Ús de Sensor Web Enablement per a gestió de dades marines i accés al servei d'observació de sensors a través de client Android

Joan Olivé, David Megías, Jaume Piera, Jordi Sorribas

(1) *Unidad de Tecnología Marina. UTM - CSIC. Barcelona, Spain,* http://www.utm.csic.es

(2) *Universitat Oberta de Catalunya. UOC. Barcelona, Spain,* http://www.uoc.cat

Resum:

El present treball és un estudi de l'estat de *Sensor Web Enablement* (SWE) aplicat a l'àmbit marí d'investigació oceanogràfica. Per assolir aquest objectiu s'han avaluat les diferents solucions d'arquitectura de sistemes més adients a la problemàtica indicada i s'ha dissenyat un client de *Sensor Observation Service* (SOS) per plataforma mòbil Android que permeti consultar la descripció del sensor (SensorML) i conjunts d'observacions mitjançant estàndards oberts de SWE i peticions web (SOAP). Els dissenys anteriorment indicats han permès que quedés palès l'ampli ventall de possibilitats i avantatges que s'obtenen pel fet d'emprar SWE, ja que ens permet integrar fàcilment les dades proporcionades pels sensors i les pròpies descripcions dels sensors en diferents entorns, tal i com és el cas del disseny presentat en Android, a través d'estàndards oberts independents de les múltiples especificacions de cada fabricant, en un entorn, com és l'àmbit de la investigació marina on hi ha una gran heterogeneïtat de sensors i protocols.

I-. Introducció

I.1-. Interès

En el context de la investigació oceanogràfica realitzada per vaixells intervenen una gran quantitat de sensors diferents que adquireixen dades molt variades. Aquest escenari dóna lloc a que la informació disponible es trobi dispersa i molt dependent del sensor. El que es busca amb la proposta de l'Open Geospatial Consistorium (OGC, http://www.opengeospatial.org) del Sensor Web Enablement (SWE) és la definició d'un marc de serveis sobre els sensors que facilitin la interoperabilitat, tant sobre la definició del propi sensor: que fa el sensor, a quina organització pertany, quina és la freqüència de refresc de les dades, si es tracta d'un sensor que està en moviment o bé si es tracta d'un sensor fix; com en l'accés a les dades adquirides pel sensor, ja sigui mitjançant preguntes simples del tipus retorna'm les 10 observacions més recents o bé més complexes com interrogar el servei sobre quines dades hi ha per una regió determinada de l'espai. Es tracta d'un camp de treball molt nou encara ara, però que està experimentant una gran expansió de l'estil del que han tingut els serveis sobre mapes: "web map services" i "web features services" (WMS i WFS).

Tot aquest esforç d'estandardització de les peticions i d'interactuar amb els sensors en general té una motivació molt clara que és la de garantir la interoperabilitat en un entorn on cada vegada hi ha més sensors disponibles i en el que típicament la interacció amb els mateixos ha estat sempre totalment particular i definida d'una manera purament arbitrària pel propi fabricant del sensor. El concepte d'interoperabilitat és clau en tant que és la propietat del sistema que garanteix que dos o més sistemes o components poden intercanviar informació i fer servir la informació que s'ha intercanviat.

I.2-. Marc Institucional

El present treball s'ha dut a terme dins de la Unitat de Tecnologia Marina (UTM), figura 1, que pertany a l'àrea de Recursos Naturales del Consejo Superior de Investigaciones Científicas CSIC.

Figura 1: UTM

La UTM realitza activitats de recolzament logístic i tècnic a vaixells oceanogràfics i bases polars, així com desenvolupament tecnològic en l'àmbit de les ciències marines.

Sota la responsabilitat de la UTM es troben vàries de les grans instal·lacions científiques del país, com són el vaixells oceanogràfics Garcia del Cid, Hespérides i Sarmiento de Gamboa que són laboratoris interdisciplinaris que han d'estar oberts més de 300 dies a l'any i 24 hores al dia.

Per altra banda, la base antàrtica Juan Carlos I és un laboratori dedicat en exclusiva a la ciència on no només es dóna recolzament tècnic i logístic als projectes científics sinó que es garanteix el funcionament dels serveis cada any, després d'un període de 9 mesos d'hivernada. Durant l'hivern austral es manté l'adquisició d'algunes variables físiques.

I.3-. Marc de Treball i estratègia

S'han dissenyat i desenvolupat diferents productes (*artefacts*) que han permès avaluar l'estat de l'art de Sensor Web Enablement (SWE) per a gestió de dades marines i l'accés al servei d'observació de sensors a través de client Android, com a paradigma "d'accés genèric" a les dades i la descripció dels sensors.

Aquesta estratègia ha permès aprofundir en la problemàtica associada i els possibles beneficis de treballar amb SWE aplicat a l'observació de dades de sensors marins (*Sensor Observation Services*, SOS) i a la pròpia descripció dels sensors embarcats (*Sensor Moldel Language*, SML).

Al treballar amb sensors marins ens trobem amb que moltes vegades l'accés als propis sensors i a les dades que aquests ens proporcionen no és trivial i sol implicar comunicacions satèl·lit amb un alt cost econòmic i velocitats de transferència de dades força reduïdes pels estàndards actuals.

La elaboració de models, donada la problemàtica anteriorment enunciada es considera que ha estat particularment útil, ja que a través de diagrames de flux ha permès presentar escenaris globals del funcionament de tots els sistemes SWE i sensors marins. Aquests models han permès avaluar les diferents alternatives disponibles i quines poden ser les més adients pel marc de treball de sensors marins amb comunicació satèl·lit.

L'àmbit de treball triat s'ha circumscrit a uns models en els que l'enviament de les dades dels sensors embarcats a les diferents plataformes (vaixells, boies, estacions autònomes submarines) es realitza mitjançant connexions satèl·lit, han quedat per tant fora de l'àmbit del present treball aquells escenaris en els que les dades es recullen a les diferents plataformes i de forma posterior les dades són traslladades per una persona, que per exemple accedís a una boia sense comunicació de dades amb l'exterior. Seria el cas, per exemple d'un operador que accedís físicament i de forma

mensual a un sensor i copiés les seves dades a un disc dur o a una targeta de memòria SD per posteriorment portar-la a un sistema SWE centralitzat.

L'àmbit del present treball engloba els següents models:

1. Sistemes amb connexió de dades permanent (dades en temps real). Cal remarcar que tot i que parlem de dades en temps real, en el cas de connexions satèl·lit actualment treballem amb uns temps de latència força alts.

    1.1 Estacions autònomes submarines [20].

    1.2 Plataformes amb connexió satèl·lit permanent. El cas més típic serien connexions mitjançant satèl·lits Immarsat en òrbita equatorial. [19].

2. Sistemes amb connexió de dades NO permanent (sistemes de dades diferides). Les dades no es recullen en temps real però són enviades de forma periòdica i de forma automàtica a un repositori central. En aquest cas el més típic són connexions Iridium.

Pels models citats anteriorment s'ha avaluat els avantatges i desavantatges de fer viatjar les dades a un repositori central de SOS situat a terra o bé fer les peticions XML d'inserció d'observacions directament des de les plataformes remotes aprofitant el perfil transaccional de SOS que permet inserir les dades al sistema.

II-. Sensor Web Enablement (SWE)

Cal partir de la base de que parlem de quelcom molt nou i que intenta donar solució a una problemàtica molt àmplia tant en l'abast dels diferents elements de maquinari involucrats (sensors) amb els que cal que es comuniqui, com en l'abast de la gran heterogeneïtat de la natura de les observacions i dades adquirides pels sensors, que poden ser de tot tipus: atmosfèric, terrestre o marítim. A efectes de l'estudi actual circumscriurem el nostre interès a l'àmbit marí.

SWE proporciona tota una sèrie de mecanismes estàndards que garanteixen que donat un sensor accessible a través de la xarxa s'han de poder realitzar les següents operacions:

- Descobrir sensors i sistemes de sensors (serveis de descobriment). És a dir, a l'interrogar un servei web de sensors (SWE) mitjançant peticions estàndards, aquest servei ha de retornar una resposta, també de forma estàndard, que permeti saber:
    - Quins sensors estan disponibles.
    - Quines dades ofereixen i quines són les unitats de cada mesura.
    - Amb quina freqüència es refresca la informació.
    - La posició actual de sensor i a quina organització pertany.
- Permetre un accés estandarditzat a les observacions adquirides pel sensor, de manera que es puguin recuperar subconjunts de dades ofetes pel sensor. S'ha de permetre l'accés a les dades en temps real tot i que resulta molt convenient poder accedir a tot l'històric d'observacions aplicant filtres temporals del tipus data inicial i data final i d'altres tipus de filtres com poden ser els espacials. Així per exemple en algunes implementacions podem trobar que es permet interrogar al servei preguntant pel conjunt d'observacions disponibles en una determinada àrea geogràfica, de tota manera és important remarcar que les últimes funcionalitats citades no formen part de l'estàndard OGC SWE 1.0 però que són susceptibles de ser incloses a la versió 2.0 en la que s'està treballant. És important recalcar algunes de les operacions que manquen a la versió 1.0 de

SWE i sobre les que hi ha més consens que caldria incorporar a la nova versió:

- Petició que retorni les n últimes observacions (getLastObservations).
- Petició que permeti actualitzar la posició d'un sensor (updateSensor).
- Petició que permeti desregistrar un sensor associat a un servidor SOS. (unregisterSensor).
- El servei responsable de realitzar aquesta tasca es coneix com a Sensor Observation Service (SOS) i actualment junt amb el SensorML constitueixen les dues parts més madures de SWE.
- Permetre subscriurie's a un servei de publicació d'alertes, Sensor Alarm Service (SAS) de manera que l'usuari en funció d'uns valors donats de les observacions pugui rebre aquests avisos.

II.1-. Web Semàntica

L'altre concepte que resulta clau per entendre com s'articula SWE és el de la web semàntica:

"*The Semantic Web, as described by the W3C Semantic Web Activity, is an evolving extension of the World Wide Web in which the semantics, or meaning, of information on the Web is formally defined*" (An Ontological Representation of Time Series Observations on the Semantic Sensor Web , 2009 [6])

De la Web Semàntica cal remarcar els següents punts clau:

- La informació es codifica en fitxers que fan servir identificadors XML autodescriptius de manera que es fa possible l'ús de processadors estàndards de XML que interpretin la informació.
- Els identificadors codifiquen propietats i s'implementen mitjançant Resource Description Framework (RDF). Cada RDF està format per tres parts: subjecte, verb i un objecte i, cadascuna d'aquestes parts es defineix mitjançant uniform resource identifier (URI) que ha de ser accessible a través de la web.
- Es fan servir ontologies de manera que permeten les interrelacions entre identificadors.

En aquest punt cal destacar el projecte "The Marine Metadata Interoperability (MMI)" [30], amb el seu desenvolupament d'una ontologia per dispositius, sensors i xarxes de sensors. Per la implementació d'aquesta ontologia s'ha emprat W3C Web Ontology Language (OWL) per tal d'implementar un model conceptual i vocabularis controlats per tal de descriure diferents tipus de sensors amb característiques i atributs molt diferents. Aquesta ontologia resulta de gran utilitat com a base per tal de descriure les metadades de diferents sensors (en el present treball: sensor de dades meteorològiques, sensor de dades de termosalinitat), la relació dels sensors amb la plataforma on es troben (en el present treball: els vaixells d'investigació oceanogràfica Hespérides i Sarmiento de Gamboa).

Un concepte que cada vegada està guanyant més importància dins de la web semàntica aplicada a sensors és el de les anotacions.

"*Specifically, we are herein concerned with a form of semantic service discovery. The idea is that each Web service of interest is annotated with (an abstract representation of) its meaning – what does it do? – and services are discovered by matching this annotation against a discovery query – what kind of service is wanted? – given in the same logic." (Ontology-based Integration of Sensor Web Services in Disaster Management, 2009* [1]).

