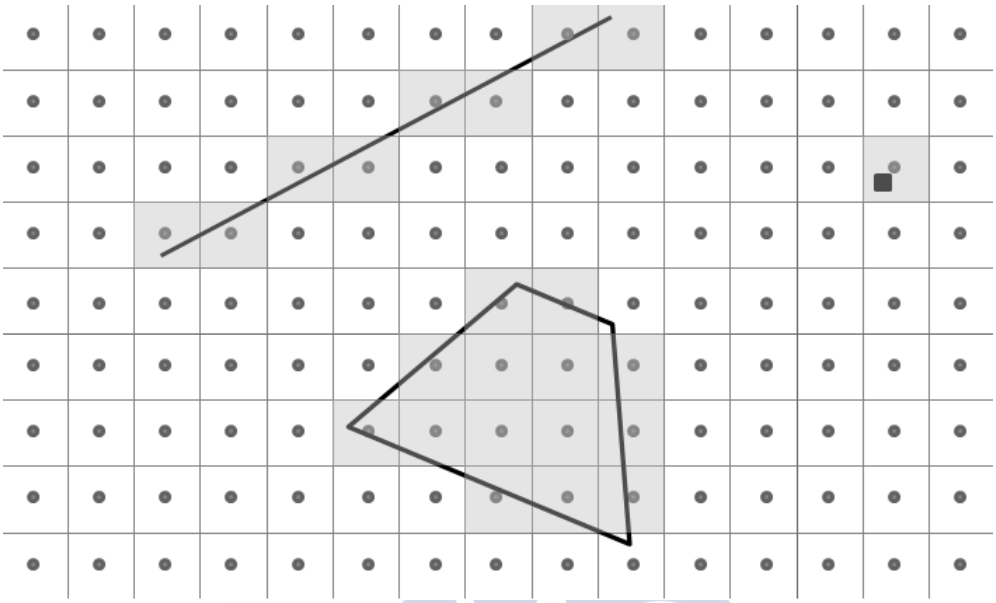


Figure 3.5: Rasterization of spatial filter geometries

the reflectivity values measured by meteorological radar at a given location (pixel) during one day, which might be requested from a SOS, are a time series and not a raster, thus they cannot be retrieved with a WCS. Time series are supported by *NetCDFSubset* services, however those services do not incorporate O&M semantics in the requests and responses. Future versions of the present framework might incorporate compact raster representations in the response O&M encoding, either directly embedded in the response or out of band by referencing relevant WCS or *NetCDFSubset* requests.

3.3 Framework validation and evaluation

The framework was evaluated by experts of two public agencies of the Spanish region of Galicia (northwest of Spain), namely MeteoGalicia [2] and INTECMAR [3]. MeteoGalicia is a meteorological agency with a wide range of meteorological and oceanographic observation *Processes*, including the following.

[2] Http://www.meteogalicia.es
[3] http://www.INTECMAR.org/

Figure 3.6: Map of the galician weather station network and an example of one of this meteorological station

Meteorological stations

A network of more than 80 automatic stations (Figure 3.6) equipped with a total of 700 different physical sensors. The majority of them are equipped with thermometer, barometer, hygrometer, anemometer, pyranometer and rain gauge. Around 120 different *Observed Properties* are observed and aggregated for periods of 10 minutes, one day and one month. Typically, the measured properties are related with temperature, humidity, wind, precipitation and pressure. The data is recorded in around 30 tables in a Microsoft SQL Server database.

Radio-sounding

A single radio-sounding or weather balloon (Figure 3.7) that measures various atmospheric parameters and transmits then by radio to a ground receiver. It carries a gps, barometer, thermometer and anemometer which measure six different Properties (wind direction and speed, height, temperature and pressure) with a time frequency of 10 minutes in radiosounding campaigns and recorded in a very simple Microsoft SQL Server database.

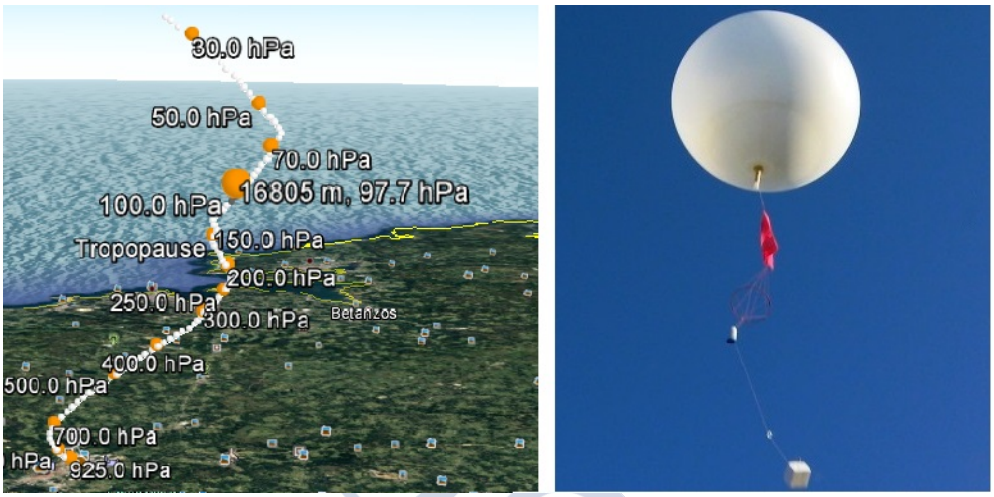


Figure 3.7: 3D Radio Sounding trajectory and device

Weather surveillance radar

Also called Doppler weather radar, is a type of radar used to locate precipitation, calculate its motion, and estimate its type (rain, snow, hail, etc.). This particularly observes 5 different Properties related to precipitation with a time frequency of 5 minutes and with a spatial resolution of one kilometer (390 x 390 cells). All the generated data is recorded in a series of files in NetCDF format, which result can be watched in Figure 3.8.

The second organization, INTECMAR, performs various types of analysis and observations related to the quality of the Galician marine environment.

Oceanographic automatic stations

A network of 8 heterogeneous stations located near the Galician coast (buoys like showed in Figure 3.9)that measure both meteorological properties at various elevations (wind speed, sea surface temperature, etc.) and oceanographic properties at various depths (chlorophyll, oxygen, etc.), aggregating values every 10 minutes, daily and monthly. Some of them have either horizontal or vertical Acoustic Doppler Current Profilers (ADCP), which are static sensors that measure current velocities in remote. The data is recorded in around 15 tables in a Microsoft SQL Server database.

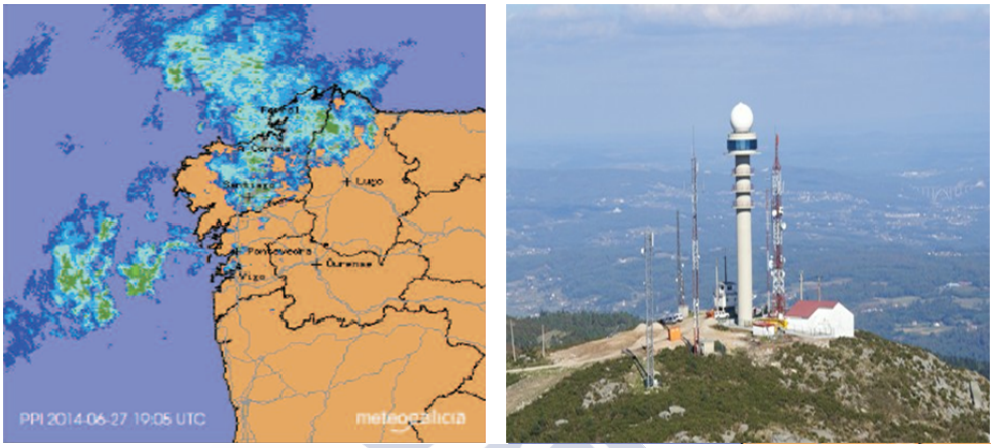


Figure 3.8: Weather Surveillance Radar reflectivity image and galician doppler facility

Figure 3.9: Map with some buoys at the galician coast and an example of one of these ocean buoys

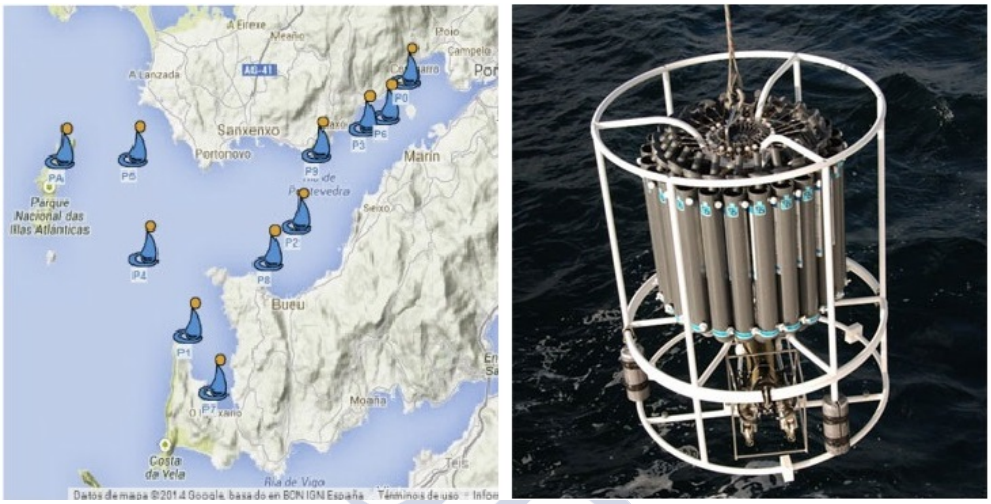


Figure 3.10: Map with the virtual fixed locations at the galician coast and an example of a CTD

Conductivity, Temperature and Depth (CTD) profiles

They are obtained weekly at 43 fixed locations inside the Galician Rias (estuaries). Figure 3.10provides a geographic representation of such locations in one the Rias. Thirteen different *Observed Properties* (10 primitive and 3 derived) are measured at various different depths along the profile. All the data is recorded in a Microsoft SQL Server database, whose relational model contains 20 tables.

High Frequency Radar (HF Radar)

The Galician network of HF Radar consists of 5 different SeaSonde LR HF radar stations (Figure 3.11). INTECMAR records the east and north components of sea surface current vectors provided by HF Radar with a spatial resolution of 6 kilometers (50 x 62 cells) and with a temporal resolution of one hour. Such data is recorded in files of NetCDF format, which result can be watched in Figure 3.11, and served through a thredds server.

A distinct wrapper was implemented for each of the above data sources. Besides, a wrapper has been designed and implemented that enables the access to external SOS services, thus MeteoGalicia is going to be a data source for INTECMAR and vice versa. Such wrapper enables the deployment of SOS servers in cascade.

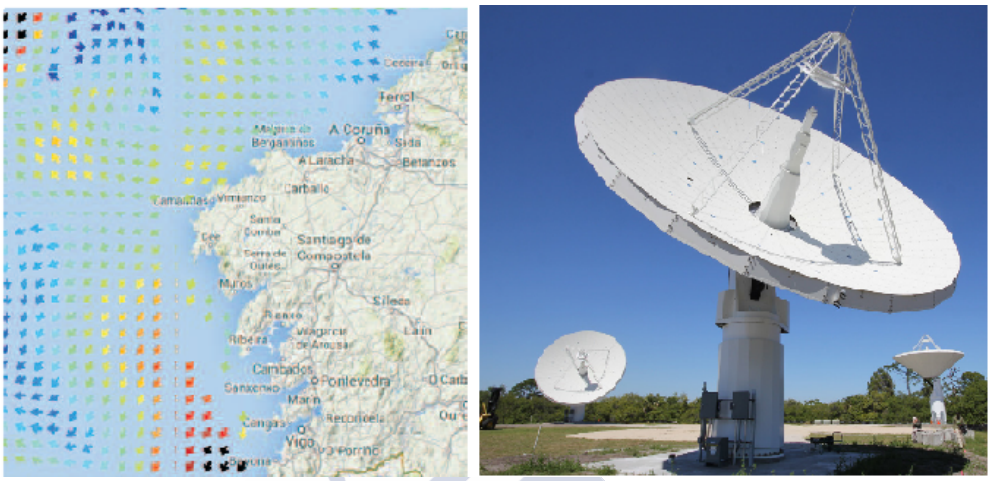


Figure 3.11: High Frequency Radar image and a radar example

In addition to the validation of the framework in two real scenarios, it has also been evaluated with respect to related approaches considering the following criteria.

Client complexity

Performing data integration in the sever-side leads to much simpler client-side software that may be tailored to users with lower skills. In spite of this, if high skill client-side experts need to control the integration process, then the server may also provide the data in different Offerings, without any kind of integration. That is, the admin choose can enable data integration within same offering or disable by defining different offering for each data source.

Data management infrastructure investment

A virtual data integration approach has a reduced impact in the investment in data management infrastructures to be undertaken by the organization. Therefore, it is advocated as a realistic solution for organizations that do not want to make important changes in their information systems Notice that the software adapter replaces this entire hard staff and avoids annoyances to the data providers.

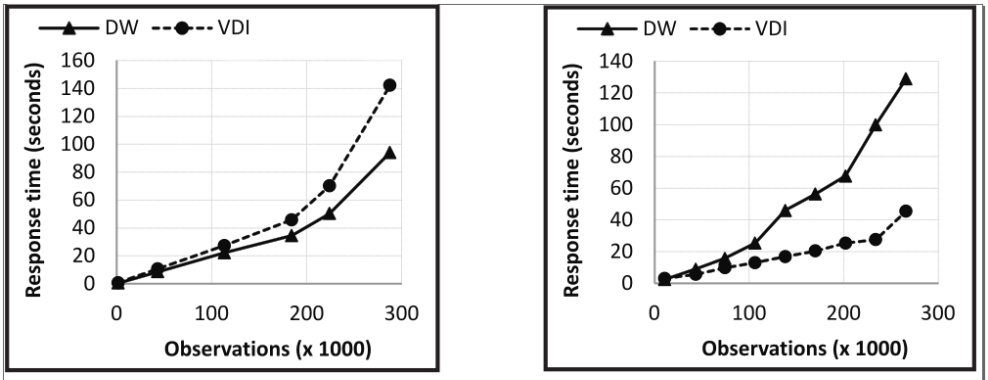


Figure 3.12: SOS virtual data integration performance evaluation

Vector and raster integration

Currently available SOS servers implement data warehouse solutions implemented with relational DBMSs. Such an approach is clearly not designed to take into account the characteristics of raster data sources, as it is also deduced from the performance evaluation below in figure performance.

Performance

Two performance tests were executed in a conventional computer with CPU Intel Core i3 (2.8 GHz) and 4 GB of RAM, in order to compare the present framework with a representative data warehouse SOS implementation [4]. First, around 1.2 million observations of meteorological stations and 1.1 million observations of oceanographic stations where loaded both in two data sources of the present framework and in the PostgreSQL database of 52° North SOS. The relationship between response times and the number of observations retrieved are shown in Figure 3.12(a). The data warehouse approach has better performance since its data model has been specifically designed for SOS publishing and data integration is not performed at query time, but at ETL time. Second, the 1.2 million observations of meteorological stations above where combined with 8.9 million of radar observations (raster pixels) recorded in a NetCDF file (raster wrapper of the present framework). Both data sources where loaded in

[4] Version 3.1.1 of 52° North SOS implementation - http://52north.org/communities/sensorweb/sos/index.html

the PostgreSQL database of 52° North SOS. Now clearly the present framework has much better performance (see Figure 3.12(b)) since the relational model of the 52° North SOS implementation has not been designed to record raster observations.

3.4 Chapter conclusions

Along this chapter the design, implementation and evaluation of an initial framework for environmental data integration was described. Such framework provides a real solution to the problem of virtual integration of heterogeneous observation data sources in environmental application domains. The framework is currently being validated in two real scenarios with meteorological and oceanographic data. The source code is licensed under GPL version 3. Some advantages of the approach are the following:

- The proposed solution is working in two public agencies, one meteorological and other oceanographic. Hence, the framework is being evaluated with real data that includes almost all types of sensors.
- Server-side data integration leads to simpler client software that may be tailored to users with lower skills. Besides, same data integration code is not replicated in many clients.
- The proposed virtual data integration approach does not require additional data duplication and reduces the required investment in data management infrastructure.
- Minimize the impact in current information systems since it is not necessary to add additional hardware and avoid changes in current systems.
- Efficient integration of vector and raster data is directly supported by the construction of raster data wrappers.
- Currently SOS implementations follow a centralized data warehouse architecture based on the use of database technology, which is not designed to incorporate raster observation data unlike the proposed implementation.
- Highly heterogeneous observation data is integrated including:
  - Vector data from in-situ static sensors (stations)
  - Raster data from remote static sensors

        - 1D raster from ADCP (vertical and horizontal)
        - Raster data from remote static sensors
        - 2D raster from Radar (Weather surveillance radar and High frequency radar)
    - 3D vector data in the form of trajectories from radio sounding campaigns
    - Vector data in the form of profiles from CTD sensors

- The well known Mediator/Wrapper architecture paradigm followed makes the framework very flexible in the incorporation of new data sources.
- Its multi-thread implementation approach for data source querying enables the framework to leverage currently available multi-core hardware architectures.
- The implementation of generic external SOS wrappers enables the deployment of SOS services in cascade, which is an interesting functionality for the construction of Spatial Data Infrastructures, that is, possibility of SOS hierarchy among different suppliers.

In conclusion, the content of this chapter, that is the description of a data mediation approach, satisfies the first objective of the thesis described at the introduction section.

CHAPTER 4

SEMANTIC MEDIATION THROUGH SENSOR OBSERVATION SERVICES

The main challenges to be faced in order to achieve integrated access to environmental data sources are related to both data source heterogeneity and semantic conflict resolution, as it was already stated at Chapter 1. Heterogeneity in data modeling frameworks, data models, interfaces and encodings was already addressed by the data mediation architecture proposed in Chapter 3. As it was already stated also, semantic conflicts appear when either different terminology is used in different data sources for the same concepts or the same terminology is used in different data sources to denote different concepts.

Based on the above, this chapter provides an overall description of a framework for the semantic mediation between heterogeneous environmental observation datasets through OGC SOS interfaces. Version 1.0 of the SOS interface is used by the current version of the framework which follows a *Local as View* data integration approach in the mediator and simplifies the incorporation of new data sources. The system uses both SSN and SWEET ontologies as the basis for the specification of data integration knowledge by the domain expert. This approach has two main advantages: i) The framework may be applied to new application domains with different data sources by just developing new wrappers and changing the expert ontology and ii) general purpose semantically enabled applications that exploit the knowledge of the expert ontology may be developed on top of the framework by users without specific application domain knowledge.

The remainder of this chapter can be outlined as follows. The data mediation architecture is described in section 4.1. Section 4.2 illustrates the contents of data source ontologies. The definition of data integration knowledge is provided in section 4.3. Section 4.4 describes the implementation of the semantic data mediation process. Qualitative and performance evaluation results are discussed in section 4.5. Finally, section 4.6 summarizes the conclusions of the chapter.

4.1 Data mediation architecture

The architecture of the proposed framework is based on the well-known Mediator/Wrapper data integration architecture [79], as it was shown in Chapter 3. Each wrapper is specifically designed for the characteristics of a data source and it adapts its specific data model and data access interface to O&M and SOS. Beyond that, wrappers provide also a means to add the semantic annotation that will later be used during querying and semantic mediation. The mediator will receive integrated SOS requests and distribute them among the various *Offerings* of the available data sources. The distribution of the request is guided by data integration knowledge defined by the domain expert in a mediator level ontology.

The data integration architecture that shows the interdependencies between the data source and mediator ontology is depicted in Figure 4.1. The ***SSN Ontology***, at the top of the figure, provides the basic concepts that are required by the O&M data model (*Observation*, *Process*, *FeatureOfInterest*, *Property*) and also relationships among these concepts (observes, observedBy, featureOfInterest, observedProperty). The ***Core Ontology*** completes SSN with other required concepts in O&M, which will be explained in following sections.

Local concepts of data sources are defined in ***Data Source Ontologies***, which are specific for each domain. Data source ontology classes may also be related to SWEET classes by the definition of relevant class annotations. A more detailed description of these data source ontologies is given below in Section 4.2.

The ***Mediator Ontology*** includes both global classes that may be used to integrate various local ones and semantic relationships between global and local classes and individuals. Beyond the above concept mappings (similar to the *glue knowledge* of [46]), the domain expert may also define global *Offerings*, which might simplify the specification of many typical user queries. A more detailed description of the contents of *Mediator Ontology* and how it is used to achieve semantic integration is given below in Section 4.3.

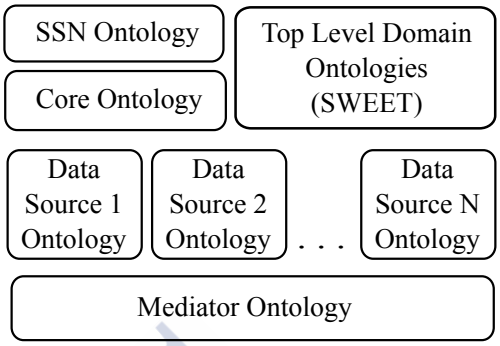


Figure 4.1: Data mediation architecture

The use of the standardized O&M model and SOS interface to communicate mediator and wrappers enables the simplification of the data integration challenges identified in [42].

- The assumption of standardized SOS interfaces and consequently O&M data model at both global and local levels avoid the need to define relationships between global and local data model elements.
- The resolution of syntax conflicts during the integration process is also avoided by the use of SOS interface. On the other hand, semantic conflicts may still arise. Those conflicts must be solved by the specification of appropriate data integration knowledge at the mediator ontology in the form of semantic relationships between global and local classes and individuals.
- Query reformulation algorithms for global queries are also simplified by the use of common SOS interface in all the wrappers, therefore, a Local As View (LAV) approach becomes feasible with a reduced effort.

As a consequence of the above, the main contribution of the present data mediation framework is the resolution of semantic conflicts between data sources during the query evaluation. This is achieved by the appropriate processing of the RDF graph of the *Mediator Ontology* with the help of SPARQL.

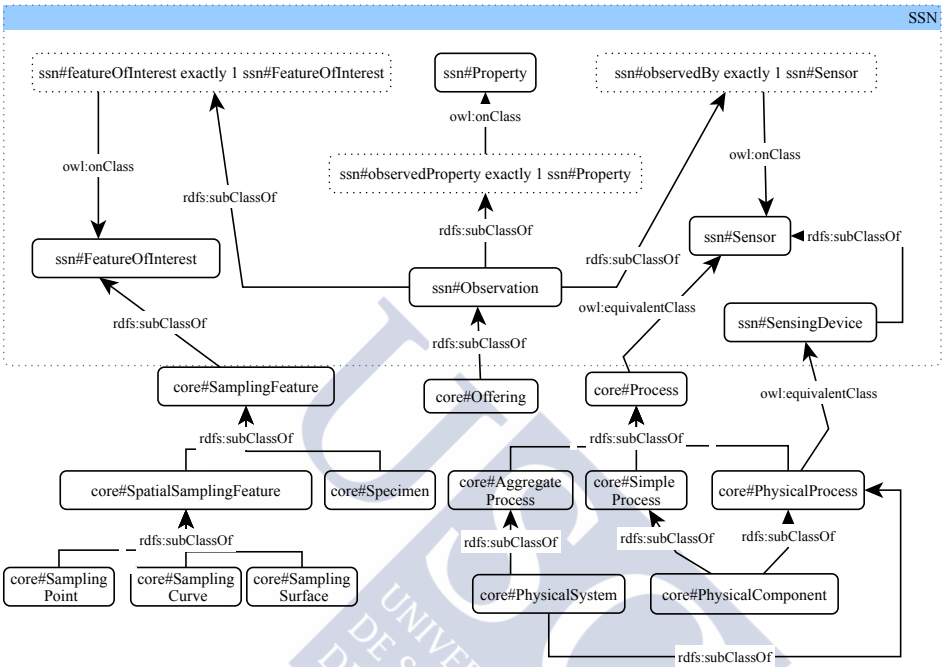


Figure 4.2: Core ontology.

4.2 Data source ontologies

A more detailed description of the Core and Data Source Ontologies is provided in the following subsections.

4.2.1 Core ontology

The *Core Ontology* defines required O&M [20] and SensorML [5] concepts as specializations of relevant SSN concepts. Figure 4.2 depicts a graphical representation of the classes, restrictions and predicates of this ontology, together with appropriate SSN context.

At the top of the figure, required SSN concepts are depicted, together with representative restrictions and *subClassOf* predicates. Restrictions are depicted inside dotted rectangles

using Manchester Syntax, and they represent the fact that each observation must reference exactly one *Sensor*, *Property* and *FOI*.

Based on the above SSN concepts, the following core classes and hierarchies required by the framework are defined.

core#Offering

All the *Offerings* provided by data sources and mediator will be subclasses of this core class. Thus, the semantic interpretation of each *Offering* subclass will be the set of individuals (observations) that belong to it, that is, each single observation will be an instance of this class.

core#SamplingFeature

It represents the concept of *Sampling Feature* defined in the O&M standard data model [20]. As it is argued in [20], the ultimate domain specific *FOI* whose properties are of interest does not match in most cases the proximate FOI linked to each observation. For an example consider a collection of buoys sampling seawater temperature in the Gulf of Mexico. The ultimate *FOI* is the seawater of the Gulf of Mexico, however, it is fundamental to know which buoy is associated to each observation in order to perform required analytics (spatial interpolation for example). Two major types of *Sampling Features* are identified in O&M. *Spatial Sampling Features* arise when the ultimate *FOI* has a geospatial nature and proximate *FOIs* provide samplings at specific locations. Various subclasses are defined based on its underlying geometry (point, curve, surface or solid). A typical example of a *Sampling Spatial Feature* is a sampling station (meteorological station, buoy, etc.). A *Specimen* is used to model physical samples obtained from the ultimate *FOI* and carried out to be observed. An example is a sample of water obtained from a specific location in a river to be analyzed in a laboratory.

core#Process

It represents a SensorML *Process* [5], which is equivalent to a SSN *Sensor*. A typical example of such concept is a thermometer. Both physical and non-physical (computing processes for example) and simple and aggregate processes may be represented. A physical process is also represented by the SSN *Sensing Device* class. A *Physical Component* is a simple and physical process whereas a *Physical System* is an aggregate process that has some physical component.