Emprar anotacions permet la interoperabilitat entre serveis SOS fent ús d'anotacions breus que permetin descobrir el servei SOS i el que ofereixen.

SWE agrupa tota una sèrie d'especificacions de manera que cadascuna d'aquestes dóna resposta a una part del problema que es vol abordar.

"*The SWE information model comprises a set of standards which define data models primarily for the encoding of sensor observations as well as sensor metadata*" (New Generation Sensor Web Enablement, 2011 [2]).

Les especificacions relacionades són les següents:

- Sensor Model Language (SensorML): proporciona la descripció del sensor, així com la seva posició i estat.
- Sensor Observation Service (SOS): servei web que fa accessible tant la descripció dels sensors com les observacions realitzades. A través del perfil transaccional és possible interactuar directament amb el servei SOS de manera que a través de peticions SOAP es poden afegir nous sensors (registerSensor) i inserir mesures (InsertObservation).
- Sensor Planning Service (SPS): permet controlar els sensors i realitzar accions tals com definir tasques, comprovar el seu estat o modificar-les.

- Sensor Alert Service (SAS): és el servei que implementa alertes en temps real mitjançant extensible messaging and presence protocol(XMPP).
- Web Notification Services (WNS): és el servei que permet notificacions asíncrones dels events que es produeixen als sensors.
- Observations & Measurements Schema (O&M): proporciona els models i les codificacions XML per les observacions que seran servides pels diferents serveis associats: SOS, SAS o WNS.
- Transducer Markup Language (TransducerML or TML): permet interoperar sistemes de sensors heterogenis.

Hi han diferents iniciatives per agrupar i descriure d'una manera sistemàtica els sensors i les observacions que realitzen així com per crear ontologies que relacionin els diferents models:

"*The observation model of the SSN ontology, is similar to the O&M model, with observations of properties of particular features of interest being made at paticular times and yielding a result of a value for the property. The SSN model describes the sensor (similar to procedure in O&M) that made the observation*" (The Stimulus-Sensor-Observation Ontology Design Pattern and its Integration into the Semantic Sensor Network Ontology, 2010 [9])

Una altra iniciativa força interessant és la que ha portat a terme W3C Semantic Sensor Network Incubator Group:

"The group had two main objectives. The first was to develop an ontology to describe sensors and sensor networks for use in sensor network and sensor web applications. The second was to study and recommend methods for using the ontology to semantically enable applications developed according to available standards such as the Open Geospatial Consortium's (OGC) Sensor Web Enablement (SWE) standards." (Semantics and Sensors, 2011 [10])

Actualment 52North és responsable del que podem considerar que són les implementacions més avançades de SWE. Concretament podem trobar implementacions de SOS, SAS, WNS, SPS. D'ara en endavant ens centrarem en les implementacions de SOS i SensorML, que van íntimament lligades.

En el moment actual podem trobar d'altres implementacions però cap d'elles presenta el grau de maduresa de la de 52 North. A tall d'exemple podem citar les més importants:

- oostethys (http://www.oostethys.org). Implementació de SOS.
- MapServer (http://www.mapserver.org). Implementació de SOS.

El present treball es centra en les dues primeres que són les que en el moment actual, 2011, presenten un grau més alt de maduresa tot i que l'abast de la feina que queda per fer encara és gran.

II.2-. Open Geospatial Consortium (OGC)

OGC és un consorci format per més de 370 companyies, agències governamentals i universitats. L'objectiu d'OGC és el de promoure i dirigir el desenvolupament i l'harmonització d'estàndards espacials oberts. Va ser fundada l'any 1994 amb l'objectiu de fomentar la indústria al voltant dels sistemes d'informació geogràfica (SIG) i fomentar la interoperabilitat entre sistemes, components i serveis de informació geogràfica

OGC ha elaborat un marc de referència pel conjunt d'estàndards que fan possible interconnectar els diferents sensors de manera que es faci possible la seva explotació des d'Internet. En el moment actual la versió més recent que defineix SWE és la 1.0 i s'està treballant en la elaboració de la 2.0 que intenta donar solució a les diferents mancances que s'ha anat detectant a la versió anterior.

"*The goal of the OGC Sensor Web Enablement initiative is to enable the creation of web-accessible sensor assets through common interfaces and encodings" (Ontology-based Integration of Sensor Web Services in Disaster Management, 2009* [1]).

II.3-. Sensor Obsevation Service (SOS)

Permetre un accés estandarditzat a les observacions adquirides pel sensor, de manera que es puguin recuperar subconjunts de dades ofertes pel sensor. S'ha de permetre l'accés a les dades en temps real tot i que resulta molt convenient poder accedir a tot l'històric d'observacions aplicant filtres temporals del tipus data inicial i data final i d'altres tipus de filtre com poden ser els espacials. Així per exemple en algunes implementacions podem trobar que es permet interrogar al servei preguntant pel conjunt d'observacions disponibles en una determinada àrea geogràfica. De tota manera és important remarcar que les últimes funcionalitats citades no formen part de l'estàndard OGC SWE 1.0 però que són susceptibles de ser incloses a la versió 2.0 en la que s'està treballant. És important remarcar algunes de les operacions que manquen a la versió 1.0 de SWE i sobre les que hi ha més consens que caldria incorporar a la nova versió:

- Petició que retorni les n últimes observacions (getLastObservations).
- Petició que permeti actualitzar la posició d'un sensor (updateSensor).
- Petició que permeti desregistrar un sensor associat a un servidor SOS. (unregisterSensor).

El servei responsable de realitzar aquesta tasca es coneix com a Sensor Observation Service (SOS) i actualment junt amb el SensorML constitueixen les dues parts més madures de SWE.

Els majors avenços s'han realitzat en la implantació de serveis SOS i en la descripció de sensors a través de SOS de manera que sigui possible la publicació de dades de sensors automàtics possibilitant l'accés a aquesta informació. Depenent de la implantació hi ha hagut un enfocament pur a temps real en el que únicament es pot accedir a les últimes observacions, com és el cas de oostethys (http://www.oostethys.org) o una aproximació més ambiciosa en la que a banda de poder accedir a les observacions que s'estan produint en aquell moment és possible consultar tot l'històric de dades, com és el cas de la que podem considerar actualment com la implementació de refe-

rència, la de 52 North (http://52north.org/). En aquest últim cas es fa necessari emmagatzemar tot l'històric d'observacions en una base de dades. Cal remarcar que donat el caràcter incipient de la tecnologia SOS és possible experimentar problemes de rendiment si es treballa amb volums grans d'informació.

La majoria d'arquitectures que implementen SOS, seguint la directiva genèrica marcada per INSPIRE es poden dividir en tres capes:

- Capa d'aplicació: es fa servir per publicar-insertar les observacions del sensor. Per exemple una aplicació Java
- Capa de Servei: el servei SOS pròpiament dit. Per exemple SOS de 52 North
- Capa de recursos: on s'emmagatzemen les dades. Per exemple Postgres.

*"INSPIRE es el acrónimo de Infraestructure for Spatial Information in Europe, (infraestructura de información espacial europea). No obstante, es mucho más *http://inspire.jrc.ec.europa.eu porque establece un marco de desarrollo común para las diferentes IDE de cada uno de los Estados miembros. "* (Servicios OGC, 2011 [4]).

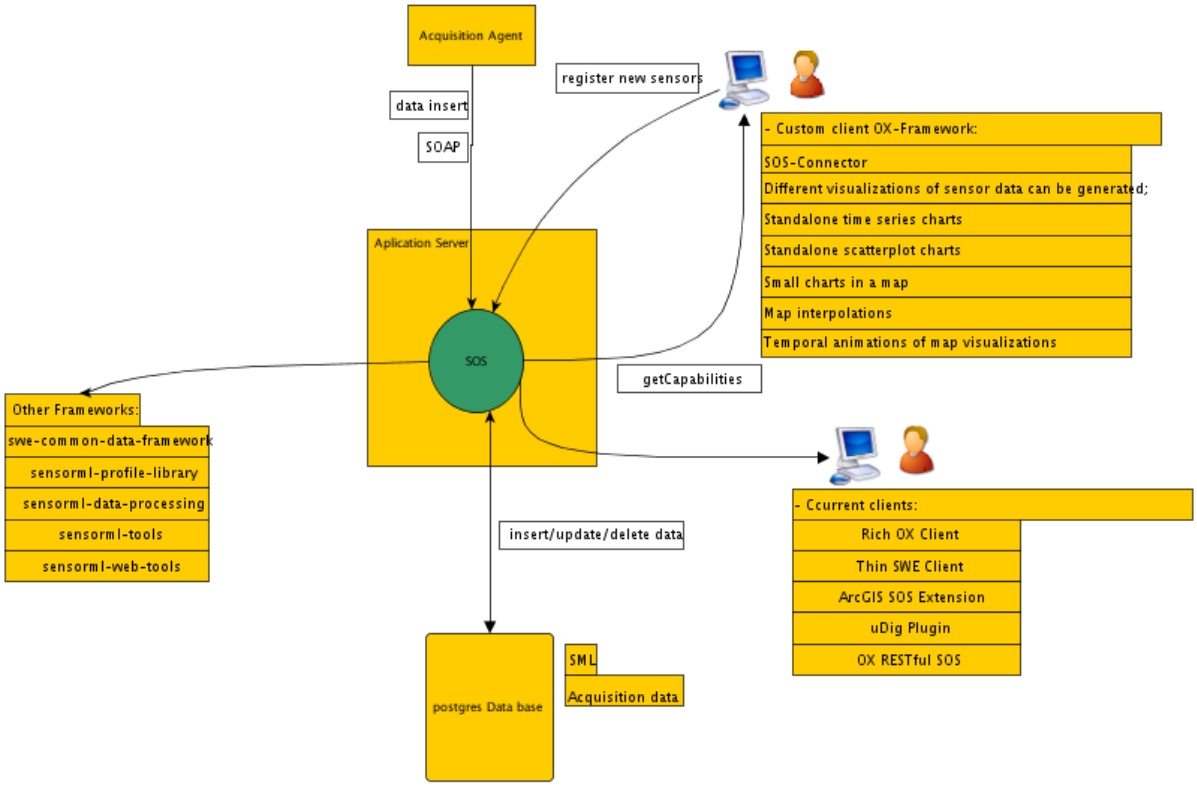

Figura 2: Implementacions SOS i funcionament

A la figura anterior [Figura 2: Implementacions SOS i funcionament] podem veure de forma esquemàtica l'arquitectura SOS i les diferents implementacions actuals.

A la part central tenim el servei SOS que típicament es desplega en un servidor d'aplicacions (Tomcat per exemple).

A la part superior podem veure Acquisition Agent, que correspon a un mecanisme o aplicació externa encarregada d'inserir les dades observades pels sensors al servei SOS, a grans trets trobem dues maneres de procedir:

- Es genera un fitxer de text amb un format estàndard que posteriorment es serveix per al servei SOS. És el cas de oostethys. Es tracta d'implementacions menys madures i no fan ús del perfil transaccional de SOS.

- Es treballa amb aplicacions encarregades de llegir les dades adquirides pels sensors i que interactuant directament amb el servei SOS insereixen les dades. Aprofiten el perfil transaccional, que permet fer insercions i actualitzar algunes dades del sensor. És el cas de la implementació de 52North. A la part inferior de la figura 2 podem veure com

aprofitant el perfil transaccional s'insereixen tant les observacions (Acquisition Data) com els propis sensors (SML) al servei SOS. Concretament tota aquesta informació, en el cas de 52 North s'emmagatzema en una base de dades PostgresSQL.

Un operador especialitzat pot realitzar la operació d'afegir nous sensors al servei SOS a més de poder interactuar amb ell de manera que obtingui les dades registrades pel sensor.