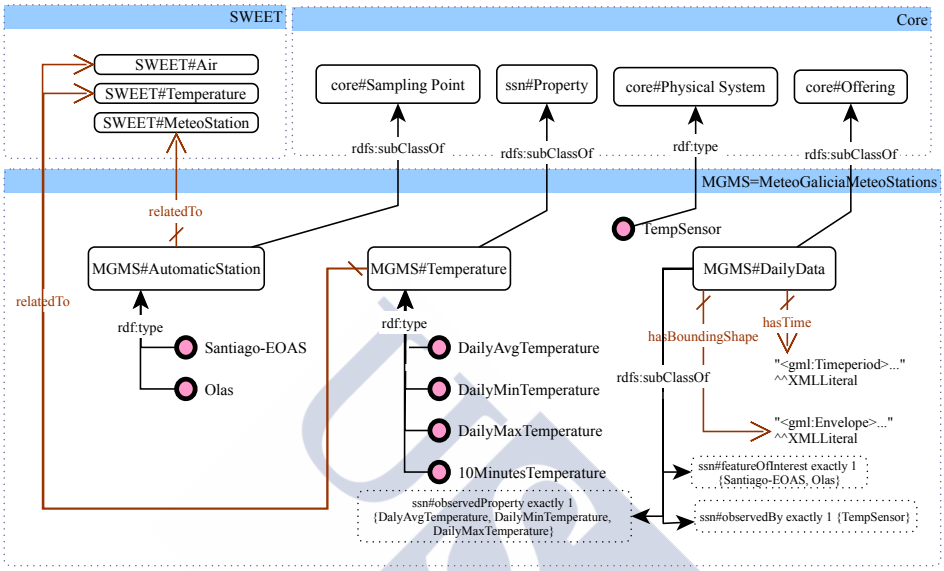


Figure 4.3: Data source ontology

4.2.2 Representation of data source concepts

Data source *Properties*, *Processes*, *FOIs* and *Offerings* are defined in each *Data Source Ontology* as either specializations of relevant *Core Ontology* classes or as individuals of them. Besides, to broaden the mediator query capabilities, defined classes might be related to classes of some well-known top-level application domain ontology. In the current implementation, SWEET was used as such an environmental domain specific ontology [60]. Figure 4.3 represents some concepts of the MeteoGalicia meteorological stations data source.

Each *Property* of a data source is defined as an individual of either *ssn#Property* or some subclass of it specifically defined in the data source (see *MGMS#Temperature* class and relevant individuals in Figure 4.3). Relationships between data source classes and SWEET are modeled with *relatedTo* annotations.

A similar approach is followed for the representation of *FOIs* and *Processes* in each data source. Notice that a subclass of *core#SamplingPoint* is defined to model automatic meteorological stations in the example. Such a new class is also defined to be related to SWEET

Meteostation class. An individual of *core#PhysicalSystem* is included to represent a temperature sensor. Notice that it is defined as a physical system because it includes a physical sensing device that obtains temperature measures in the station and it also includes algorithms to compute daily aggregates.

Data source *Offerings* are modeled by specific subclasses of *core#Offering*. For an example, see the subclass *MGMS#DailyData* in Figure 4.3. The temporal and spatial extent of the *Offering* are represented by two annotation properties of RDF type *XMLLiteral* that contain respectively relevant GML *TimePeriod* and *Envelope* elements. Two more optional annotation properties might be included to provide the name and description of the *Offering*. Besides, three class restrictions are used to represent the *Properties*, *Processes* and *FOIs* referenced by the observations of the *Offering*. Thus, as it is shown in the figure, the *DailyData Offering* provides daily average, minimum and maximum temperatures, generated by the *TempSensor* at *Santiago-EOAS* and *Olas* meteorological stations. Notice that all the metadata required to describe the capabilities of each *Offering* are represented in this way in the data source ontology. It is also noticed that *Offerings* are defined as views of the global O&M data model, following a LAV approach [42]. Thus, the above definitions will be used to automatically determine which local *Offerings* have to be accessed to obtain the observations of each global *Offering*.

4.3 Representation of data integration knowledge

Data integration knowledge includes the definition of new classes, the specification of semantic relationships between local and global concepts and the definition of global *Offerings*. Three types of semantic relationships may be specified between classes and individuals defined in data source and mediator ontologies. Figure 4.4 illustrates the definition of these relationships for the three data sources of the proposed use case.

4.3.1 Subclass relationships

They are represented by the property *subClassOf* of RDFS and they enable the integration of various *Property*, *Process* or *FOI* classes into a single one. In the example of Figure 4.4, class *med#SamplingStation* is used to integrate meteorological and oceanographic stations of the three data sources. Now, this new mediator class can be used in both *GetObservation* requests and the definition of global *Offerings*. Notice also that the global *med#SamplingStation* class

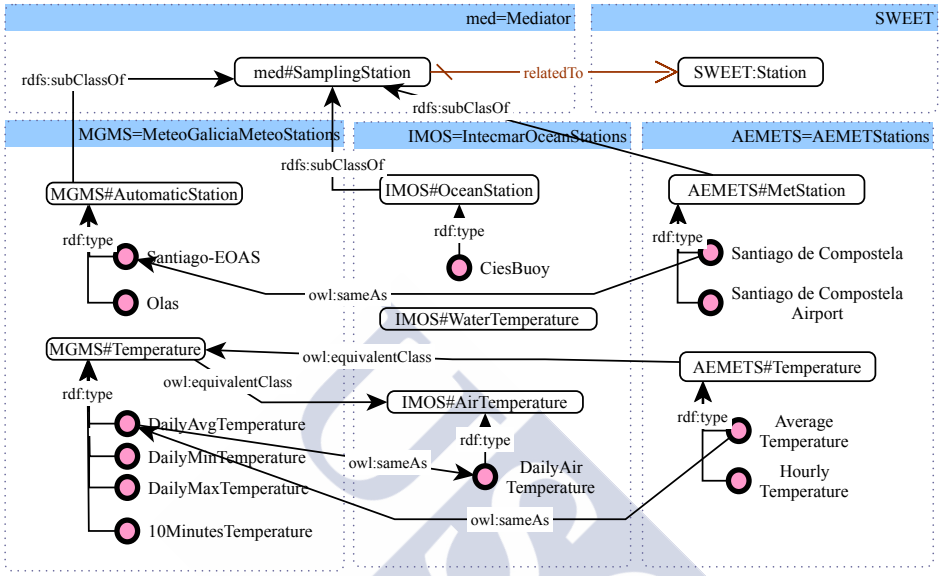


Figure 4.4: Data integration knowledge representation.

is defined to be related to the class *Station* of SWEET, therefore, this SWEET class may be used by semantic clients to find all the stations of the three data sources.

4.3.2 Class equivalence relationships

They are represented by the predicate *equivalentClass* of OWL. They enable the representation of the fact that various *Property*, *Process* or *FOI* classes represent actually the same class, despite of having different names in different data sources. Notice that this enables querying the *Offerings* of one data source using URIs of concepts that may be defined in other data sources. As an example, various property classes representing air temperature are defined to be equivalent in the example provided in Figure 4.4.

4.3.3 Individual equality relationships

They state that two *Property*, *Process* or *FOI* individuals represent actually the same individual. They are represented by the predicate *sameAs* of OWL. This enables the representation of the fact that a given individual might be present in various data sources with different names. Thus, in the example of Figure 4.4 the automatic station *Santiago-EOAS* of MeteoGalicia is exactly the same station that is accessed through AEMET with the name *Santiago de Compostela*. Therefore, queries referencing *Santiago-EOAS* at the mediator should also retrieve the data of *Santiago de Compostela* recorded by AEMET. As another example, in the figure various daily air temperature properties are defined to be the same one, despite of having different names.

Beyond the definition of relationships between local and global concepts, in order to simplify typical user queries, the application domain expert may also define global *Offerings* that might integrate observations of various data sources. Each such new *Offering* will be defined as a subclass of *core#Offering*. The optional name and description of the *Offering* may be provided with *hasName* and *hasDescription* annotations properties. Temporal, Spatial and value filters may be specified, respectively, with *hasTemporalOps*, *hasSpatialOps* and *hasComparisonOps* annotation properties. A restriction on the possible *Properties* that the *Offering* observations may reference is specified by an expression of the form (using Manchester Syntax)

ssn#observedProperty exactly 1
P_1 OR P_2 OR ... OR P_n OR
$\{p_1, p_2, \ldots, p_m\}$

where, P_i are direct or indirect subclasses of *ssn#Property* and p_i are individuals of direct or indirect subclasses of *ssn#Property*. Similar restrictions may be defined to filter *Processes* and *FOIs*, using SSN properties *ssn#observedBy* and *ssn#featureOfInterest*, respectively. As an example, Figure 4.5 provides a graphical representation of the definition of a global *Offering med#AirTempJan2013* that enables the access to daily averages of air temperature from all the stations of the three data sources. Notice that the temporal filter is specified with a *hasTemporalOps* annotation property and *Property* and *FOI* filters are defined by relevant class restrictions. It is finally noticed that although the restriction references just the property *AEMETS#AverageTemperature*, observations of the other two data sources are also accessed,

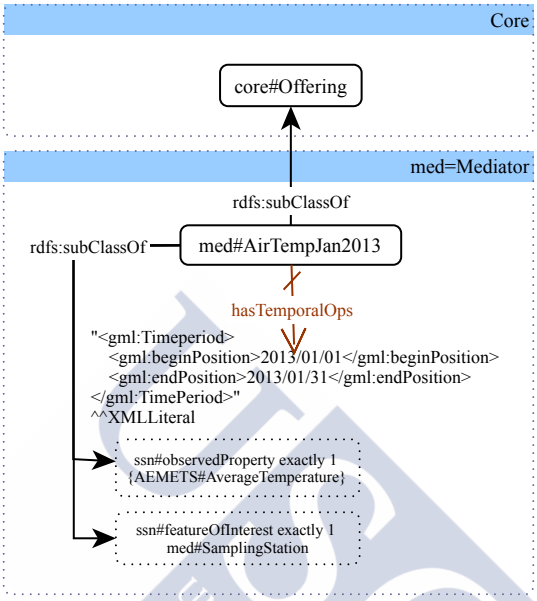


Figure 4.5: Global *Offering*

due to specific individual equality relationships defined in the *Mediator Ontology* (see Figure 4.4).

4.4 Implementation of semantic data mediation

The three mandatory operations of the SOS interface are implemented by the mediator of the proposed framework. At the current version of the framework, *DescribeSensor* operation does not take advantage of semantic integration capabilities, thus only the other two operations *GetCapabilities* and *GetObservation* are discussed in the following subsections. To ease the description of the algorithms that implement these operations, some preliminary pieces of functionality have to be introduced.

If p is an RDF predicate, then $inv(p)$ denotes the inverse predicate of p. If r is an RDF resource and p is an RDF predicate, then $r.p$ denotes the set of resources $\{r_i\}$ such that the triple $(r\ p\ r_i)$ belongs to the ontology RDF graph.

If R is a set of RDF resources and p is an RDF predicate, then $TClosure(R,p)$ denotes all the resources in the transitive closure of p, i.e., all the resources r_i for which a sequence of triples of the form

$< (r\ p\ s_1), (s_1\ p\ s_2), \ldots (s_{n-1}\ p\ s_n)(s_n\ p\ r_i) >, r \varepsilon R$

exists in the ontology RDF graph.

Let R be a set of RDF resources and P be a set of RDF predicates. Then the *Generalized Transitive Closure* of P from R, denoted $GTClosure(R,P)$ is obtained by iteratively adding to R the $TClosure(R,p)$ for each p in P, until the size of R does not change in two consecutive iterations. Informally, $GTClosure(R,P)$ yields all the resources related directly or indirectly with resources of R by some predicate of P.

Let C be a set of OWL classes. Then the operation $Individuals(C)$ yields all the individuals of classes of C.

Let C be a set of OWL classes and let s be another OWL class. The set of all subclasses of s that are related to classes in C is denoted by $RelSubClasses(C,s)$ and it contains all the subclasses of s that also belong to

$GTClosure(C, \{inv(\text{rdfs:subClassOf}),$
$\quad inv(\text{owl:equivalentClass}),$
$\quad inv(\text{owl:equivalentClass})\})$

4.4.1 Operation GetCapabilities

All the data required by the *GetCapabilities* response for each Offering of each wrapper is already contained in its *Data Source Ontology*, as it was shown in Subsection 4.2.2. However, for mediator *Offerings*, the temporal and spatial extension and the set of *Property*, *Process* and *FOI* individuals are not directly available and they have to be deduced by some reasoning algorithm. The following steps provide an overall description of such an algorithm.

1. Obtain the spatial and temporal filters from relevant annotation properties of the mediator *Offering*.

2. If *PropInds* and *PropClasses* are respectively the sets of individuals and classes referenced in the restriction on property *ssn#observedProperty* of the mediator *Offering*,

then the set *AllPropInds* of all *Property* individuals referenced either directly or indirectly by the restriction is obtained as follows.

$C \leftarrow RelSubClases(PropClasess, ssn\#Property)$
$I \leftarrow Individuals(C) \cup PropInds$
$AllPropInds \leftarrow GTClosure(\text{I}, \{\text{owl:sameAs},$
$\text{inv(owl:sameAs)}\})$

As an example, for the mediator *Offering* of Figure 4.5, the obtained set of all *Property* individuals would be

{MGMS#DailyAvgTemperature,
IMOS#DailyAirTemperature,
AEMETS#AverageTemperature}

3. In a similar manner, obtain also the set of all *Process* and *FOI* individuals referenced either directly or indirectly by relevant restrictions of the mediator *Offering*. For the mediator *Offering* of Figure 4.5, the set of all *Process* individuals would be empty and the set of all *FOI* individuals would be

   {MGMS#Santiago-EOAS,
   MGMS#Olas,
   IMOS#CiesBuoy,
   AEMETS#SantiagoDeCompostela,
   AEMETS#SantiagoDeCompostelaAirport}

4. For each wrapper *Offering*

   a) Obtain the wrapper *Offering* temporal and spatial extent and filter them using the filters obtained in step 1.

   b) Obtain the wrapper *Offering* sets of *Property*, *Process* and *FOI* individuals and filter them using the relevant sets of individuals obtained in steps 2 and 3. For example, for the wrapper *Offering* of Figure 4.3, the filtered *Properties*, *Processes* and FOIS are respectively the following:

      {MGMS#DailyAvgTemperature},
      {MGMS#TempSensor}
      {MGMS#Santiago-EOAS, MGMS#Olas}

c) If none of the above filtered elements is empty then the wrapper *Offering* will contribute to the observations of the mediator *Offering*. Therefore, the filtered temporal and spatial extensions and the filtered sets of *Properties*, *Processes* and *FOIs* have to be merged with the mediator *Offering* relevant extensions and sets.