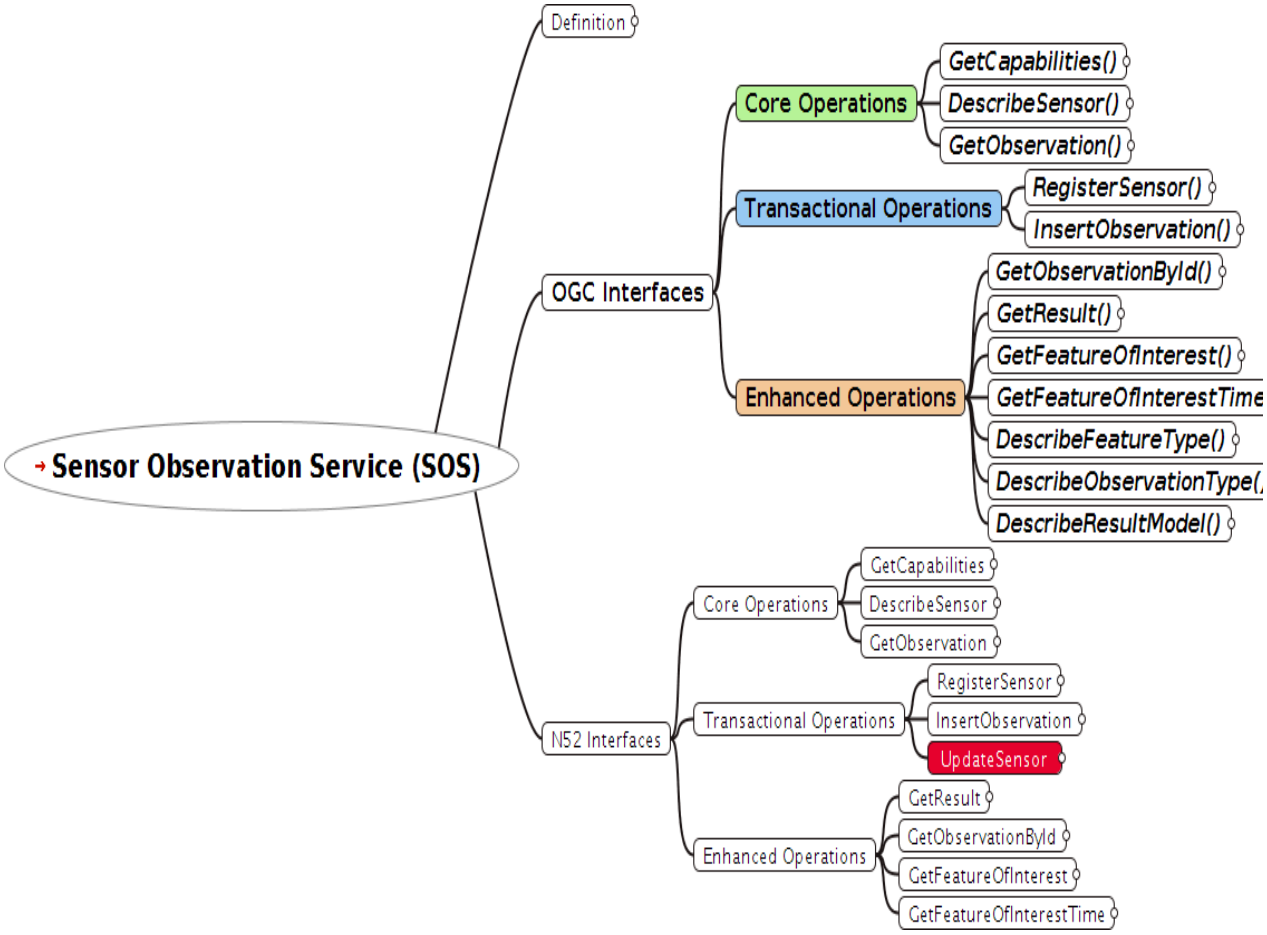

Figura 3: SOS operacions de l'estàndard OGC i implementació 52North

A la figura anterior [Figura 3: SOS operacions de l'estàndard OGC i implementació 52North] es poden veure de forma resumida les interfícies SOS definides per OGC i quines implementa 52 North:

- OG Interfaces:
  - Core Operations: agrupa les operacions principals i bàsiques que tot servei SOS ha de implementar per complir l'estàndard.
    - GetCapabilities: retorna una descripció del servei. La organització que serveix les dades, quins sensors hi ha disponibles i quin conjunt d'observacions ofereix cada sensor.
    - Describeensor: al ser interrogat sobre un sensor en concret (obtingut prèviament amb describeSensor) retorna una descripció complerta del sensor en SensorML
    - GetObservation: retorna un subconjunt d'observacions d'un sensor ofert pel servei SOS.
  - Transactional Operations: són un subconjunt d'operacions que permeten interactuar directament amb el servei SOS i realitzar operacions que d'altra forma caldria realitzar a través de fitxers de configuració per exemple, tal i com fa oostethys que no implementa aquest perfil. Aquest subconjunt d'interfícies SOS estan definides dins de l'estàndard però no cal que siguin implementades obligatòriament a l'implementar un servei SOS, ja que són considerades opcionals.
    - RegisterSensor: permet afegir un nou sensor al servei SOS a través d'una petició SOAP amb el SensorML que descriu el sensor.
    - InsertObservation: afegeix una observació al servei SOS.
  - Enhanced Operations: a l'igual que les anteriors, són opcionals. Permeten retornar tant observacions com descripcions dels propis sensors, mitjançant peticions més avançades.

- N52 Interfaces: la implementació de 52North evidentment té en compte les interfícies principals (core operations) i n'incorpora alguna més.
  - Transactional Operations: destaca l'adició de l'operació UpdateSensor que permet actualitzar la posició del sensor i el seu estat. Aquesta operació es considera imprescindible per sensors mòbils (SensorMobile).
    - RegisterSensor:
    - InsertObservation:
    - UpdateSensor:
  - Enhanced Operations:
    - GetResult:
    - GetObsevationByID:
    - GetFeatureOfInterest
    - GetFeatureOfInterestTime

A continuació es mostra la petició SOAP d'inserció d'un registre de meteorologia:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><InsertObservation
xmlns="http://www.opengis.net/sos/1.0" xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:om="http://www.opengis.net/om/1.0"
xmlns:ows="http://www.opengis.net/ows/1.1" xmlns:sa="http://www.opengis.net/sampling/1.0"
xmlns:sos="http://www.opengis.net/sos/1.0" xmlns:swe="http://www.opengis.net/swe/1.0.1"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" service="SOS" version="1.0.0" xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosInsert.xsd http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd http://www.opengis.net/om/1.0
http://schemas.opengis.net/om/1.0.0/extensions/observationSpecialization_override.xsd">

 <AssignedSensorId>urn:ogc:object:feature:Sensor:IFGI:sado-hes-termosal</AssignedSensorId>

 <om:Observation>

  <om:samplingTime>
   <gml:TimePeriod xsi:type="gml:TimePeriodType">
    <gml:beginPosition>2011-01-28T11:28:42</gml:beginPosition>
    <gml:endPosition>2011-01-28T11:28:42</gml:endPosition>
   </gml:TimePeriod>
  </om:samplingTime>

  <om:procedure xlink:href="urn:ogc:object:feature:Sensor:UTM:sado-hes-meteo"/>

  <om:observedProperty>
   <swe:CompositePhenomenon dimension="1" gml:id="met">
    <gml:name>resultComponents</gml:name>
    <swe:component xlink:href="urn:ogc:data:time:iso8601"/>
    <swe:component xlink:href="urn:SDN:P021:51:EWSB"/>
    <swe:component xlink:href="urn:SDN:P021:51:CDTA"/>
    <swe:component xlink:href="urn:SDN:P021:51:CHUM"/>
    <swe:component xlink:href="urn:SDN:P021:51:CSLR"/>
    <swe:component xlink:href="urn:SDN:P021:51:CAPH"/>
    <swe:component xlink:href="urn:SDN:P021:51:TEMP"/>
   </swe:CompositePhenomenon>
  </om:observedProperty>

  <om:featureOfInterest>
   <gml:FeatureCollection>
    <gml:featureMember>
     <sa:SamplingPoint gml:id="urn:ogc:object:feature:Sensor:UTM:sado-hes-meteo">
```

```
    <gml:name>urn:ogc:object:feature:Sensor:UTM:sado-hes-meteo</gml:name>
    <sa:sampledFeature xlink:href=""/>
    <sa:position>
     <gml:Point>
      <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">-7.9616603 -29.3117371</gml:pos>
     </gml:Point>
    </sa:position>
   </sa:SamplingPoint>
  </gml:featureMember>
 </gml:FeatureCollection>
</om:featureOfInterest>

<om:result>
 <swe:DataArray>

  <swe:elementCount>
   <swe:Count>
    <swe:value>8</swe:value>
   </swe:Count>
  </swe:elementCount>

  <swe:elementType name="Components">
   <swe:SimpleDataRecord>
    <swe:field name="Time">
     <swe:Time definition="urn:ogc:data:time:iso8601"/>
    </swe:field>
    <swe:field name="feature">
     <swe:Text definition="urn:ogc:data:feature"/>
    </swe:field>
    <swe:field name="meteo_direcion_viento">
     <swe:Quantity definition="urn:SDN:P021:51:EWSB">
      <swe:uom code=""/>
     </swe:Quantity>
    </swe:field>
    <swe:field name="meteo_temperatura_aire">
     <swe:Quantity definition="urn:SDN:P021:51:CDTA">
      <swe:uom code="celsius"/>
     </swe:Quantity>
    </swe:field>
    <swe:field name="meteo_humedad">
```

```
      <swe:Quantity definition="urn:SDN:P021:51:CHUM">
       <swe:uom code="%"/>
      </swe:Quantity>
     </swe:field>
     <swe:field name="meteo_radiacion_solar">
      <swe:Quantity definition="urn:SDN:P021:51:CSLR">
       <swe:uom code="WM2"/>
      </swe:Quantity>
     </swe:field>
     <swe:field name="meteo_presion_atm">
      <swe:Quantity definition="urn:SDN:P021:51:CAPH">
       <swe:uom code="hPa"/>
      </swe:Quantity>
     </swe:field>
     <swe:field name="meteo_temperatura_Agua">
      <swe:Quantity definition="urn:SDN:P021:51:TEMP">
       <swe:uom code="C"/>
      </swe:Quantity>
     </swe:field>

    </swe:SimpleDataRecord>
   </swe:elementType>

   <swe:encoding>
    <swe:TextBlock blockSeparator=";" decimalSeparator="." tokenSeparator=","/>
   </swe:encoding>

                        <swe:values>2011-01-28T11:28:42,urn:ogc:object:feature:Sensor:UTM:sado-hes-
meteo,1,2,3,4,5,6</swe:values>

  </swe:DataArray>
 </om:result>
</om:Observation>
</InsertObservation>
```

Tal i com es pot veure la petició SOAP al servei SOS és força llarga i té una gran sobrecàrrega de metadades, fet que s'agreuja al ser una operació que s'ha de repetir cada vegada que hi han noves observacions.

II.3.1-. Clients SOS

Actualment hi ha poques implementacions de clients SOS. A continuació es citen les dues que es consideren més madures i que es centren en representar les observacions quedant en un segon pla la representació dels propis sensors i que constitueix l'objecte de bona part de l'estudi posterior del present projecte.

- Client OpenLayers [Figura 4: Client SOS OpenLayers.]: desenvolupat en JavaScript, presenta poques funcionalitats però permet representar la informació juntament amb altres capes i serveis GIS com *Web Map Service* (WMS) o *Web Feature Service* (WFS). (http://openlayers.org/)

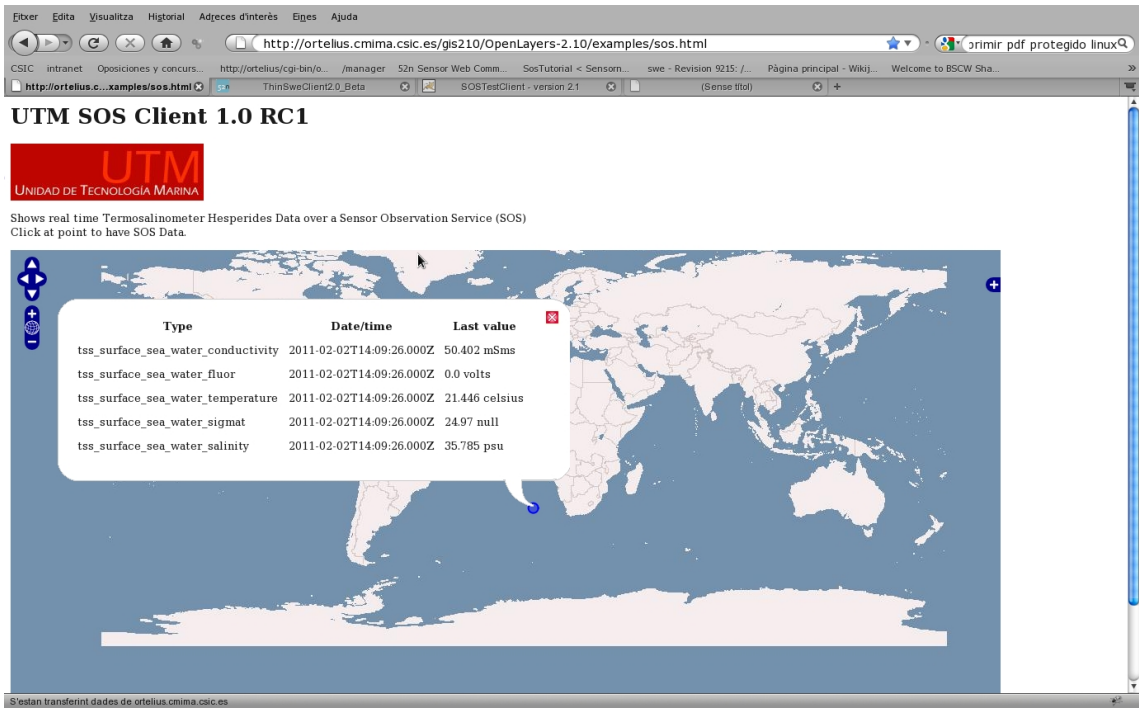

Figura 4: Client SOS OpenLayers.

- ThinClient de 52Nortg [Figura 5: Client SOS ThinClient]: presenta un gran nombre de funcionalitats, com exportació de subsets de dades a pdf o full de càlcul. Permet filtrar les observacions observades per data. Presenta el problema de que pel cas de sensors mòbils, com és el cas dels que es troben sobre vaixells d'investigació oceanogràfica, no és capaç de recollir ni representar el canvi de posició del sensor de manera que en tot moment el sensor es representa a la posició inicial on s'ha realitzat la inserció del sensor. Aquest problema no el tenim en OpenLayers. (http://52north.org/communities/sensorweb/clients/Thin_SWE_Client/index.html)

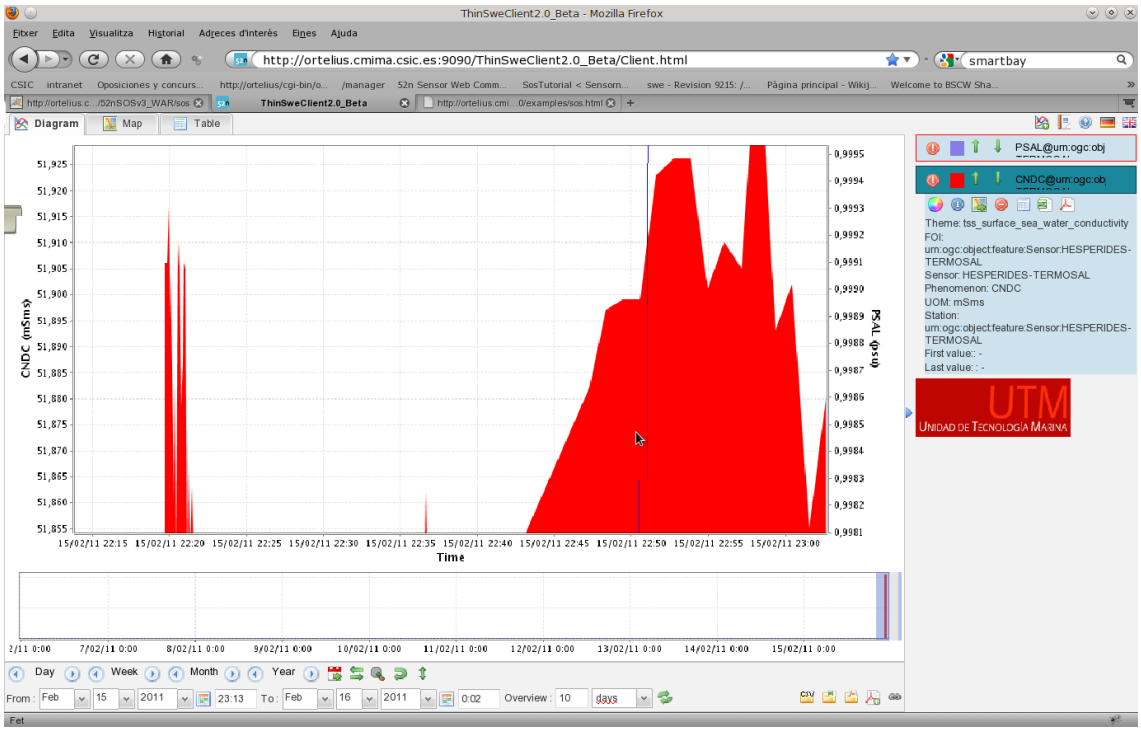

Figura 5: Client SOS ThinClient

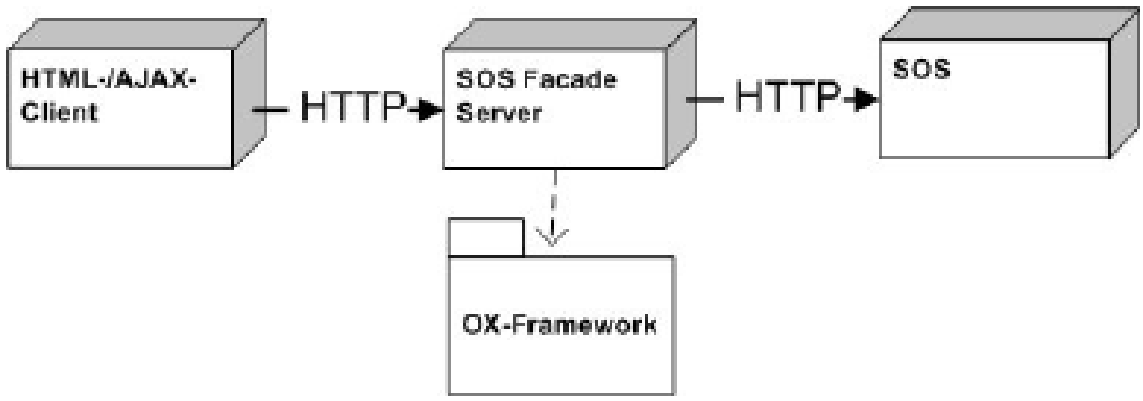

Figura 6: "Development of Sensor Web Applications with Open Source Software", 2009 [3]

A la figura 6 ("Development of Sensor Web Applications with Open Source Software", 2009 [3]), es mostra l'arquitectura del Thinclient on la interfície de l'aplicació està desenvolupada en HTML/AJAX que es comunica amb la part servidora, típicament desplegada en un servidor d'aplicacions com Tomcat i aquesta és la que interactua amb el servei SOS. Per implementar l'aplicació desplegada al servidor d'aplicacions es treballa amb el OX-Framework de 52 North.

A banda dels dos clients anteriors a la figura 2 podem veure altres clients, però que no es consideren tan evolucionats:

- Rich OX Client: client de 52North en el moment actual s'ha aturat la seva evolució, sent la última versió disponible de 2008.

  (http://52north.org/communities/sensorweb/clients/Rich_OX_Client/index.html)

- ArcGIS SOS extension: funciona com a pluggin d'ArcGIS també de 52 North. Tal com s'indica a la web de 52North es troba en una fase molt preliminar de desenvolupament, fase beta i no he estat capaç de connectar amb un servei SOS.

(http://52north.org/communities/sensorweb/clients/ArcGIS_SOS_Extension/index.html)

- UDIG Plugin: també de 52North. La última versió és de 2009 i funciona com a extensió de uDIG.

(http://52north.org/communities/sensorweb/clients/uDig_SOS_Plugin/index.html).

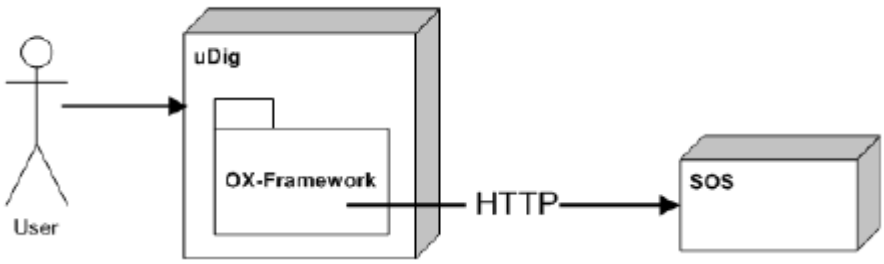

Figura 7: "Development of Sensor Web Applications with Open Source Software", 2009 [3]

A la figura 7**,** es pot mostra l'arquitectura de uDig on els usuaris interactuen amb el plugin SOS de uDig, desenvolupat amb el OX-Framework el qual a través http interactua amb el servei SOS.

II.4-. Sensor Model Language (SensorML.)

A efectes d'emmagatzemar tota la informació que descriu el sensor, que ofereix i els processos que hi tenen lloc es fa servir un llenguatge espacial: el Sensor Model Language (SensorML).

SensorML permet descriure d'una manera estandarditzada els sensors i els processos que hi tenen lloc. "*For the description of sensor metadata, the SWE framework defines the Sensor Model Language (SensorML). SensorML 1.0 "* (*New Generation Sensor Web Enablement*, 2011 [2]).

Donada la complexitat que poden assolir els sensors i els diferents aspectes que cal tenir en compte per tal de descriure els sensors d'una manera el més exacta possible hi ha un gran interès per assolir aquesta fita:

"*One aspect of describing sensors are the "technical" aspects. Information about the sensor's calibration information, its temporal resolution (sampling frequency – how often does it measure) are as relevant as what it measures (i.e. categorising sensors) as well as its accuracy. "* (*The Semantic Sensor Network Ontology: A Generic Language to Describe Sensor Assets* , 2009 [7])

Es considera que SensorML pot ser el llenguatge adient per assolir aquesta tasca.

A continuació es mostra un fitxer SensorML que descriu un termosalinòmetre.