It is noticed that the above algorithm determines automatically, which data source *Offerings* have to be accessed for each global *Offering*. Therefore, either changes in local *Offerings* or the incorporation of new data sources will not require the redefinition of global *Offerings*. This is a clear advantage of the LAV approach followed.

Finally, it is remarked that the *Mediator Ontology* is referenced in a specific XML element inside the contents section of the *GetCapabilities* response. This way, advanced clients may take advantage of the whole ontology maintaining at the same time backward compatibility with standard SOS clients.

4.4.2 Operation GetObservation

A request to this operation references just one *Offering* and at least one *Property*. Additionally, it may contain temporal, spatial and value filters and lists of *Processes* and *FOIs*. Now, URIs of individuals and classes of the *Mediator Ontology* may be used to reference *Properties*, *Processes* and *FOIs* in a request. Therefore, another reasoning algorithm hast to be used to determine the *GetObservation* request that has to be sent to each wrapper. The following steps provide an overall description of such an algorithm.

1. Obtain the sets of all the *Property*, *Process* and *FOI* individuals referenced either directly or indirectly by classes and individuals included in the request (see steps 2 and 3 in subsection 4.4.1)

2. If the requested *Offering* is a wrapper *Offering*, then

   a) Obtain the *Offering* temporal and spatial extents and filter them using the relevant request filters.

   b) Obtain the *Offering* sets of *Property*, *Process* and *FOI* individuals and filter them using the relevant sets obtained in step 1 above.

   c) If none of the above filtered elements is empty, then the *Offering* has to be queried, therefore a *GetObservation* request is sent to the relevant wrapper. Filtered tem-

poral and spatial extensions and filtered sets of *Properties*, *Processes* and *FOIs* obtained in the previous two sub-steps will be included in the request.

3. If the requested *Offering* is a mediator *Offering*, then

   a) Obtain the spatial and temporal filters of the request and combine them with the spatial and temporal filters of the *Offering*.

   b) Obtain the set of all the *Property*, *Process* and *FOI* individuals referenced directly or indirectly by classes and individuals in the *Offering* relevant restrictions (see steps 2 and 3 in subsection 4.4.1). Combine the above sets with the sets obtained in step 1.

   c) Using the temporal and spatial filters and the sets of all the set of all the *Property*, *Process* and *FOI* individuals obtained above apply steps 2(a-c) for each wrapper *Offering*. The *GetObservation* request to all the required wrappers are submitted asynchronously by the mediator using a pool of threads. To achieve this, the current Java implementation uses the ExecutorService class of the *java.util.concurrent* package. The responses of all the generated *GetObservation* requests are merged by the mediator to generate the result integrated response.

4.5 Qualitative and performance evaluation

A first prototype of the framework was already implemented, using the real datasets of MeteoGalicia and INTECMAR described in Section 3.3. The Apache Jena SPARQL engine ARQ was used to query the *Mediator* and *Data Source Ontologies* during *GetCapabilities* and *GetObservation* processing. The results of a first evaluation of the system, which include both a use case application and performance analysis are described in the following subsections. Beyond that, the functionality of the prototype was already evaluated by experts of MeteoGalicia and INTECMAR and it is the basis of the currently on-going implementation of their standardized environmental time series data access point, including both observation and model data. Such development has already started in the scope of a technology transfer project funded by these two entities.

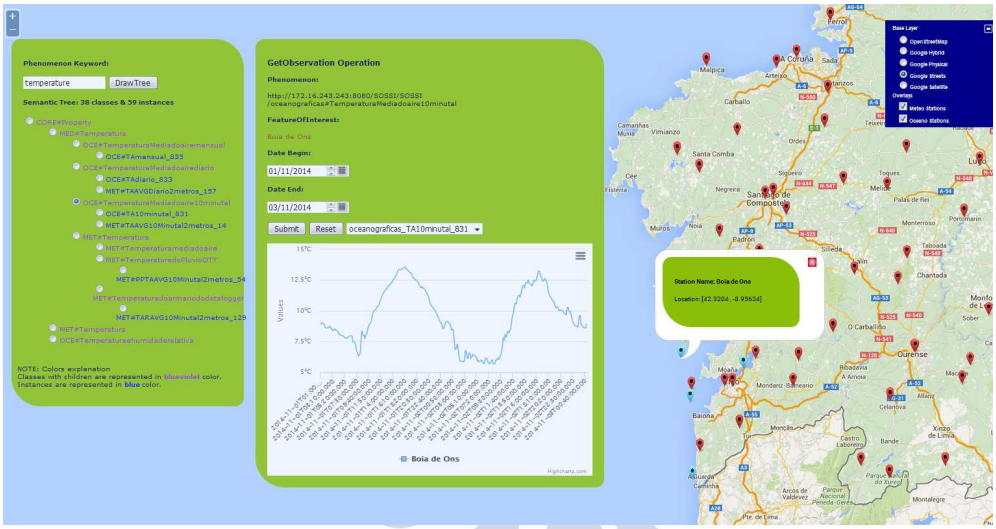


Figure 4.6: Use case semantic web application.

4.5.1 Use case: meteorological and oceanographic station data mediation

Two applications, a general purpose semantic web application and a specific mobile application for yachting in Galician estuaries (Rias), were developed to test the functionality of the framework. Both applications demand semantic integration of meteorological and oceanographic station networks of MeteoGalicia and INTECMAR. The web application exploits the contents of the *Mediator Ontology* to construct the enhanced end-user interface depicted in Figure 4.6.

The panel located at the left side of the interface contains a search box where the user may type keywords. Those keywords are used to query the *Mediator Ontology* referenced in the *GetCapabilities* response to obtain the result. The following relationships are taken into account:

– Subclasses of *Core#Property* that are directly or indirectly *rdfs:subClassOf* some class whose URI contains the query terms. Thus, *Property Classes* defined in *Mediator* and *Data Source Ontologies* will be queried.

– Subclasses of *Core#Property* that are *core:relatedTo* some class that is directly or indirectly subclass of some class whose URI contains the query terms. Thus, if the user enters the keyword “Temperature", which is contained in the URI SWEET#Temperature, all the *Properties* of the ontology defined as *core:relatedTo* some subclass of SWEET#Temperature will also be retrieved.

– Classes declared directly or indirectly as equivalent (*owl:equivalentClass*) to some of the above classes.

– Instances of any of the above classes.

– Instances of some direct or indirect subclass of the class *Core#Property* whose URI contains the query terms.

– Instances declared directly or indirectly as *owl:sameAs* some of the above instances.

The result hierarchy of *Core#Property* subclasses and instances is presented to the user as a tree immediately below the search box. The user may choose any element of the tree to construct a *GetObservation* SOS request. At the right side of the interface, a map is used to represent the *CORE#SamplingPoint* individuals (meteorological and oceanographic stations) obtained with a SOS *GetFeatureOfInsterest* request. The panel at the center of the interface is used to create the *GetObservation* request, using the Property element selected in the tree, the station selected in the map and a couple of dates. One or various time series may be obtained and graphically depicted in the center panel, as it is shown in the figure 4.6.

Regarding the yachting mobile application, it consumes both data of observation stations and prediction models. The former is obtained through the present framework, which performs semantic integration of meteorological and oceanographic data from the two station networks of MeteoGalicia and INTECMAR. Figure 4.7 shows an screenshot of the application displaying meteorological and oceanographic stations.

4.5.2 Performance evaluation

The advantages of using semantic web technologies have already been described throughout the chapter. Now, the impact of the use of semantic web technologies in the performance of the framework will also be shown. To achieve this, the semantic mediation approach of the present framework (denoted here SM) has been compared with the virtual data integration

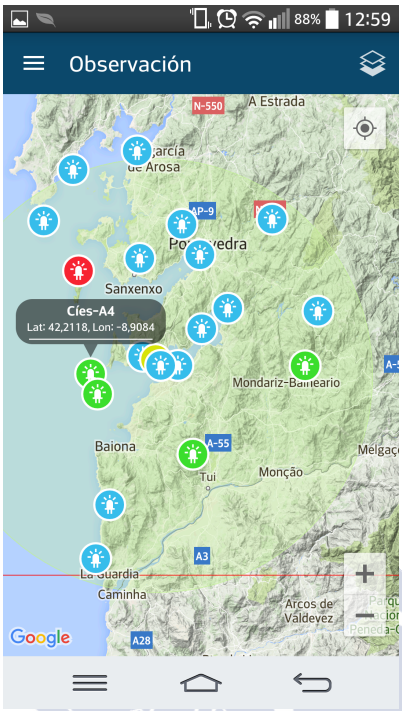
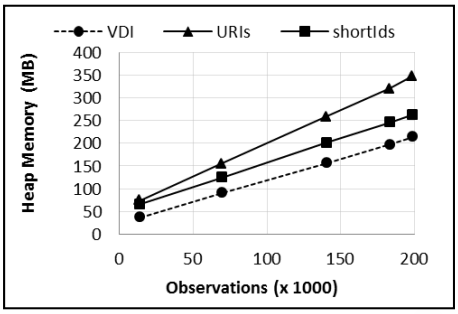


Figure 4.7: Yachting mobile application

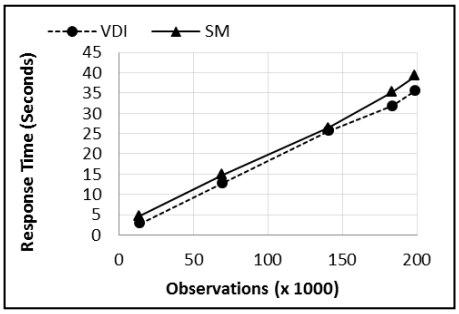
solution (denoted here VDI and described in Chapter 3) in terms of both memory usage and response time.

Notice that the VDI only attempts to solve conflicts between Properties by enabling the definition of a global Offering that subsume various local ones, which is a much restrictive approach compared to the general purpose one adopted by SM. It is also worth mentioning that VDI solution has already been evaluated with respect to data warehouse oriented solutions in chapter 3.

Both the VDI and SM implementation were deployed in an Apache Tomcat web server configured with 2GB of Java Virtual Machine memory and installed in a computer with CPU Intel Core i3(2.8GHz) and 8GB of RAM. Around 1.2 million observations of meteorological stations and 1.1 million observations of oceanographic stations were loaded in the two data sources of MeteoGalicia and INTECMAR. Microsoft SQL Server was used as the underlying



(a) Memory Usage



(b) Response Time

Figure 4.8: SOS Semantic Mediation performance evaluation

DBMS for both datasets. Five different *GetObservation* request that combine results of both datasets, with an increasing number of result observations (ranging from around 14 thousand to around 200 thousand) were executed in both implementations.

The use of both permanent and heap Java memory is increased in SM with respect to VDI. Java permanent memory is increased by a constant amount of around 10 MB due to the greater number of classes used in SM implementation, mainly to support ontology management. The use of Java heap memory is also increased for two reasons. First, the representation of SOS and O&M metadata (*Properties*, *FOIs*, *Processes* and *Offerings*) within the Apache Jena objects in SM requires more memory than the simple hash tables of Java objects used by VDI. Second, URIs used by SM are larger than the non-universal identifiers generated by VDI. Figure 4.8(a) shows the amount of heap memory used by VDI and SM during the evaluation of the requests. SM was tested both using the required URIs and using short identifiers as those used by VDI. It is noticed that the difference between SM with short identifiers and VDI is an almost constant amount of around 40 MB, therefore it is claimed that the use of semantic technology does not have an important impact in terms of memory usage. On the other hand, the impact of large URIs in memory usage clearly increases with the number of observations retrieved. However, this is a payload that has to be assumed to enable universal identifiers within the web of data.

The comparison of VDI and SM solutions with respect to response time is shown in Figure 4.8(b). It is noticed that the difference between them is of around 2 seconds and does not increases with the number of observations. This is the time payload of the reasoning algorithm

described in Subsection 4.4.2, which enables automatic semantic mediation. Such a payload is clearly dependent on the size of the base ontology (SWEET in this case) and may have an important impact in small request retrieving few result observations.

4.6 Chapter conclusions

A framework for the semantic mediation between environmental observation datasets through OGC SOS interfaces has been described. The main characteristics of the proposed solution may be resumed as follows: First, it is remarked that, to the best of found knowledge, this is the first attempt for the support of semantic integration in an SOS implementation. The framework enables domain experts to define semantic data integration knowledge that might simplify data access tasks of many users. Second, advanced semantic clients may take advantage of *Property*, *Process* and *FOI* classifications provided in the *Mediator Ontology*, to provide powerful user interfaces. Third, new applications may arise that perform semantic mediation between SOS and other semantic and linked data sources. Fourth, backward compatibility with the SOS interface is maintained, thus even standard clients will benefit from the new semantic integration capabilities. Fifth, a LAV data integration approach was enabled in the mediator, which simplifies the incorporation of new data sources. Sixth, the data source semantic mappings are defined within the scope of well-known top-level ontologies like SSN and SWEET. Regarding performance, the use of semantic technologies and representations (large URIs) has the expected impact in both memory usage and response time. Response time impact may be important if the SOS is used to reply to many requests of few observations each.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

CHAPTER 5

GENERIC WRAPPERS FOR IN-SITU AND REMOTE DEVICES

As it was already stated in the introduction, two major data modeling paradigms are used to represent the data generated by environmental sensing devices. In particular, in-situ devices generate time series that fit well classical entity-relationship (ER) models, whereas remote devices generate large spatio-temporal arrays. In spite of the above, the wrappers that address the heterogeneity problems in the mediator/wrapper architecture used in previous chapters, are provided by ad-hoc implementations.