```
<sml:SensorML version="1.0.1">
  <sml:member>
    <sml:System xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <!--sml:identification element must contain the ID of the sensor-->
      <sml:identification>
        <sml:IdentifierList>
          <sml:identifier>
            <sml:Term definition="urn:ogc:def:identifier:OGC:uniqueID">
              <sml:value>urn:ogc:object:feature:Sensor:IFGI:sado-hes-termosal
              </sml:value>
            </sml:Term>
          </sml:identifier>
        </sml:IdentifierList>
      </sml:identification>
      <!--
        sml:capabilities element has to contain status and mobility
        information
      -->
      <sml:capabilities>
        <swe:SimpleDataRecord>
          <!--
            status indicates, whether sensor is collecting data at the
            moment (true) or not (false)
          -->
          <swe:field name="status">
            <swe:Boolean>
              <swe:value>true</swe:value>
            </swe:Boolean>
          </swe:field>
          <!--
            status indicates, whether sensor is mobile (true) or fixed
            (false)
          -->
          <swe:field name="mobile">
            <swe:Boolean>
              <swe:value>true</swe:value>
            </swe:Boolean>
          </swe:field>
```

```
            </swe:SimpleDataRecord>
          </sml:capabilities>
          <!-- last measured position of sensor -->
          <sml:position name="sensorPosition">
            <swe:Position referenceFrame="urn:ogc:def:crs:EPSG:4326">
              <swe:location>
                <swe:Vector gml:id="STATION_LOCATION">
                  <swe:coordinate name="easting">
                    <swe:Quantity>
                      <swe:uom code="degree" />
                      <swe:value>-9.52</swe:value>
                    </swe:Quantity>
                  </swe:coordinate>
                  <swe:coordinate name="northing">
                    <swe:Quantity>
                      <swe:uom code="degree" />
                      <swe:value>42.90</swe:value>
                    </swe:Quantity>
                  </swe:coordinate>
                  <swe:coordinate name="altitude">
                    <swe:Quantity>
                      <swe:uom code="m" />
                      <swe:value>0</swe:value>
                    </swe:Quantity>
                  </swe:coordinate>
                </swe:Vector>
              </swe:location>
            </swe:Position>
          </sml:position>
          <!-- list containing the input phenomena for this sensor system -->
          <sml:inputs>
            <sml:InputList>
              <sml:input name="tss_surface_sea_water_salinity">
                <swe:ObservableProperty
                  definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_salinity" />
              </sml:input>
              <sml:input name="tss_surface_sea_water_temperature">
                <swe:ObservableProperty
                  definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_temperature" /
>
```

```
            </sml:input>
            <sml:input name="tss_surface_sea_water_fluor">
              <swe:ObservableProperty
                definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_fluor" />
            </sml:input>
            <sml:input name="tss_surface_sea_water_conductivity">
              <swe:ObservableProperty
                definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_conductivity" /
>
            </sml:input>
            <sml:input name="tss_surface_sea_water_Sigmat">
              <swe:ObservableProperty
                definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_Sigmat" />
            </sml:input>
          </sml:InputList>
        </sml:inputs>
        <!--
          list containing the output phenomena of this sensor system;
          ATTENTION: these phenomena are parsed and inserted into the
          database; they have to contain offering elements to determine
          the correct offering for the sensors and measured phenomena
        -->
        <sml:outputs>
          <sml:OutputList>
            <sml:output name="tss_surface_sea_water_salinity">
                                                                <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_salinity">
              <gml:metaDataProperty>
                <offering>
                  <id>tss_surface_sea_water_salinity</id>
                  <name>tss_surface_sea_water_salinity</name>
                </offering>
              </gml:metaDataProperty>
              <swe:uom code="psu" />
            </swe:Quantity>
            </sml:output>
            <sml:output name="tss_surface_sea_water_temperature">
                                                                <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_temperature">
              <gml:metaDataProperty>
                <offering>
```

```
                    <id>tss_surface_sea_water_temperature</id>
                    <name>tss_surface_sea_water_temperature</name>
                   </offering>
                  </gml:metaDataProperty>
                  <swe:uom code="celsius" />
                 </swe:Quantity>
                </sml:output>
                <sml:output name="tss_surface_sea_water_fluor">
                                                                    <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_fluor">
                  <gml:metaDataProperty>
                   <offering>
                    <id>tss_surface_sea_water_fluor</id>
                    <name>tss_surface_sea_water_fluor</name>
                   </offering>
                  </gml:metaDataProperty>
                  <swe:uom code="volts" />
                 </swe:Quantity>
                </sml:output>
                <sml:output name="tss_surface_sea_water_conductivity">
                                                                    <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_conductivity">
                  <gml:metaDataProperty>
                   <offering>
                    <id>tss_surface_sea_water_conductivity</id>
                    <name>tss_surface_sea_water_conductivity</name>
                   </offering>
                  </gml:metaDataProperty>
                  <swe:uom code="mSms" />
                 </swe:Quantity>
                </sml:output>
                <sml:output name="tss_surface_sea_water_sigmat">
                                                                    <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:tss_surface_sea_water_sigmat">
                  <gml:metaDataProperty>
                   <offering>
                    <id>tss_surface_sea_water_sigmat</id>
                    <name>tss_surface_sea_water_sigmat</name>
                   </offering>
                  </gml:metaDataProperty>
                  <swe:uom code="" />
```

```
            </swe:Quantity>
          </sml:output>
        </sml:OutputList>
      </sml:outputs>
      <!--
        description of components of this sensor system; these are
        currently not used by the 52N SOS
      -->
      <sml:components>
        <sml:ComponentList>
          <sml:component name="surface_sea_water_temperature_Sensor">
            <sml:Component>
              <sml:identification>
                <sml:IdentifierList>
                  <sml:identifier>
                    <sml:Term definition="urn:ogc:def:identifier:OGC:uniqueID">
                      <sml:value>urn:ogc:object:feature:Sensor:surface_sea_water_temperature_sensor
                      </sml:value>
                    </sml:Term>
                  </sml:identifier>
                </sml:IdentifierList>
              </sml:identification>
              <sml:inputs>
                <sml:InputList>
                  <sml:input name="surface_sea_water_temperature_hes">
                    <swe:ObservableProperty
                      definition="urn:ogc:def:phenomenon:OGC:1.0.30:surface_sea_water_temperature"
/>
                  </sml:input>
                </sml:InputList>
              </sml:inputs>
              <sml:outputs>
                <sml:OutputList>
                  <sml:output name="surface_sea_water_temperature_hes">
                    <swe:Quantity
                      definition="urn:ogc:def:phenomenon:OGC:1.0.30:surface_sea_water_temperature"
>
                      <swe:uom code="celsius" />
                    </swe:Quantity>
                  </sml:output>
```

```
                </sml:OutputList>
              </sml:outputs>
            </sml:Component>
          </sml:component>
        </sml:ComponentList>
      </sml:components>
    </sml:System>
  </sml:member>
</sml:SensorML>
```

Tal i com es pot observar es tracta d'un fitxer força extens i complex pel que es considera clau de cara a l'usuari final, el poder treballar amb representacions el més entenedores possibles de tota aquesta informació.

III-. Desenvolupament

III.1-. Arquitectura

S'han avaluat dues arquitectures diferents per tal d'avaluar els avantatges i desavantatges de fer viatjar les dades a un repositori central de SOS situat a terra o bé fer les peticions XML d'inserció d'observacions directament des de les plataformes remotes aprofitant el perfil transaccional de SOS que permet inserir les dades al sistema.

En els següents diagrames (Figura 8: Architecture 1 i Figura 9: Architecture 2) es mostra de forma esquemàtica les dues arquitectures estudiades.

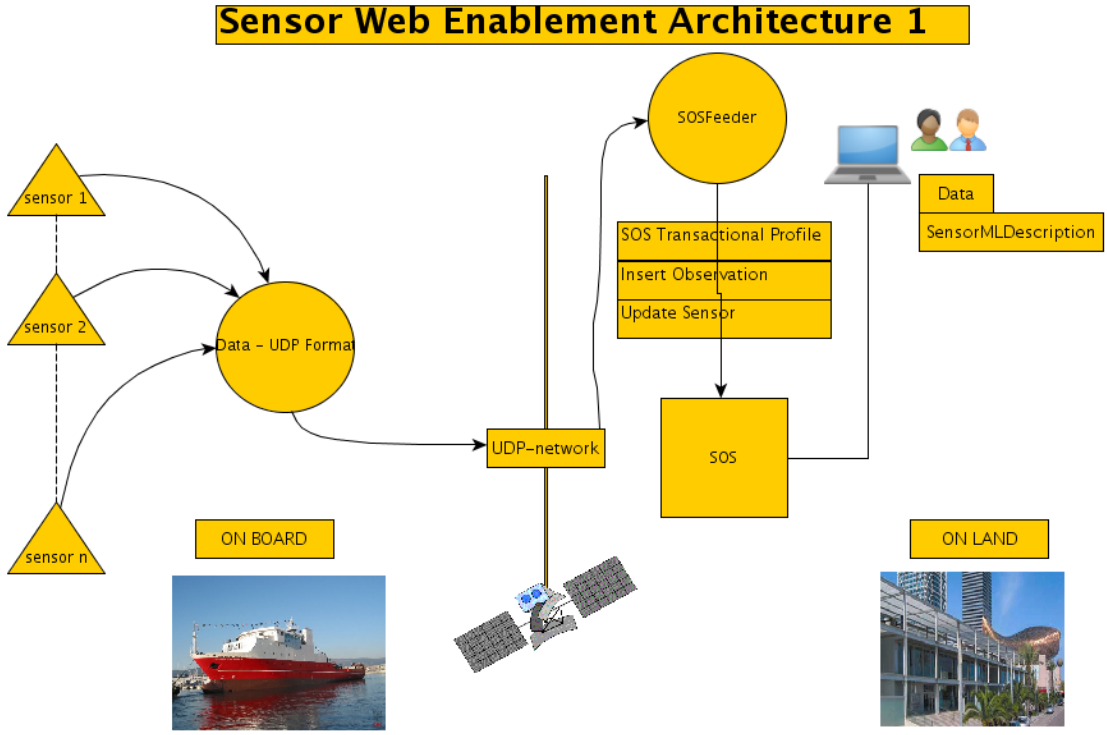

Figura 8: Architecture 1

A la figura 8 es mostra com en aquesta solució de disseny les dades que envien els diferents sensors a dins de les plataformes són normalitzades i es transmeten dins de

la xarxa utilitzant trames UDP. Aquesta part queda fora del marc de treball del present treball. A continuació les trames UDP mitjançant l'enllaç satèl·lit arriben a l'estació central a terra i és una vegada a terra que les trames de dades són processades pel prototipus encarregat d'inserir les dades al sistema SOS a través de peticions SOAP.

Una vegada a terra les dades i les descripcions dels sensors són accessibles a través de peticions SOAP al mateix servei SOS.

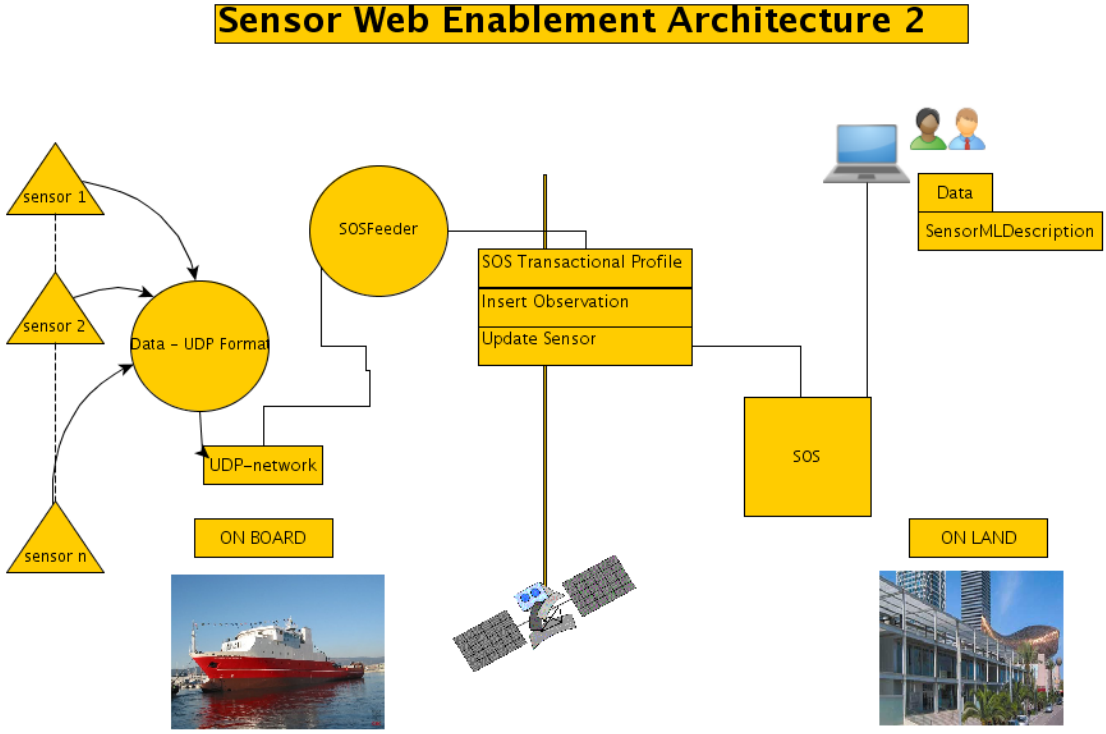

Figura 9: Architecture 2

A la figura 9 es mostra com en aquesta solució de disseny les dades que envien els diferents sensors a dins de les plataformes són normalitzades i es transmeten dins de la xarxa utilitzant trames UDP. En aquest segon model el SOSFeeder (veure punt III.4) envia les peticions SOAP mitjançant l'enllaç satèl·lit, arriben a l'estació central a terra on hi ha el servei SOS que les processarà.

A l'igual que al primer model una vegada a terra les dades i les descripcions dels sensors són accessibles a través de peticions SOAP al mateix servei SOS.

Per tal de generar les dades necessàries per avaluar els models anteriors s'ha procedit a dissenyar un "alimentador d'observacions" (*SOSFeeder*) que aprofitant l'estat de l'art actual de SOS permetés afegir dades al sistema.

III.2-. Programari

El programari que s'ha utilitzat per configurar l'entorn bàsic de treball ha estat el següent:

- Màquina virtual Centos sobre VirtualBox.
- Base de dades Postgres.
- Servidor d'aplicacions Tomcat.
- Servidor web Apache.
- Sensor Observation Service de 52 North desplegat sobre Tomcat.
- Servei estàndard de catàleg (CWS) Geonetwork [26].

S'ha creat aquest escenari bàsic per tal d'avaluar les solucions a nivell d'arquitectura proposades al llarg del projecte.

Per tot el projecte s'ha fet servir la implementació de l'estàndard SOS de 52North (http://52north.org/) donat que actualment és la més madura i la única que implementa en perfil transaccional que ens permet interactuar directament amb el sistema a l'hora d'afegir observacions i actualitzar dades del sensor.

III.3-. Peticions SOS

A continuació es detalla de forma ordenada les diferents peticions involucrades en un servei SOS per passar a continuació a un estudi més en detall en el que s'avaluïn quins punts són susceptibles de millora.

1 - A la figura 11 es mostra el diagrama de seqüència del funcionament de les peticions entre un client que actua tant com a productor de dades i com a consumidor. S'observa com es pot accedir a un servei de descobriment de catàleg (*catalog web Service*, CWS). En el present treball com a serveis de catàleg lliure i de programari lliure s'ha triat Geonetwork [26] per fer les proves. A través d'una petició estàndard OGC getRecords el servei de catàleg retorna els serveis SOS que té indexats, és a dir les metadades que descriuen el serveis SOS que permetran que el client una vegada rebi la resposta *SOS Instances*, pugui interrogar un servei SOS en concret.

2 - A continuació el client a través d'una petició getCapabilities interroga el servei SOS. Una vegada sap quins sensors té el servei SOS, entre d'altra informació recuperada en la petició anterior, el client a través d'una petició describeSensor recupera la informació del sensor. Per poder finalment recuperar les observacions del sensor que són del seu interès.

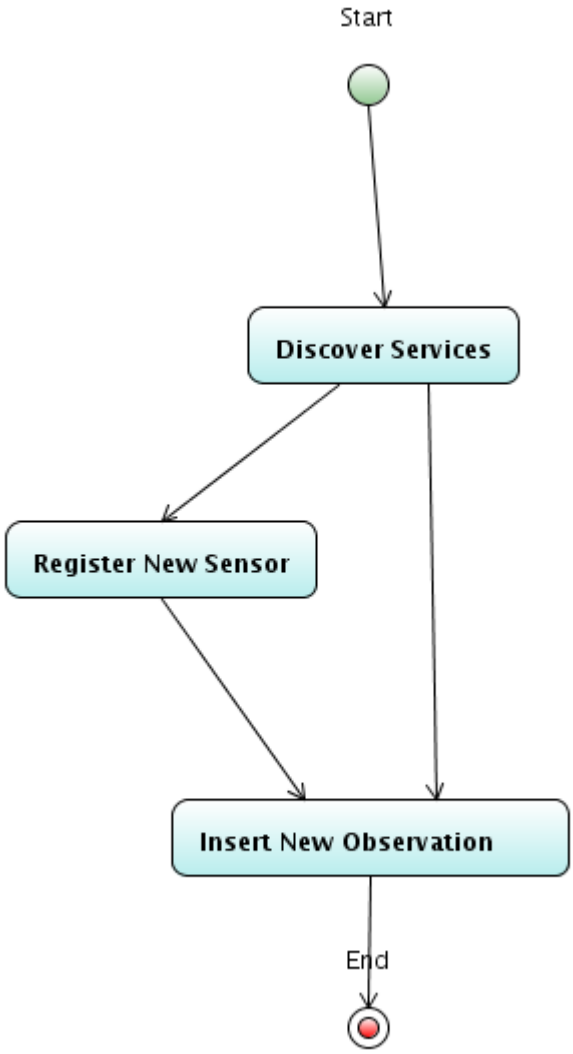

Figura 10: transactional Profile 1

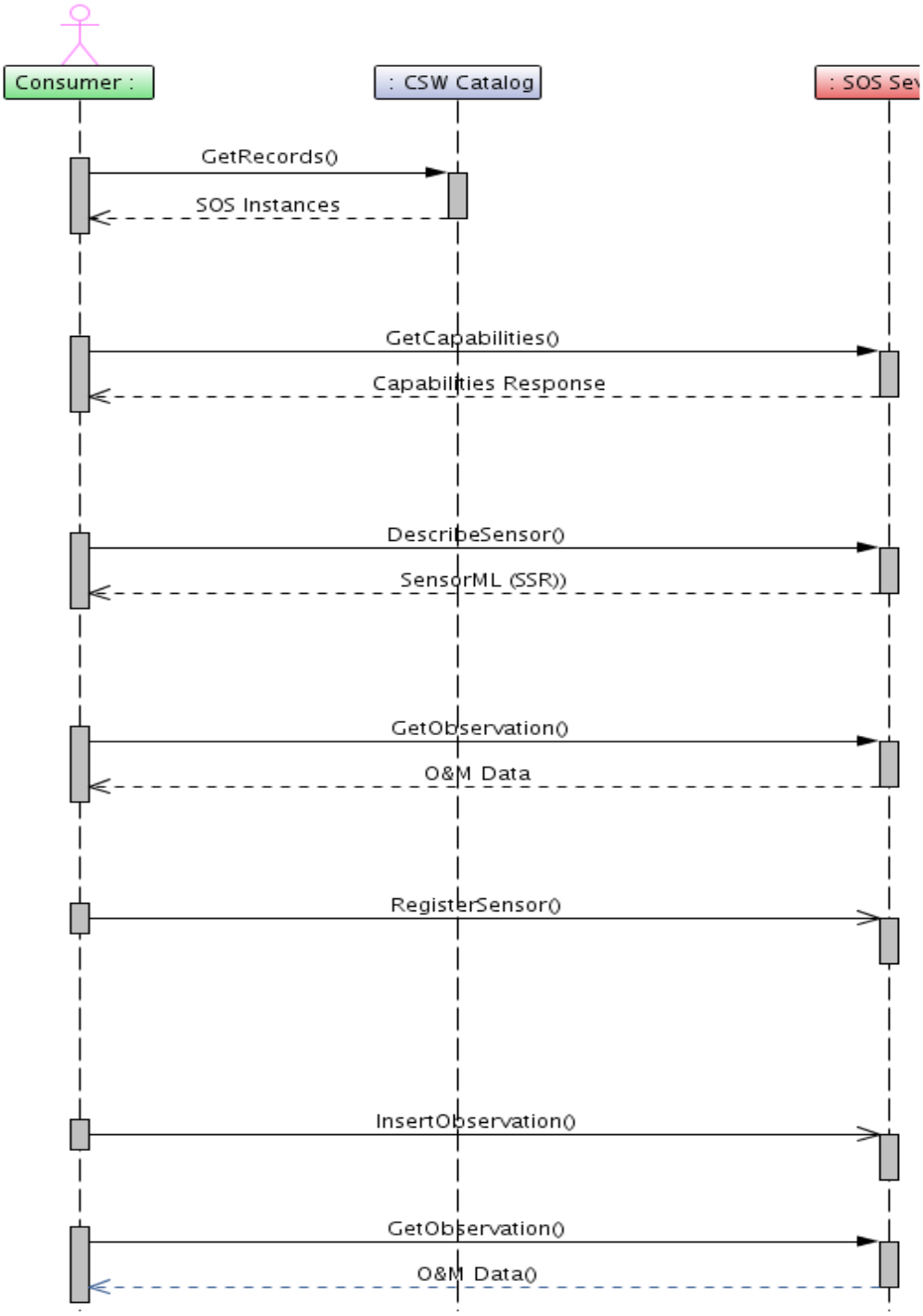

Figura 11: transactional profile 2

3 - A les figures 10 i 11 es pot veure com a través del perfil transaccional també el client pot registrar un sensor directament i inserir observacions. En aquest punt és rellevant apuntar la necessitat de protegir els sistemes SOS de peticions no desitjades, tant peticions de recuperació d'informació, que moltes vegades només ha de ser accessible per grups autoritzats d'usuaris, com de peticions de registre de sensors i d'inserció d'informació. Cal remarcar la importància del projecte Web Enforcement Service (WSS) de 52North: [27], tot i que encara es troba en una fase molt inicial de desenvolupament i disseny . A la figura 12 es pot veure com totes les peticions a serveis OGC als diferents serveis, ja siguin WMS, WFS, WCS o pel cas que ens ocupa, serveis SOS, passen a través d'un servei intermedi que el protegeix d'accesos indesitjats. En cas contrari el servei es trobaria obert a tot tipus d'accesos tant a l'hora de consultar les dades com a afegir nous sensors al sistema o modificar el seu estat .

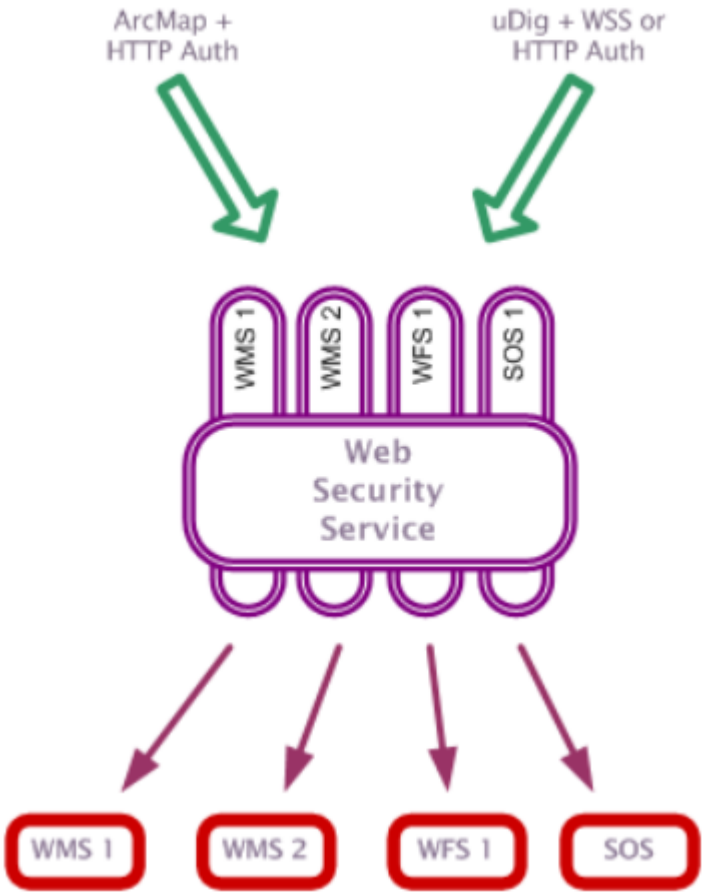

Figura 12: WSS

A continuació podem veure les peticions SOS que s'han dissenyat en el present treball:

III.3.1-. RegisterSensor().

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
 xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:sos="http://www.opengis.net/sos/1.0"
 xmlns:ows="http://www.opengis.net/ows/1.1">
 <env:Header>
  <wsa:To>http://www.52north.org/services/sos</wsa:To>
  <wsa:Action>http://www.opengis.net/sos/1.0#RegisterSensor
  </wsa:Action>
  <wsa:ReplyTo>
   <wsa:AddressTo>http://my.client.com/client/myReceiver</wsa:AddressTo>
  </wsa:ReplyTo>
 </env:Header>
 <env:Body>
  <sos:RegisterSensor service="SOS" version="1.0.0" mobileEnabled="true"
   xmlns="http://www.opengis.net/sos/1.0" xmlns:swe="http://www.opengis.net/swe/1.0.1"
   xmlns:ows="http://www.opengeospatial.net/ows" xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
   xmlns:om="http://www.opengis.net/om/1.0" xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosRegisterSensor.xsd
http://www.opengis.net/om/1.0
http://schemas.opengis.net/om/1.0.0/extensions/observationSpecialization_override.xsd">
   <!--
    Sensor Description parameter; Currently, this has to be a sml:System
   -->
   <sos:SensorDescription>
    <sml:SensorML version="1.0.1">
     <sml:member>
      <sml:System xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

       <!--sml:identification element must contain the ID of the sensor-->
       <sml:identification>
        <sml:IdentifierList>
         <sml:identifier>
          <sml:Term definition="urn:ogc:def:identifier:OGC:uniqueID">
           <sml:value>urn:ogc:object:feature:Sensor:SARMIENTO-METEO
```

```
            </sml:value>
          </sml:Term>
        </sml:identifier>
                        <sml:identifier name="longName">