To ease the incorporation of new data sources in the framework, this chapter describes the implementation of two generic data access wrappers. Such an implementation has already been done with the most recent 2.0 version of the SOS interface. Notice that the migration of the solutions developed in Chapters 3 and 4 to version 2.0 of SOS is straightforward in terms of research.

– A generic SOS semantic mediation wrapper for in-situ geospatial observation data sources, recorded in spatial relational DBMSs. A generic data model for environmental in-situ sensor databases is proposed. Besides, a query optimization strategy is used based on the decomposition of URIs into collections of primitive key components.

– A generic SOS semantic mediation wrapper for remote geospatial observation data sources, recorded in array data formats and accessible through NetCDFSubset stan-

dardized array data services. A global ontology for remote sensor datasets is designed based on which generic algorithms to solve SOS requests are implemented.

The reminder of the chapter is organized as follows. The design and implementation of the in-situ sensor observation wrapper is described in Section 5.1. Section 5.2 is devoted to the remote sensor observation data wrapper. The evaluation of the performance of both wrappers and a required optimization strategy are discussed in Section 5.3. Finally, Section 5.4 concludes de chapter.

5.1 In-situ sensor observation data wrapper

A generic wrapper was developed that enables SOS access to any database of in-situ observations recorded in a spatially enabled DBMS. Such wrapper will ease the incorporation of new data sources in future. Currently, the software developer needs to know deeply each data source and then code in *Java* a certain amount of classes to adapt the local model to O&M model. Using the generic wrapper, she only has to make some new SQL queries (views) to adapt the local models to the generic one, since the wrapper will be working against that model. To illustrate this, let us first describe the specific data models of the two real data sources that were used during the evaluation of the proposed solution.

Meteorological Stations

A relational database with observations of various meteorological stations [1], whose conceptual model is outlined by the UML class diagram of Figure 5.1. Observation data is generated every 10 minutes (*10MinutesData*), daily (*DalyData*) and monthly (*MonthlyData*). Each data element has a real value, a time instant and metadata represented by a *Measurement*. Each *Measurement* represents the fact that a sensing device (*Sensor*) that measures a given property (*Parameter*) is installed in a *Station* at a given *Elevation* above the soil and an aggregation process (*Function*) is next applied with a given time frequency (*Interval*). The sensing devices are classified in accordance with their nature (meteorological, oceanographic, etc.) and are represented in the model with *SensorType*. Such devices are installed in stations which all together form a network of meteorological stations (*Network*). Finally, each given property

[1] http://www2.meteogalicia.es/galego/observacion/estacions/estacions.asp

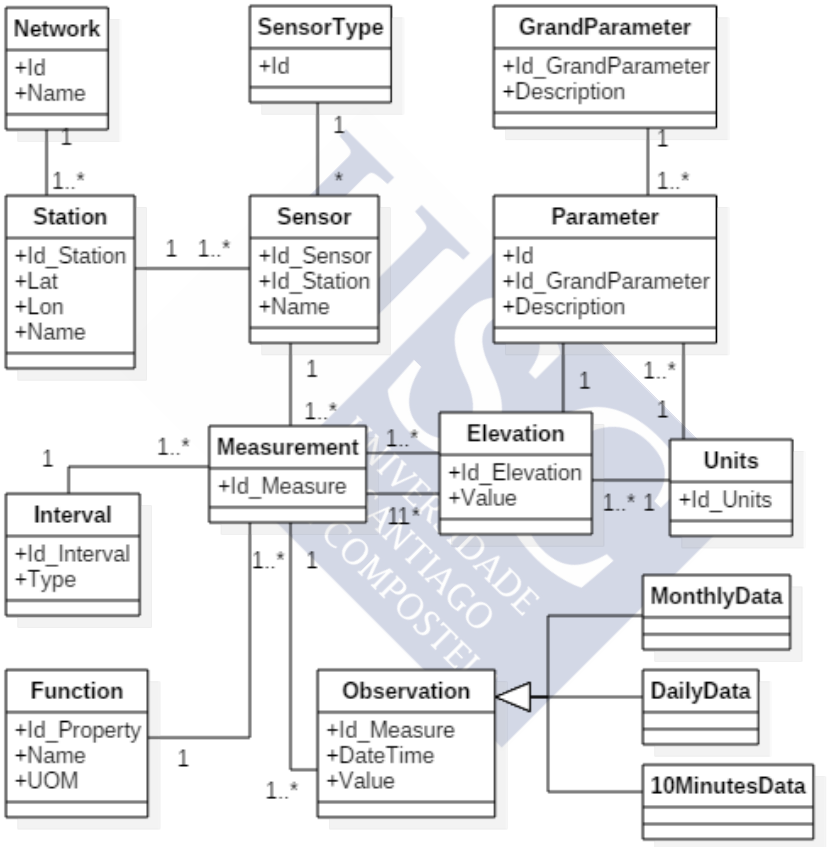


Figure 5.1: Conceptual model of meteorological stations

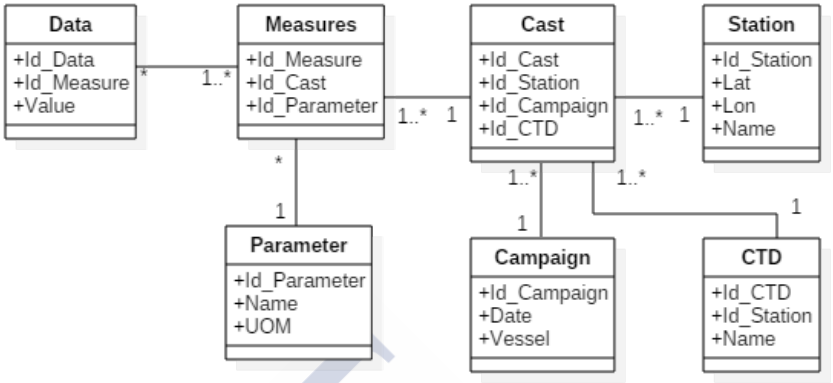


Figure 5.2: Conceptual model of CTD

can be categorized under a set of main properties (Temperature, Pressure, etc.) which are represented with *GrandParameter*.

CTD Profiles

A relational database with observations of CTD profiles [2], whose conceptual model is outlined by the UML class diagram of Figure 5.2. Each data element (*Data*) records a value, a sea depth level and a reference to a *Measurement*. A *Measurement* references a measured property (*Parameter*) and a *Profile*, which represents the use of a specific *CTD Device* at a given time instant and at a given location in the sea (*Station*).

5.1.1 Generic data model

To achieve what I explained in the first paragraph of this section, first a generic data model for the representation of in-situ sensor observation data was defined. This model enables both the generation of the required data source ontology and the implementation the SOS *GetObservation* operation. A UML class diagram of this model is given in Figure 5.3. The instances of each entity of this model are obtained by an SQL view over the specific data

[2] http://www.intecmar.org/Ctd/Default.aspx

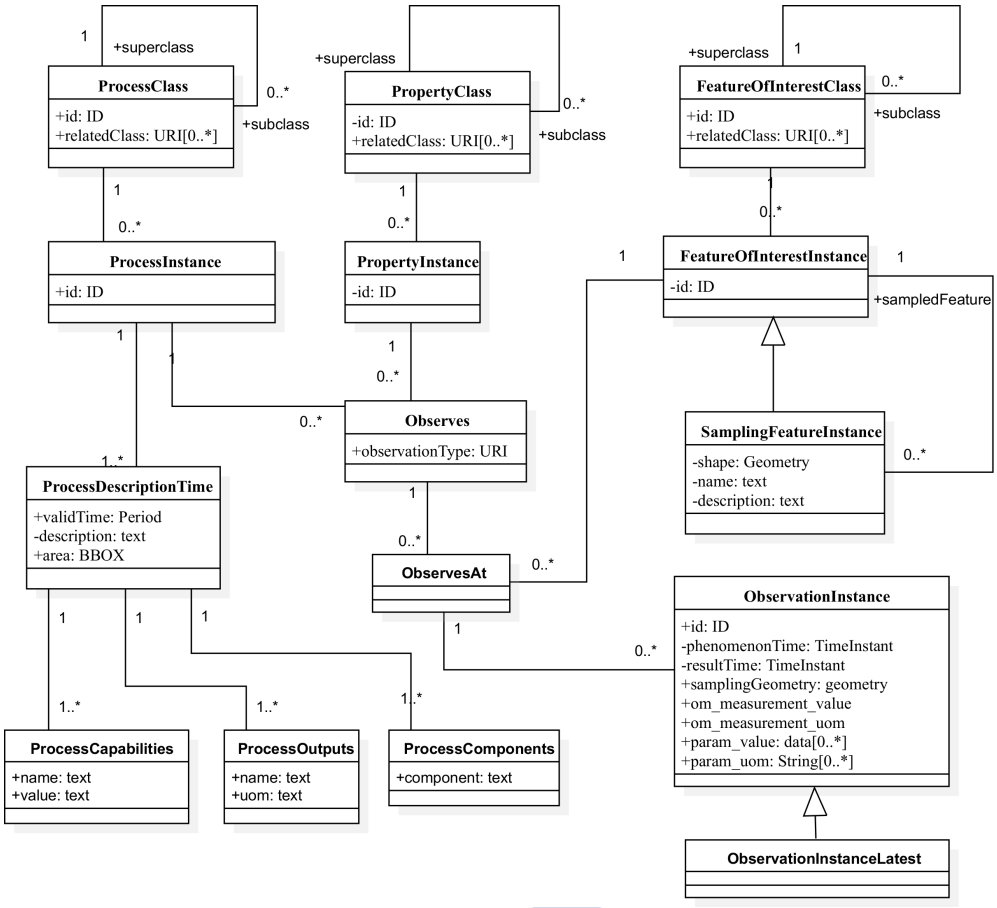


Figure 5.3: Generic conceptual model for in-situ observation databases

source model. In this way we can easily add a new data source by just providing SQL views over the local data model.

At the top of the diagram, three UML classes enable the representation of the *Process*, *Property* and *FOI* OWL classes available in the data source. Notice that for each OWL class the model provides an identifier (id), which concatenated with the data source identifier provides an URI for the class in the data source ontology. Besides, various URIs that relate the concept with other concepts of the selected well-known top-level application domain ontology (SWEET in our case) are also provided. Finally, each OWL class has also a reference to

its superclass in the model. This enables the creation of OWL class hierarchies from the data source data. In the case of the data source of meteorological stations, *Process*, *Property* and *FOIs* OWL classes are generated from *SensorType*, *GrandParameter* and *Network* entities, respectively. In the case of CTD profiles, OWL classes are not provided by the data source, thus the relevant SQL views return no data.

A relevant UML class in the model is provided to represent individuals of each of the above OWL classes. Those are the *Processes*, *Properties* and *FOIs* available in the dataset. The temporal evolution of the SensorML description of each *Process* is modeled with UML class *ProcessDescriptionTime*. The model also represents which *Processes* are used to observe each *Property*, and for each such combination at which *FOIs* observations are obtained. Finally, observations are represented by UML class *ObservationInstance*. *ObservationInstanceLatest* is used to enable more efficient access to the last observations of each *Property* obtained by each *Process* at each *FOI*, which is a typical data need in many real applications.

To illustrate how the instances of the above UML classes are generated with SQL from the specific data source data model, the SQL code of the *PropertyInstance* and *SamplingFeatureInstance* views for the data sources of meteorological stations and CTD are given below.

The query below retrieves the instances of class *PropertyInstance* for the data source of meteorological stations.

```
SELECT CAST(p.id AS VARCHAR)||"_"||replace(p.name, " ", "-")||_||
        CAST(e.id AS VARCHAR)||"_"||replace(e.elevation, " ", "-") AS
            id,
     CAST(gp.id AS VARCHAR)||"_"||gp.name AS propertyClass
FROM Parameter AS p, GrandParameter AS gp, Measurement AS m,
    Elevation AS e
WHERE p.grandParameter=gp.id AND m.parameter=p.id AND m.elevation=e.
    id
```

Each SOS *Property* is constructed by the combination of a *Parameter* measured at a given *Elevation*. Therefore, a relevant identifier is generated accordingly, concatenating appropriate keys of the database elements with other attributes that can be better interpreted by humans. For example, *Parameter* “Temperature”, whose identifier in the database is 25, measured at “10 meters’ (elevation identifier 15) produces a *Property* whose identifier is “25_Temperature_15_10-meters”.

The instances of *PropertyInstance* for the CTD data source are obtained by the following SQL statement.

```
SELECT
    CAST(Id_Parameter AS VARCHAR) ||_|| Name AS id,
   'http://purl.oclc.org/NET/ssnx/ssn#Property' AS propertyClass
FROM
   Parameter
```

In this case the property identifier does not depend on the elevation. Thus, following with the previous example, *Parameter* "Temperature", whose identifier in the database is 2, produces a *Property* whose identifier is "2_Temperature". Nevertheless, a notable difference with the previous example is that the properties in this data source do not have an associated local OWL class. As a consequence, the attribute *propertyClass*, will be always the *ssn#Property* class.

Given that both data sources have a table *Station* with identical attributes, the following SQL statement may be used to generate the instances of the *SamplingFeatureInstance* class in either of the meteorological stations or CTD data sources.

```
SELECT
  CONVERT(varchar(30),e.Id_Station)||"_"||e.Name AS id,
  geometry:: STGeomFromText('POINT(' + CAST(e.Lat AS varchar) + ' '
      + CAST(e.Lon AS varchar) + ')', 4326) AS shape,
   e.Name AS name,
   " " AS description,
   'http://cograde.usc.es/SOSSI/meteo#Galicia' AS sampledFeature,
FROM
   Station e
```

Notice that the URI of each station is obtained by the concatenation of its id and its name. The shape of the station of data type point is constructed from the Lat and Lon coordinates obtained from the database. Thus, a meterological station whose identifier is "65_SantiagoEOAS" has a shape "POINT(42.4 -8.3)" with EPSG projection 4326. Casually, the same view can also be used for CTD profiles.

5.1.2 SOS core operations evaluation

Now, SQL queries on the tables resulting from the generic data model are used to get the required data to generate the classes, individuals and restrictions of the data source ontology and to process *GetCapabilities* and *GetObservation* requests. The following lines provide an overall description of ***GetCapabilities***. Note that, in SOS 2.0, each *Offering* has one *Process*, and one *Process* may have several *Offerings*. Thus, first of all, we obtain the list of offerings

```
SELECT DISTINCT p.id
FROM ProcessInstance AS p JOIN ObservesAt AS o ON p.id = o.process
```

where an *Offering* is created for each *Process Instance*. Then, all the constraints for each offering are defined, that is, the list of *properties* and *Features of Interest* are defined. Finally, the *Spatial Extent*, *Phenomenon Time* and *Result Time* are also defined. The following example depicts the query to define the spatial extent.

```
SELECT MIN(f.shape.STY) AS lowerCornerLat, MIN(f.shape.STX) AS
    lowerCornerLon, MAX(f.shape.STY) AS upperCornerLat,
    MAX(f.shape.STX) AS upperCornerLon
FROM ObservesAt as o JOIN SamplingFeatureInstance f ON f.id=o.foi
WHERE o.process = p
```

GetObservation request that retrieves all the observations of a *Property* with identifier *prop* generated by a *Process* with identifier *proc*, during the period defined by instants *s* and *e* at FOIs located inside a given rectangle *b* is implemented with the following SQL statement.

```
SELECT oi.*
FROM ObservationInstance oi JOIN
    SamplingFeatureInstance sfi ON oi.foi = sfi.id
WHERE oi.process = proc AND oi.property = prop
  AND oi.phenomenonTime BETWEEN s AND e
  AND st_intersects(b, sfi.shape)
```

The use of geometric data types and functions (see *st_intersects* in the above query) demands from the underlying DBMSs the implementation of a relevant spatial SQL standard [35].

5.2 Remote sensor observation data wrapper

A generic wrapper was developed that enables the semantically integrated access to array datasets produced by remote sensors and published through NetCDFSubset services.

5.2.1 Raster core ontology

To achieve this, together with the URLs of the relevant NetCDFSubset services, the application domain expert must provide some important metadata. To ease the generation of the data source ontology, such metadata is represented in a compatible ontology, defined as a specialization of the Core Ontology described in Subsection 4.2.1. The base for the definition of those ontologies is the Raster Core Ontology, whose main concepts and restrictions are graphically represented in Figure 5.4.

Processes that generate the array data are represented by individuals of core#Process. A raster#ProcessDescriptionTime class is added whose individuals record sensor descriptions according to SensorML standard. Each *Offering* of the data source will be defined normally as a subclass of core#Offering, specifying with a relevant restriction the reference to its *Process* (see Subsection 4.2.1). Besides, each *Offering* will be annotated with the URL of the THREDDS data server and with a reference to a specific catalog of such server where all the datasets accessible through relevant NetCDFSubset services are included. Each such dataset is nothing but an array file recording a spatio-temporal tile of the whole *Offering* array. The variables recorded in those datasets that are going to be accessed are specified as individuals of class raster#Variable and referenced in a restriction of the form “raster#hasVariable exactly 1 {var1, var2, ..., varN}”. Each raster#Variable individual has a name in the dataset (raster#hasName) and a reference to a *Property* individual (raster#hasProperty).

Once the above subclasses, individuals and restrictions have been manually inserted by the application domain expert in the data source ontology, an algorithm is periodically executed to update such ontology with metadata obtained from the THREDDS data server, which is required to solve future *GetCapabilities* and *GetObservation* requests. In particular, first, for each *Offering* the restriction “ssn#observedProperty exactly 1 {prop1, prop2, ..., propN}” is generated, where {prop1, prop2, ..., propN} is the set of properties related to the *Variables* of the *Offering*. Next, the THREDDS catalog referenced by the *Offering* is accessed to generate a subclass of raster#Dataset for each available array file. From the metadata of the relevant NetCDFSubset service, the temporal and spatial extension are also obtained. The former

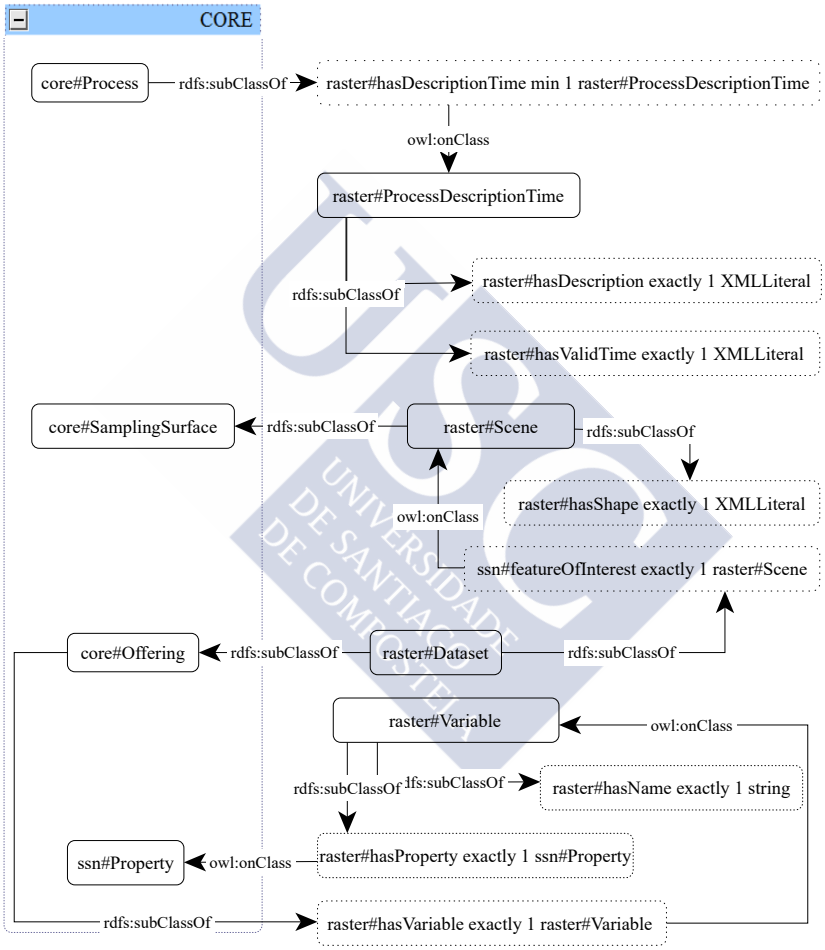


Figure 5.4: Raster Core Ontology

is used to annotate the generated raster#Dataset subclass and the later is used to generate relevant raster#Scene individuals. Each raster#Scene has a rectangular shape. Finally, for each *Offering*, the restriction "ssn#featureOfinterest exactly 1 {foi1, foi2, ..., foiN}" is generated where {foi1, foi2, ..., foiN} is the set of raster#Scene individuals related to *Dataset* subclasses of the *Offering*.

5.2.2 SOS core operations evaluation

SOS ***GetCapabilities*** requests can be evaluated by just accessing the above described generated ontology. An SPARQL implementation [3] is used by the current implementation. The algorithm was already described in Section 4.4.1.

On the other hand, the implementation of operation ***GetObservation*** consists of three steps. First, the request filter parameters are used to produce a SPARQL query that obtains the appropriate raster#Dataset classes of the ontology. An example of such query could be as follows, where *OntoNS* represents the namespace of the ontology [*http://cograde.usc.es/SOSSI/roms.owl*], and the last eight lines of code are examples of filters by *Offering*, *Process*, *ObservedProperty* and *Feature of Interest*.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX raster: <http://cograde.usc.es/SOSSI/raster#>
PREFIX core: <http://cograde.usc.es/SOSSI/core#>
PREFIX ssn: <http://purl.oclc.org/NET/ssnx/ssn#>
SELECT ?ds (str(?datasetURL) as ?dsURL) (str(?timeXML) as ?time)
    ?offering ?varName ?observedProperty ?procedure ?foi
    (str(?foiBBOX) as ?foiEnvelope)
WHERE {
  ##datasets and offerings to which they belong
  ?ds rdfs:subClassOf raster:Dataset .
  ?ds rdfs:subClassOf ?offering .
  ?offering rdfs:subClassOf core:Offering .
  FILTER NOT EXISTS { ?offering rdf:type owl:Restriction} .
```

[3] https://jena.apache.org/

```
FILTER (!regex(str(?offering),\"Dataset\")) .
##variables por dataset
?offering rdfs:subClassOf ?res .
?res rdf:type owl:Restriction .
res owl:onProperty raster:hasVariable .
?res owl:onClass ?variables .
?variables owl:oneOf ?lista .
?lista rdf:rest*/rdf:first ?variable .
?variable raster:hasName ?varName .
##filter parameter: property
?variable raster:hasProperty ?observedProperty .
 ##filter by procedure in dataset
 ?offering rdfs:subClassOf ?resP .
 ?resP rdf:type owl:Restriction .
 ?resP owl:onProperty ssn:observedBy .
 ?resP owl:onClass ?procedures .
 ?procedures owl:oneOf ?listaP .
 ?listaP rdf:rest*/rdf:first ?procedure .
 ##filter by foi in dataset (scenes)
 ?ds rdfs:subClassOf ?resF .
 ?resF rdf:type owl:Restriction .
 ?resF owl:onProperty ssn:featureOfInterest .
 ?resF owl:onClass ?scenes .
 ?scenes owl:oneOf ?listaF .
 ?listaF rdf:rest*/rdf:first ?foi .
 ?foi raster:hasShape ?foiBBOX .
 ##datasetURL
 ?ds raster:hasDatasetURL ?datasetURL .
 ##datasetTime
 ?ds core:hasPhenomenonTime ?timeXML .
  ##Added offering filter
  FILTER (regex(str(?offering),"OntoNS#OfferingRomsMeteogalicia")
      || regex(str(?offering),"OntoNS#OfferingCFSRmonNOAA")) .
  ##Added procedure filter
  FILTER (regex(str(?procedure),"OntoNS#RadarMeteogalicia")) ||
      (regex(str(?procedure),"OntoNS#RomsMeteogalicia")) .
```

```
##Added observed property filter
FILTER(regex(str(?observedProperty),"OntoNS#salt")) ||
    (regex(str(?observedProperty),"OntoNS#temp")) .
##Added feature of interest filter
FILTER
    (regex(str(?foi),"OntoNS#Scene/-11.41/-7.80/41.40/44.69""))
```

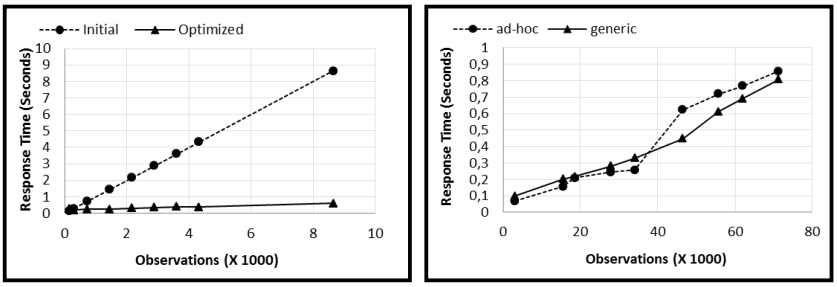
Second, a NetCDFSubset request is performed for each such dataset, using the appropriate variable names and spatial and temporal filters. This request is built with the results obtained in the *SELECT* clause of the SPARQL query. An example of NetCDFSubset request to a MeteoGalicia THREDDS server is shown below. The *Observed Properties* "temp" and "salt" are requested, specifying also the *Spatial Extent* (north, west, east, south) and the *Temporal Range* (time_start, time_end):

http://mandeo.meteogalicia.es/thredds/ncss/roms/fmrc/files/20160222/roms_002_20160222_0000.nc?var=salt&var=temp,salt&north=46.0000&west=-14.0000&east=-4.5000&south=38.0000&time_start=2016-02-22&time_end=2016-02-23&addLatLon=true&accept=netcdf

Finally, the NetCDF file obtained from the NetCDFSubset service is processed to generate an observation for each array element. Latitude, Longitude and Time dimensions are used to generate the *SamplingGeometry* parameter and the *Phenomenon* and *Result Time* stamps, respectively. All the other dimensions of the arrays are treated as additional parameters.

5.3 Framework evaluation

The efficacy and efficiency of the implementations of both wrappers were evaluated using the datasets described in subsection 5.1 (Meteorological stations and CTD). Regarding the in-situ sensor observation data wrapper, the initial direct implementation of operation *GetObservation* described in section 5.1 offered a very poor performance in terms of response time. To understand the reason of this, we have to look at the WHERE clause of the SQL statement used by the implementation. Remember that *Property* identifiers are constructed by the underlying SQL view by concatenating various components, including key and non-key attributes. Thus, for example, to check the condition "oi.property = '25_Temperature_15_10-meters' ", all the elements generated by the underlying SQL view must be scanned, given that appropriate index structures cannot be provided by the database on an attribute that has been generated



(a) In-situ meteo stations wrapper (b) Remote HF Radar wrapper

Figure 5.5: Performance evaluation

by a query. To overcome this restraint, some additional information on how the identifiers of *Processes*, *Properties*, *FOIs* and *Observations* are generated by concatenating primitive components must be provided by the application domain expert. In particular, for each such concept the expert must provide a set of tuples of the form {(position, key, type)}, which identify key components of the concept identifier that must be used in filter conditions. Thus, for example, for the above *Property* identifiers of the meteorological station dataset, the following set of tuples is provided: {(1, paramId, integer), (3, elevId, integer)}. The interpretation is that to filter *Properties* the first component of the identifier, called "paramId" of integer type and the third component, called "elevId" also of type integer must be used. Therefore, the above condition is replaced by the following one: "oi.paramId = 25 AND oi.elevId=15", as it is shown in the following sql clause:

```
SELECT oi.*
FROM ObservationInstance oi JOIN
    SamplingFeatureInstance sfi ON (oi.foi = sfi.id)
WHERE oi.paramId = 5 AND oi.elevId = 15
  AND oi.phenomenonTime BETWEEN s AND e
  AND st_intersects(b, sfi.shape)
```

Now, the database can use the indexes on relevant key attributes of tables *Parameter* and *Elevation* to speed-up the generated SQL query. A comparison of the response time of the initial and optimized versions of the implementation of the wrapper for the meteorological stations dataset is given in Figure 5.5(a).

Regarding the remote sensor observation data wrapper, the main difference between the generic implementation described in Section 5.2 and an ad-hoc implementation is that the former has to access the data source ontology with SPARQL to obtain the datasets to be queried, whereas the later does not. However, the time to access the ontology is too low in comparison with the time to access the datasets to have any incidence in the performance, as it is shown in the comparison between ad-hoc and generic implementations given in Figure 5.5(b).

5.4 Chapter conclusion

The design and implementation of generic data access wrappers for in-situ and remote sensor observation data sources was described. Those wrappers are key components of a mediator/wrapper architecture for sensor observation semantic data mediation. Generic models and ontologies are designed and based on them relevant SOS operation implementations are provided. The domain application expert has to focus now on the specificities of the data source data model and related semantics, leaving technological issues to the provided generic implementation. In particular, in the case of in-situ observation data sources recorded with relational technologies, the expert has to provide SQL views for the components of the generic data model and specify how concept URIs are decomposed into primitive key attributes. In the case of remote sensor observation data, she has to describe the *Process*, specify the *Properties* to publish and the relationships with dataset variables, and provide the required THREDDS catalog URLs. As a consequence, the development cost of data wrappers is decreased, without a sensitive impact in the system performance.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

CHAPTER 6

CONCLUSIONS AND PERSPECTIVES

The following sections present a summary of the contributions of this thesis, the conclusions as well as future research perspectives.

6.1 Summary of contributions

The achievements of the research work are summarized in this section. Firstly, the design, implementation and evaluation of a framework has been described that provides a real solution to the problem of virtual integration of heterogeneous observation data sources in environmental application domains (Chapter 3). The framework is currently being validated in two real scenarios with meteorological and oceanographic data. Advantages of the approach are the following.

- Server-side data integration leads to simpler client software that may be tailored to users with lower skills. Besides, same data integration code is not replicated in many clients.
- The proposed virtual data integration approach does not require additional data duplication and reduces the required investment in data management infrastructure.
- Efficient integration of vector and raster data is directly supported by the construction of raster data wrappers.
- The well known Mediator/Wrapper architecture paradigm [79] followed makes the framework very flexible in the incorporation of new data sources.

– Its multi-thread implementation approach for data source querying enables the framework to leverage currently available multi-core hardware architectures.

– The implementation of generic external SOS wrappers enables the deployment of SOS services in cascade, which is an interesting functionality for the construction of Spatial Data Infrastructures.

Secondly, the semantic mediation between environmental observation datasets through OGC SOS interfaces has been described (Chapter 4). The main characteristics of the proposed solution may be resumed as follows:

– It is remarked that, as far as this author knows, this is the first attempt for the support of semantic integration in an SOS implementation. The framework enables domain experts to define semantic data integration knowledge that might simplify data access tasks of many users.

– Advanced semantic clients may take advantage of *Property*, *Process* and *FOI* classifications provided in the *Mediator Ontology*, to provide powerful user interfaces.

– New applications may arise that perform semantic mediation between SOS and other semantic and linked data sources.

– Backward compatibility with the SOS interface is maintained, thus even standard clients will benefit from the new semantic integration capabilities.

– A LAV data integration approach was enabled in the mediator, which simplifies the incorporation of new data sources.

– Regarding performance, the use of semantic technologies and representations (large URIs) has the expected impact in both memory usage and response time. Response time impact may be important if the SOS is used to reply to many requests of few observations each.

Finally, the design and implementation of generic data access wrappers for in-situ and remote sensor observation data sources has been explained (Chapter 5). Those wrappers are key components of a mediator/wrapper architecture for sensor observation semantic data mediation. Generic models and ontologies are designed and based on them relevant SOS operation

implementations are provided. The domain application expert has to focus now on the specifications of the data source data model and related semantics, leaving technological issues to the provided generic implementation. In particular, in the case of in-situ observation data sources recorded with relational technologies, the expert has to provide SQL views for the components of the generic data model and specify how concept URIs are decomposed into primitive key attributes. In the case of remote sensor observation data, she has to describe the *Process*, specify the *Properties* to publish and the relationships with dataset variables, and provide the required THREDDS catalog URLs. As a consequence, the development cost of data wrappers is decreased, without a sensitive impact in the system performance.

6.2 Future research perspectives

The outcomes of the research work open some interesting research perspectives. Here, the most significant ones are outlined.

A first one is related to the efficient encoding of raster data in O&M SOS responses. The framework should detect when a compact raster encoding has better performance for each specific service response. The main disadvantage of using such compact raster representations in some responses is that the complexity of both server and clients would be increased.

Second, validation of a fully semantic SOS 2.0 operative prototype. Although a virtual integration of heterogeneous observation data sources and a semantic mediation prototype was already tested in a real scenario, would be interesting update such prototype.

The third one is related to the 5-star Open Data scheme ([37]) proposed by Tim Berners-Lee in his linked data design principles. With this thesis, the first three levels are achieved (data are available on the Web, data are structured and data are available in a non-proprietary open format), however, the fourth and fifth level (use URIs to denote things and link data to other data to provide context) would be interesting since we will obtain advantages like establishing bookmarks over FOIS, ObservedProperties, Sensors or even Observations.

Finally, looking into the possibility of extending the semantic part to add a SPARQL endpoint. This path makes me foresee that in-situ (vectorial data) and remote datasets (raster data) could be easily integrated and accessible through SPARQL queries.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Appendix A: Publications

.1 International journals

– Manuel A. Regueiro, J.R.R. Viqueira, José A Taboada, José M. Cotos. Virtual integration of sensor observation data. Computer & Geosciences (JCR Impact Factor 2016: 2.47) Elsevier ISSN: 0098-3004. August 2015.

– Manuel A. Regueiro, Jose R.R. Viqueira, Christoph Stasch, José A Taboada.Semantic Mediation fo Observation Datasets through Sensor Observation Services Future Generation Computer Systems (JCR Impact Factor 2016: 2.43) Elsevier ISSN: 0167-739X. August 2016.

.2 International conferences

– X. Méndez, J. Touriño, J. Parapar, M. Hermida, V. García, M. A. Regueiro, J.R. Viqueira, F. Landeira. MeteoSIX – Bulding a meteorological SDI for the Region of Galicia (Spain). Inspire Conference 2011. Edinburgh, Scotland.

– M. A. Regueiro, Jose R.R. Viqueira, Christoph Stasch, Jose Angel Taboada. Sensor Observation Service Semantic Mediation: Generic Wrappers for In-Situ and Remote Devices. ER 2016. The 35th International Conference on Conceptual Modeling (CORE A). Gifu, Japan

.3 National conferences

- Manuel A. Regueiro, Sebastián Villarroya, Gabriel Sanmartín, José R.R. Integración de observaciones medioambientales: Solución Inicial y retos futuros. XVII Jornadas del Ingeniería del Software y Bases de Datos – SISTEDES 2012. Almería, España.
- Sebastián Villarroya, David Mera, Manuel A. Regueiro, José M. Cotos. Diseño de Servidores de Adquisición y Publicación de Datos de Sensores. XVII Jornadas del Ingeniería del Software y Bases de Datos – SISTEDES 2012. Almería, España.
- M. A. Regueiro, J.R.R. Viqueira, C. Cortizas, P. Díaz, X. Méndez, J. Touriño, J. Parapar, F. Landeira. MeteoSIX: Difusión de datos meteorológicos y oceanográficos en MeteoGalicia. III Jornadas Ibéricas de las Infraestructuras de Datos Espaciales – JIIDE 2012. Madrid, España.
- Manuel A. Regueiro, Sebastián Villarroya, Gabriel Sanmartín, José R.R. Integración semántica de datos de observación mediante servicios SOS. XX Jornadas del Ingeniería del Software y Bases de Datos – SISTEDES 2015. Santander, España.

.4 Other publications

- Sebastián Villarroya, J.R.R. Viqueira, M. A. Regueiro, José A Taboada, José M. Cotos. SODA: A framework for Spatial Observation Data Analysis. Distributed and Parallel Databases (JCR Impact Factor 2011: 1.15) Springer ISSN: 0926-8782. November 2014.
- Sebastián Villarroya, J.R.R. Viqueira, M. A. Regueiro, José M. Cotos. Spatio-Temporal Integrated Analysis With MAPAL. ST-Analytics 2014. Advanced in Spatio-Temporal Analytics WorkShop@ICCSA 2014 (CORE C). Guimarães, Portugal.

Bibliography

[1] Hamid R. Arabnia and Rose Joshua, editors. *Proceedings of the 2005 International Conference on Artificial Intelligence, ICAI 2005, Las Vegas, Nevada, USA, June 27-30, 2005, Volume 2*. CSREA Press, 2005.

[2] Thilini Ariyachandra and Hugh Watson. Key organizational factors in data warehouse architecture selection. *Decision Support Systems*, 49(2):200 – 212, 2010.

[3] Payam M. Barnaghi and Mirko Presser. Publishing linked sensor data. In Taylor et al. [72].

[4] Pim Borst and Hans Akkermans. An ontology approach to product disassembly. In *Knowledge Acquisition, Modeling and Management, 10th European Workshop, EKAW'97, Sant Feliu de Guixols, Catalonia, Spain, October 15-18, 1997, Proceedings*, pages 33–48, 1997.

[5] Mike Botts and Alexandre Robin. *OGC SenorML: Model and XML Encoding Standard*. Open Geospatial Consortium (OGC), 2014. http://www.opengeospatial.org/standards/sensorml.

[6] Omar Boucelma, Mehdi Essid, and Zoé Lacroix. A wfs-based mediation system for gis interoperability. In *Proceedings of the 10th ACM international symposium on Advances in geographic information systems*, GIS '02, pages 23–28, New York, NY, USA, 2002. ACM.

[7] Shawn Bowers, JoshuaS. Madin, and MarkP. Schildhauer. A conceptual modeling framework for expressing observational data semantics. In Qing Li, Stefano Spaccapietra, Eric Yu, and Antoni Olivé, editors, *Conceptual Modeling - ER 2008*,

volume 5231 of *Lecture Notes in Computer Science*, pages 41–54. Springer Berlin Heidelberg, 2008.

[8] E. Bridger, L. E. Bermudez, M. Maskey, C. Rueda, B. L. Babin, and R. Blair. Oostethys - open source software for the global earth observing systems of systems. In *American Geophysical Union Fall 2009 Meeting*, 2009.

[9] Arne Bröring, Johannes Echterhoff, Simon Jirka, Ingo Simonis, Thomas Everding, Christoph Stasch, Steve Liang, and Rob Lemmens. New generation sensor web enablement. *Sensors*, 11(3):2652, 2011.

[10] Arne Bröring, Patrick Maué, Krzysztof Janowicz, Daniel Nüst, and Christian Malewski. Semantically-enabled sensor plug & play for the sensor web. *Sensors*, 11(8):7568, 2011.

[11] Arne Bröring, Christoph Stasch, and Johannes Echterhoff. *OGC Sensor Observation Service Interface Standard*. Open Geospatial Consortium (OGC), 2012. http://www.opengeospatial.org/standards/sos.

[12] Silvio D. Cardoso, Flor K. Amanqui, Kleberson J.A. Serique, José L.C. dos Santos, and Dilvan A. Moreira. Swi: A semantic web interactive gazetteer to support linked open data. *Future Generation Computer Systems*, 54:389 – 398, 2016.

[13] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The tsimmis project: Integration of heterogenous information sources. In *Information Processing Society of Japan (IPSJ 1994)*, 1994.

[14] Nengcheng Chen, Liping Di, Genong Yu, and Jianya Gong. Geo-processing workflow driven wildfire hot pixel detection under sensor web environment. *Computers & Geosciences*, 36(3):362 – 372, 2010.