                          <sml:Term definition="urn:ogc:def:identifier:OGC:1.0:longName">
                            <sml:value>Meteo en Sarmiento de Gamboa - UTM</sml:value>
                          </sml:Term>
                        </sml:identifier>

      </sml:IdentifierList>
    </sml:identification>
    <!--
      sml:capabilities element has to contain status and mobility
      information
    -->
    <sml:capabilities>
      <swe:SimpleDataRecord>
        <!--
          status indicates, whether sensor is collecting data at the
          moment (true) or not (false)
        -->
        <swe:field name="status">
          <swe:Boolean>
            <swe:value>true</swe:value>
          </swe:Boolean>
        </swe:field>
        <!--
          status indicates, whether sensor is mobile (true) or fixed
          (false)
        -->
        <swe:field name="mobile">
          <swe:Boolean>
            <swe:value>true</swe:value>
          </swe:Boolean>
        </swe:field>
      </swe:SimpleDataRecord>
    </sml:capabilities>

    <!-- last measured position of sensor -->
    <sml:position name="sensorPosition">
      <swe:Position referenceFrame="urn:ogc:def:crs:EPSG::4326">
```

```
        <swe:location>
          <swe:Vector gml:id="SARMIENTO-METEO">
            <swe:coordinate name="easting">
              <swe:Quantity>
                <swe:uom code="degree" />
                <swe:value>-9.52</swe:value>
              </swe:Quantity>
            </swe:coordinate>
            <swe:coordinate name="northing">
              <swe:Quantity>
                <swe:uom code="degree" />
                <swe:value>42.90</swe:value>
              </swe:Quantity>
            </swe:coordinate>
            <swe:coordinate name="altitude">
              <swe:Quantity>
                <swe:uom code="m" />
                <swe:value>0</swe:value>
              </swe:Quantity>
            </swe:coordinate>
          </swe:Vector>
        </swe:location>
      </swe:Position>
    </sml:position>

    <!-- list containing the input phenomena for this sensor system -->
    <sml:inputs>
      <sml:InputList>
        <sml:input name="meteo_average_speed_wind">
          <swe:ObservableProperty
            definition="urn:SDN:P021:51:EWSB" />
        </sml:input>

        <sml:input name="meteo_instantaneous_speed_wind">
          <swe:ObservableProperty
            definition="urn:SDN:P021:51:EWSB" />
        </sml:input>

        <sml:input name="meteo_wind_direction">
          <swe:ObservableProperty
            definition="urn:SDN:P021:51:EWSB" />
```

```
      </sml:input>
      <sml:input name="meteo_air_temperature">
       <swe:ObservableProperty
        definition="urn:SDN:P021:51:CDTA" />
      </sml:input>
      <sml:input name="meteo_humidity">
       <swe:ObservableProperty
        definition="urn:SDN:P021:51:CHUM" />
      </sml:input>
      <sml:input name="meteo_solar_radiation">
       <swe:ObservableProperty
        definition="urn:SDN:P021:51:CSLR" />
      </sml:input>
      <sml:input name="meteo_atmospheric_pressure">
       <swe:ObservableProperty
        definition="urn:SDN:P021:51:CAPH" />
      </sml:input>
      <sml:input name="meteo_water_temperature">
       <swe:ObservableProperty
        definition="urn:SDN:P021:51:TEMP" />
      </sml:input>
    </sml:InputList>
  </sml:inputs>
  <!--
    list containing the output phenomena of this sensor system;
    ATTENTION: tsdge phenomena are parsed and inserted into the
    database; they have to contain offering elements to determine
    the correct offering for the sensors and measured phenomena
  -->
  <sml:outputs>
    <sml:OutputList>
      <sml:output name="meteo_wind_direction">
        <swe:Quantity definition="urn:SDN:P021:51:EWSB">
          <gml:metaDataProperty>
            <offering>
              <id>meteo_wind_direction</id>
              <name>meteo_wind_direction</name>
            </offering>
          </gml:metaDataProperty>
          <swe:uom code="" />
        </swe:Quantity>
```

```
        </sml:output>
        <sml:output name="meteo_air_temperature">
          <swe:Quantity definition="urn:SDN:P021:51:CDTA">
            <gml:metaDataProperty>
              <offering>
                <id>meteo_air_temperature</id>
                <name>meteo_air_temperature</name>
              </offering>
            </gml:metaDataProperty>
            <swe:uom code="C" />
          </swe:Quantity>
        </sml:output>
        <sml:output name="meteo_humidity">
          <swe:Quantity definition="urn:SDN:P021:51:CHUM">
            <gml:metaDataProperty>
              <offering>
                <id>meteo_humidity</id>
                <name>meteo_humidity</name>
              </offering>
            </gml:metaDataProperty>
            <swe:uom code="%" />
          </swe:Quantity>
        </sml:output>
        <sml:output name="meteo_solar_radiation">
          <swe:Quantity definition="urn:SDN:P021:51:CSLR">
            <gml:metaDataProperty>
              <offering>
                <id>meteo_solar_radiation</id>
                <name>meteo_solar_radiation</name>
              </offering>
            </gml:metaDataProperty>
            <swe:uom code="wm2" />
          </swe:Quantity>
        </sml:output>
        <sml:output name="meteo_atmospheric_pressure">
          <swe:Quantity definition="urn:SDN:P021:51:CAPH">
            <gml:metaDataProperty>
              <offering>
                <id>meteo_atmospheric_pressure</id>
                <name>meteo_atmospheric_pressure</name>
              </offering>
```

```
            </gml:metaDataProperty>
            <swe:uom code="hPa" />
           </swe:Quantity>
          </sml:output>
          <sml:output name="meteo_water_temperature">
           <swe:Quantity definition="urn:SDN:P021:51:TEMP">
            <gml:metaDataProperty>
             <offering>
              <id>meteo_water_temperature</id>
              <name>meteo_water_temperature</name>
             </offering>
            </gml:metaDataProperty>
            <swe:uom code="C" />
           </swe:Quantity>
          </sml:output>
         </sml:OutputList>
        </sml:outputs>
        <!--
         description of components of this sensor system; tsdge are
         currently not used by the 52N SOS
        -->
       </sml:System>
    </sml:member>
  </sml:SensorML>
</sos:SensorDescription>
<!--
  ObservationTemplate parameter; this has to be an empty measurement
  at the moment, as the 52N SOS only supports Measurements to be
  inserted
-->
<sos:ObservationTemplate>
  <om:Measurement>
    <om:samplingTime />
    <om:procedure />
    <om:observedProperty />
    <om:featureOfInterest>weather_observation_point</om:featureOfInterest>
    <om:result uom=""></om:result>
  </om:Measurement>
</sos:ObservationTemplate>
          <domainFeature>
           <GenericDomainFeature gml:id="SDG_GEOMARGEN">
```

```
            <gml:description>Geomargen 20111 on Sarmiento de Gamboa</gml:description>
            <gml:name>SDG_GEOMARGEN</gml:name>
            <gml:location>
             <gml:Polygon srsName="urn:ogc:def:crs:EPSG::4326" xsi:type="gml:PolygonType">
                        <gml:exterior>
              <gml:LinearRing xsi:type="gml:LinearRingType">
                 <gml:coordinates>51.7167 8.76667, 52.7167 8.76667, 52.7167 9.76667, 51.7167 9.76667, 51.7167
8.76667</gml:coordinates>
              </gml:LinearRing>
             </gml:exterior>
            </gml:Polygon>
           </gml:location>
          </GenericDomainFeature>
         </domainFeature>
  </sos:RegisterSensor>
 </env:Body>
</env:Envelope>
```