[15] Nengcheng Chen, Liping Di, Genong Yu, and Min Min. A flexible geospatial sensor observation service for diverse sensor data based on web service. *{ISPRS} Journal of Photogrammetry and Remote Sensing*, 64(2):234 – 242, 2009.

[16] Michael Compton, Payam Barnaghi, Luis Bermudez, Raúl García-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort,

Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. The {SSN} ontology of the {W3C} semantic sensor network incubator group. *Web Semantics: Science, Services and Agents on the World Wide Web*, 17(0):25 – 32, 2012.

[17] Helen Conover, Gregoire Berthiau, Mike Botts, H. Michael Goodman, Xiang Li, Yue Lu, Manil Maskey, Kathryn Regner, and Bradley Zavodsky. Using sensor web protocols for environmental data acquisition and management. *Ecological Informatics*, 5(1):32 – 41, 2010.

[18] Oscar Corcho and Raúl García-Castro. Five challenges for the semantic sensor web. *Semant. web*, 1(1,2):121–125, April 2010.

[19] Simon Cox. *Observations and Measurements - XML Implementation*. Open Geospatial Consortium (OGC), 2011. http://www.opengeospatial.org/standards/om.

[20] Simon Cox. *Geographic Information - Observations and Measurements. OGC Abstract Specification Topic 20*. Open Geospatial Consortium (OGC), 2013. http://www.opengeospatial.org/standards/om.

[21] Ramanathan V. Guha Dan Brickley. Rdf schema (rdfs). https://www.w3.org/TR/rdf-schema/. Online; accessed January-2016.

[22] Deegree. deegree sensor observation service. http://wiki.deegree.org/deegreeWiki/deegree3/SensorObservationService. Online; accessed June-2015.

[23] Thomas Devogele, Christine Parent, and Stefano Spaccapietra. On spatial database integration. *International Journal of Geographical Information Science*, 12(4):335–352, 1998.

[24] Mikel Emaldi, Jon Lázaro, Unai Aguilera, Oscar Peña, and Diego López de Ipiña. Short paper: Semantic annotations for sensor open data. In Henson et al. [33], pages 115–120.

[25] Leticia I. Gómez, Bart Kuijpers, Bart Moelans, and Alejandro A. Vaisman. A survey of spatio-temporal data warehousing. *International Journal of Data Warehousing and Mining (IJDWM)*, 5(3):28–55, 2009.

[26] John Graybeal, Anthony W. Isenor, and Carlos Rueda. Semantic mediation of vocabularies for ocean observing systems. *Computers & Geosciences*, 40(0):120 – 131, 2012.

[27] Bin Guo, Zhu Wang, Zhiwen Yu, Yu Wang, Neil Y. Yen, Runhe Huang, and Xingshe Zhou. Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm. *ACM Comput. Surv.*, 48(1):7:1–7:31, August 2015.

[28] Amarnath Gupta, Richard Marciano, Ilya Zaslavsky, and Chaitanya K. Baru. Integrating gis and imagery through xml-based information mediation. In *Selected Papers from the International Workshop on Integrated Spatial Databases, Digital Inages and GIS*, ISD '99, pages 211–234, London, UK, 1999. Springer-Verlag.

[29] Denis Havlik, Thomas Bleier, and Gerald Schimak. Sharing sensor data with SensorSA and cascading sensor observation service. *Sensors*, 9(7):5493–5502, 2009.

[30] M.J. Heavner, D.R. Fatland, E. Hood, and C. Connor. Seamonster: A demonstration sensor web operating in virtual globes. *Computers & Geosciences*, 37(1):93 – 99, 2011. Virtual Globes in Science.

[31] Dennis Heimbigner and Dennis McLeod. A federated architecture for information management. *ACM Trans. Inf. Syst.*, 3(3):253–278, July 1985.

[32] Cory A. Henson, Josh K. Pschorr, Amit P. Sheth, and Krishnaprasad Thirunarayan. Semsos: Semantic sensor observation service. In *Proceedings of the 2009 International Symposium on Collaborative Technologies and Systems*, CTS '09, pages 44–53, Washington, DC, USA, 2009. IEEE Computer Society.

[33] Cory A. Henson, Kerry Taylor, and Óscar Corcho, editors. *Proceedings of the 5th International Workshop on Semantic Sensor Networks, SSN12, Boston, Massachusetts, USA, November 12, 2012*, volume 904 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[34] W.H. Inmon. *Building the Data Warehouse, 4th Edition*. Wiley, 2005.

[35] International Organization for Standardization (ISO). *Information technology – Database languages – SQL multimedia and application packages – Part 3: Spatial. ISO/IEC 13249-3:2011*, 2011.

[36] Krzysztof Janowicz and Michael Compton. The stimulus-sensor-observation ontology design pattern and its integration into the semantic sensor network ontology. In Taylor et al. [72].

[37] Krzysztof Janowicz, Pascal Hitzler, Benjamin Adams, Dave Kolas, and Charles Vardeman. Five stars of linked data vocabulary use. *Semantic Web*, 5(3):173–176, 2014.

[38] Krzysztof Janowicz, Sven Schade, Arne Bröring, Carsten Keßler, Patrick Maué, and Christoph Stasch. Semantic enablement for spatial data infrastructures. *Transactions in GIS*, 14(2):111–129, 2010.

[39] R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. Wiley computer publishing. Wiley, 2002.

[40] T. Kobialka, R. Buyya, C. Leckie, and R. Kotagiri. A sensor web middleware with stateful services for heterogeneous sensor networks. In *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pages 491–496, Dec 2007.

[41] S. Kunz, T. Uslander, and K. Watson. A testbed for sensor service networks and the fusion SOS: towards plug & measure in sensor networks for environmental monitoring with OGC standards. In *Proceedings 18th World IMACS / MODSIM Congress, Cairns, Australia, 13-17 July*, pages 973–979, 2009.

[42] Alon Y. Levy. Logic-based artificial intelligence. In Jack Minker, editor, *Logic-based Artificial Intelligence*, chapter Logic-based Techniques in Data Integration, pages 575–595. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[43] AlonY. Levy, Divesh Srivastava, and Thomas Kirk. Data model and query evaluation in global information systems. *Journal of Intelligent Information Systems*, 5(2):121–143, 1995.

[44] Steve H.L. Liang, Arie Croitoru, and C. Vincent Tao. A distributed geospatial infrastructure for sensor web. *Computers & Geosciences*, 31(2):221 – 231, 2005. Geospatial Research in Europe: {AGILE} 2003.

[45] Anice C Lowen, Samira Mubareka, John Steel, and Peter Palese. Influenza virus transmission is dependent on relative humidity and temperature. *PLoS Pathog*, 3(10):1–7, 10 2007.

[46] B. Ludascher, A. Gupta, and M.E. Martone. Model-based mediation with domain maps. In *Data Engineering, 2001. Proceedings. 17th International Conference on*, pages 81–90, 2001.

[47] Regueiro M.A., Viqueira J.R.R., Stasch C., and Taboada J.A. Sensor observation service semantic mediation: Generic wrappers for in-situ and remote devices. In *35th International Conference on Conceptual Modeling*, pages 1–8, 2016.

[48] Mapserver. Mapserver sos server. http://mapserver.org/ogc/sos_server.html. Online; accessed June-2015.

[49] David Carral Martínez, Krzysztof Janowicz, and Pascal Hitzler. A logical geo-ontology design pattern for quantifying over types. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, pages 239–248, New York, NY, USA, 2012. ACM.

[50] Guus Schreiber Mike Dean. Web ontology language (owl). https://www.w3.org/2001/sw/wiki/OWL. Online; accessed January-2016.

[51] Christoph Mülligann, Johannes Trame, and Krzysztof Janowicz. Introducing the new sim-dla semantic similarity measurement plug-in for the protégé ontology editor. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Spatial Semantics and Ontologies*, SSO '11, pages 17–24, New York, NY, USA, 2011. ACM.

[52] M. Asif Naeem, Gillian Dobbie, and Gerald Webber. An event-based near real-time data integration architecture. In *Proceedings of the 2008 12th Enterprise Distributed Object Computing Conference Workshops*, EDOCW '08, pages 401–404, Washington, DC, USA, 2008. IEEE Computer Society.

[53] S. Nativi, M. Craglia, and J. Pearlman. Earth science infrastructures interoperability: The brokering approach. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 6(3):1118–1129, June 2013.

[54] Robert Neches, Richard Fikes, Timothy W. Finin, Thomas R. Gruber, Ramesh S. Patil, Ted E. Senator, and William R. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):36–56, 1991.

[55] 52^{o} North. 52^{o} north sensor observation service. http://52north.org/communities/sensorweb/sos/index.html. Online; accessed June-2015.

[56] Kostas Patroumpas, Giorgos Giannopoulos, and Spiros Athanasiou. Towards geospatial semantic data management: Strengths, weaknesses, and challenges ahead. In *Proceedings of the 22Nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '14, pages 301–310, New York, NY, USA, 2014. ACM.

[57] Evan W. Patton, Patrice Seyed, Ping Wang, Linyun Fu, F. Joshua Dein, R. Sky Bristol, and Deborah L. McGuinness. Semanteco: A semantically powered modular architecture for integrating distributed environmental and ecological data. *Future Generation Computer Systems*, 36(0):430 – 440, 2014. Special Section: Intelligent Big Data Processing Special Section: Behavior Data Security Issues in Network Information Propagation Special Section: Energy-efficiency in Large Distributed Computing Architectures Special Section: eScience Infrastructure and Applications.

[58] Enric Plaza and V. Richard Benjamins, editors. *Knowledge Acquisition, Modeling and Management, 10th European Workshop, EKAW'97, Sant Feliu de Guixols, Catalonia, Spain, October 15-18, 1997, Proceedings*, volume 1319 of *Lecture Notes in Computer Science*. Springer, 1997.

[59] Mohammad Ebrahim Poorazizi, Steve H. L. Liang, and Andrew J. S. Hunter. Testing of sensor observation services: A performance evaluation. In *Proceedings of the First ACM SIGSPATIAL Workshop on Sensor Web Enablement*, SWE '12, pages 32–38, New York, NY, USA, 2012. ACM.

[60] Robert G. Raskin and Michael J. Pan. Knowledge representation in the semantic web for earth and environmental terminology (SWEET). *Computers & Geosciences*, 31(9):1119 – 1125, 2005.

[61] Manuel A. Regueiro, José R.R. Viqueira, Christoph Stasch, and José A. Taboada. Semantic mediation of observation datasets through sensor observation services. *Future Generation Computer Systems*, pages –, 2016.

[62] Manuel A. Regueiro, José Ramon Rios Viqueira, José A. Taboada, and José Manuel Cotos. Virtual integration of sensor observation data. *Computers & Geosciences*, 81:12–19, 2015.

[63] Markus Lanthaler Richard Cyganiak, David Wood. Resource description framework (rdf). https://www.w3.org/RDF. Online; accessed January-2016.

[64] David J. Russomanno, Cartik R. Kothari, and Omoju A. Thomas. Building a sensor ontology: A practical approach leveraging ISO and OGC models. In Arabnia and Joshua [1], pages 637–643.

[65] Villarroya S, Viqueira J.R.R., Regueiro M.A., Taboada J.A., and Cotos J.M. SODA: a framework for spatial observation data analysis. *Distributed and Parallel Databases*, 34(1):65–99, 2016.

[66] A. Sheth, C. Henson, and S.S. Sahoo. Semantic sensor web. *Internet Computing, IEEE*, 12(4):78–83, July 2008.

[67] Amit P. Sheth, Cory A. Henson, and Satya Sanket Sahoo. Semantic sensor web. *IEEE Internet Computing*, 12(4):78–83, 2008.

[68] Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22:183–236, September 1990.

[69] Stephen W. Smoliar. Michael r. genesereth and nils j. nilsson, logical foundations of artificial intelligence. *Artif. Intell.*, 38(1):119–124, 1989.

[70] Christoph Stasch, Theodor Foerster, Christian Autermann, and Edzer Pebesma. Spatio-temporal aggregation of european air quality observations in the sensor web. *Computers & Geosciences*, 47(0):111 – 118, 2012.

[71] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data Knowl. Eng.*, 25(1-2):161–197, 1998.

[72] Kerry Taylor, Arun Ayyagari, and David De Roure, editors. *Proceedings of the 3rd International Workshop on Semantic Sensor Networks, SSN 2010, Shanghai, China, November 7, 2010*, volume 668 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2010.

[73] Thomas Uslander, Patrick Jacques, Ingo Simonis, and Kym Watson. Designing environmental software applications based upon an open sensor service architecture. *Environmental Modelling & Software*, 25(9):977 – 987, 2010. Thematic issue on Sensors and the Environment – Modelling & {ICT} challenges.

[74] Ferdinando Villa, Ioannis N. Athanasiadis, and Andrea Emilio Rizzoli. Modelling with knowledge: A review of emerging semantic approaches to environmental modelling. *Environmental Modelling & Software*, 24(5):577 – 587, 2009.

[75] Sebastián Villarroya, JoséR.R. Viqueira, ManuelA. Regueiro, JoséA. Taboada, and JoséM. Cotos. Soda: A framework for spatial observation data analysis. *Distributed and Parallel Databases*, pages 1–35, 2014.

[76] Kai Walter and Edward Nash. Coupling wireless sensor networks and the sensor observation service bridging the interoperability gap. In *Proceedings of the 12th Agile International Conference on Geographic Information Science, Hannover, Germany*, pages 1–19, 2009.

[77] Ping Wang, Jin Guang Zheng, Linyun Fu, Evan W. Patton, Timothy Lebo, Li Ding, Qing Liu, Joanne S. Luciano, and Deborah L. McGuinness. A semantic portal for next generation monitoring systems. In *Proceedings of the 10th International Conference on The Semantic Web - Volume Part II*, ISWC'11, pages 253–268, Berlin, Heidelberg, 2011. Springer-Verlag.

[78] Alexander Wöhrer, Peter Brezany, and A. Min Tjoa. Novel mediator architectures for grid information systems. *Future Generation Computer Systems*, 21(1):107 – 114, 2005.

[79] Gio Wiederhold. Mediators in the architecture of future information systems. *Computer*, 25(3):38–49, March 1992.

[80] Liang Yu and Yong Liu. Using linked data in a heterogeneous sensor web: challenges, experiments and lessons learned. *Int. J. Digital Earth*, 8(1):15–35, 2015.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

List of Figures