III.3.2-. InsertObservation().

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><InsertObservation xmlns="http://www.opengis.net/sos/1.0"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:om="http://www.opengis.net/om/1.0" xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sa="http://www.opengis.net/sampling/1.0" xmlns:sos="http://www.opengis.net/sos/1.0"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" service="SOS" version="1.0.0"
xsi:schemaLocation="http://www.opengis.net/sos/1.0 http://schemas.opengis.net/sos/1.0.0/sosInsert.xsd
http://www.opengis.net/sampling/1.0 http://schemas.opengis.net/sampling/1.0.0/sampling.xsd
http://www.opengis.net/om/1.0 http://schemas.opengis.net/om/1.0.0/extensions/observationSpecialization_override.xsd">
 <AssignedSensorId>urn:ogc:object:feature:Sensor:SARMIENTO-METEO2</AssignedSensorId>
 <om:Observation>
  <om:samplingTime>
   <gml:TimePeriod xsi:type="gml:TimePeriodType">
    <gml:beginPosition>2011-11-08T22:14:55+00:00</gml:beginPosition>
    <gml:endPosition>2011-11-08T22:14:55+00:00</gml:endPosition>
   </gml:TimePeriod>
  </om:samplingTime>
  <om:procedure xlink:href="urn:ogc:object:feature:Sensor:SARMIENTO-METEO"/>
 <om:observedProperty>
   <swe:CompositePhenomenon dimension="1" gml:id="met">
    <gml:name>resultComponents</gml:name>
    <swe:component xlink:href="urn:ogc:data:time:iso8601"/>
    <swe:component xlink:href="urn:SDN:P021:51:EWSB"/>
    <swe:component xlink:href="urn:SDN:P021:51:CDTA"/>
```

```
  <swe:component xlink:href="urn:SDN:P021:51:CHUM"/>
  <swe:component xlink:href="urn:SDN:P021:51:CSLR"/>
  <swe:component xlink:href="urn:SDN:P021:51:CAPH"/>
  <swe:component xlink:href="urn:SDN:P021:51:TEMP"/>
 </swe:CompositePhenomenon>
</om:observedProperty>
<om:featureOfInterest>
 <gml:FeatureCollection>
  <gml:featureMember>
   <sa:SamplingPoint gml:id="weather_observation_point">
    <gml:name>weather_observation_point</gml:name>
    <sa:sampledFeature xlink:href=""/>
    <sa:position>
     <gml:Point>
      <gml:pos srsName="urn:ogc:def:crs:EPSG::4326">35.2160863 -7.4329352</gml:pos>
     </gml:Point>
    </sa:position>
   </sa:SamplingPoint>
  </gml:featureMember>
 </gml:FeatureCollection>
</om:featureOfInterest>
<om:result>
 <swe:DataArray>
  <swe:elementCount>
   <swe:Count>
    <swe:value>8</swe:value>
   </swe:Count>
  </swe:elementCount>
  <swe:elementType name="Components">
   <swe:DataRecord>
    <swe:field name="Time">
     <swe:Time definition="urn:ogc:data:time:iso8601"/>
    </swe:field>
    <swe:field name="feature">
     <swe:Text definition="urn:ogc:data:feature"/>
    </swe:field>
    <swe:field name="meteo_direcion_viento">
     <swe:Quantity definition="urn:SDN:P021:51:EWSB">
      <swe:uom code=""/>
     </swe:Quantity>
    </swe:field>
```

```
    <swe:field name="meteo_temperatura_aire">
     <swe:Quantity definition="urn:SDN:P021:51:CDTA">
      <swe:uom code="celsius"/>
     </swe:Quantity>
    </swe:field>
    <swe:field name="meteo_humedad">
     <swe:Quantity definition="urn:SDN:P021:51:CHUM">
      <swe:uom code="%"/>
     </swe:Quantity>
    </swe:field>
    <swe:field name="meteo_radiacion_solar">
     <swe:Quantity definition="urn:SDN:P021:51:CSLR">
      <swe:uom code="WM2"/>
     </swe:Quantity>
    </swe:field>
    <swe:field name="meteo_presion_atm">
     <swe:Quantity definition="urn:SDN:P021:51:CAPH">
      <swe:uom code="hPa"/>
     </swe:Quantity>
    </swe:field>
    <swe:field name="meteo_temperatura_Agua">
     <swe:Quantity definition="urn:SDN:P021:51:TEMP">
      <swe:uom code="C"/>
     </swe:Quantity>
    </swe:field>
   </swe:DataRecord>
  </swe:elementType>
  <swe:encoding>
   <swe:TextBlock blockSeparator=";" decimalSeparator="." tokenSeparator=","/>
  </swe:encoding>
  <swe:values>2011-11-08T22:13:55+00:00,weather_observation_point,181.968,-40,0,0.089,1017.485,0</swe:values>
 </swe:DataArray>
</om:result>
</om:Observation>
</InsertObservation>
```

III.3.3-. GetObservation().

```
<?xml version="1.0" encoding="UTF-8"?>
<GetObservation xmlns="http://www.opengis.net/sos/1.0"
 xmlns:ows="http://www.opengis.net/ows/1.1"
 xmlns:gml="http://www.opengis.net/gml"
 xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:om="http://www.opengis.net/om/1.0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/sos/1.0
 http://schemas.opengis.net/sos/1.0.0/sosGetObservation.xsd"
 service="SOS" version="1.0.0" srsName="urn:ogc:def:crs:EPSG::4326">
 <offering>GAUGE_HEIGHT</offering>
 <eventTime>
                                                                    <ogc:TM_During>
                                                <ogc:PropertyName>om:samplingTime</ogc:PropertyName>
                                                                    <gml:TimePeriod>
                                            <gml:beginPosition>2008-03-01T17:44:15+00:00</gml:beginPosition>
                                              <gml:endPosition>2008-05-01T17:44:15+00:00</gml:endPosition>
                                                                    </gml:TimePeriod>
                                                                    </ogc:TM_During>
 </eventTime>
 <procedure><?xml version="1.0" encoding="UTF-8"?>
<GetObservation xmlns="http://www.opengis.net/sos/1.0"
 xmlns:ows="http://www.opengis.net/ows/1.1"
 xmlns:gml="http://www.opengis.net/gml"
 xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:om="http://www.opengis.net/om/1.0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/sos/1.0
 http://schemas.opengis.net/sos/1.0.0/sosGetObservation.xsd"
 service="SOS" version="1.0.0" srsName="urn:ogc:def:crs:EPSG::4326">
 <offering>tss_surface_sea_water_salinity</offering>
 <observedProperty>urn:ogc:object:feature:Sensor:SARMIENTO-TERMOSAL</observedProperty>
 <responseFormat>text/xml;subtype=&amp;quot;om/1.0.0&amp;quot;</responseFormat>
</GetObservation></procedure>
 <observedProperty>urn:SDN:P021:51:PSAL</observedProperty>

 <featureOfInterest>
                                                                    <ogc:BBOX>
                                                <ogc:PropertyName>urn:ogc:data:location</ogc:PropertyName>
                                        <gml:Envelope srsName="urn:ogc:def:crs:EPSG::4326">
                                                <gml:lowerCorner>50.0 7.0</gml:lowerCorner>
                                                <gml:upperCorner>53.0 10.0</gml:upperCorner>
                                                                    </gml:Envelope>
                                                                    </ogc:BBOX>
 </featureOfInterest>
 <result>
                                                                    <ogc:PropertyIsGreaterThan>
                                                <ogc:PropertyName>urn:SDN:P021:51:PSAL</ogc:PropertyName>
                                                                    <ogc:Literal>5</ogc:Literal>
                                                                    </ogc:PropertyIsGreaterThan>
 </result>

 <responseFormat>text/xml;subtype=&amp;quot;om/1.0.0&amp;quot;</responseFormat>
 <resultModel>om:Measurement</resultModel>
 <responseMode>inline</responseMode>
</GetObservation>
```

III.3.4-. UpdateSensor().

```
<?xml version="1.0" encoding="UTF-8"?>
<UpdateSensor
service="SOS"
version="1.0.0"
mobileEnabled="true"
xmlns="http://www.opengis.net/sos/1.0"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:sos="http://www.opengis.net/sos/1.0"
xmlns:swe="http://www.opengis.net/swe/1.0.1"
xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://mars.uni-muenster.de/SOSmobile/trunk/sos/1.0.0/sosUpdateSensor.xsd">
 <SensorID>urn:ogc:object:feature:Sensor:SARMIENTO</SensorID>
 <timeStamp>
  <gml:timePosition>2011-11-09T17:45:15+00</gml:timePosition>
 </timeStamp>
 <position referenceFrame="urn:ogc:def:crs:EPSG::4326">
  <swe:location>
   <swe:Vector>
    <swe:coordinate name="easting">
     <swe:Quantity>
      <swe:value>-7.1700705</swe:value>
     </swe:Quantity>
    </swe:coordinate>
    <swe:coordinate name="northing">
     <swe:Quantity>
      <swe:value>34.8202</swe:value>
     </swe:Quantity>
    </swe:coordinate>
   </swe:Vector>
  </swe:location>
 </position>
 <isMobile>true</isMobile>
 <isActive>true</isActive>
</UpdateSensor>
```

III.4-. Disseny de l'aplicació sosFeeder

Per poder valorar quina és l'arquitectura més adient pel sistema SWE dins del marc de treball exposat d'observacions marines amb enllaç satèl·lit, s'ha dissenyat el següent producte:

sosFeeder

Amb la finalitat de realitzar la inserció de dades al sistema SOS, les insercions es realitzen directament a través de peticions *Simple Object Access Protocol* (SOAP), aprofitant el perfil transaccional de la implementació de 52North de l'estandard OGC de servei SOS. Al diagrama de seqüència de la figura 13 es pot observar el funcionament del producte sosFeeder.

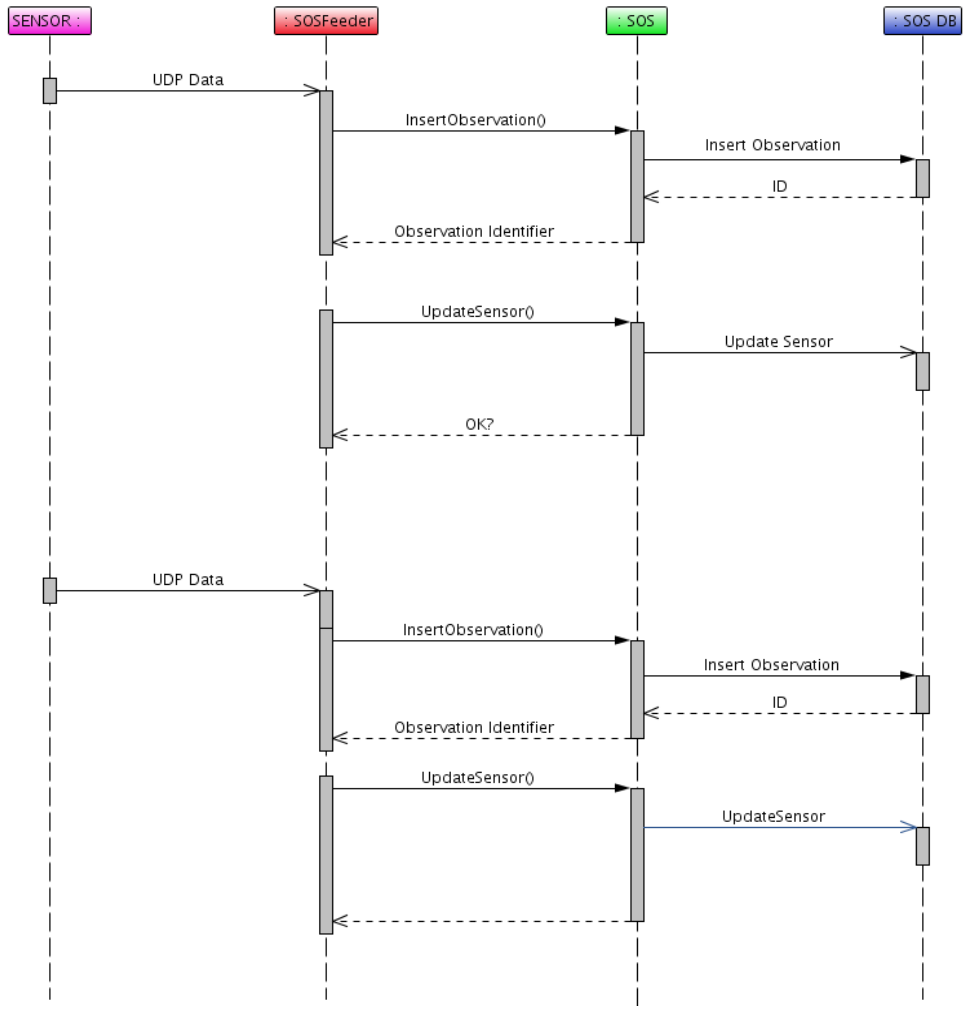

Figura 13: sosFeeder

Per altra banda el disseny d'aquest producte, sosFeeder, ha permés conèixer en profunditat les possibilitats i les limitacions del perfil transaccional de SOS exposades a l'apartat de conclusions (punt IV).

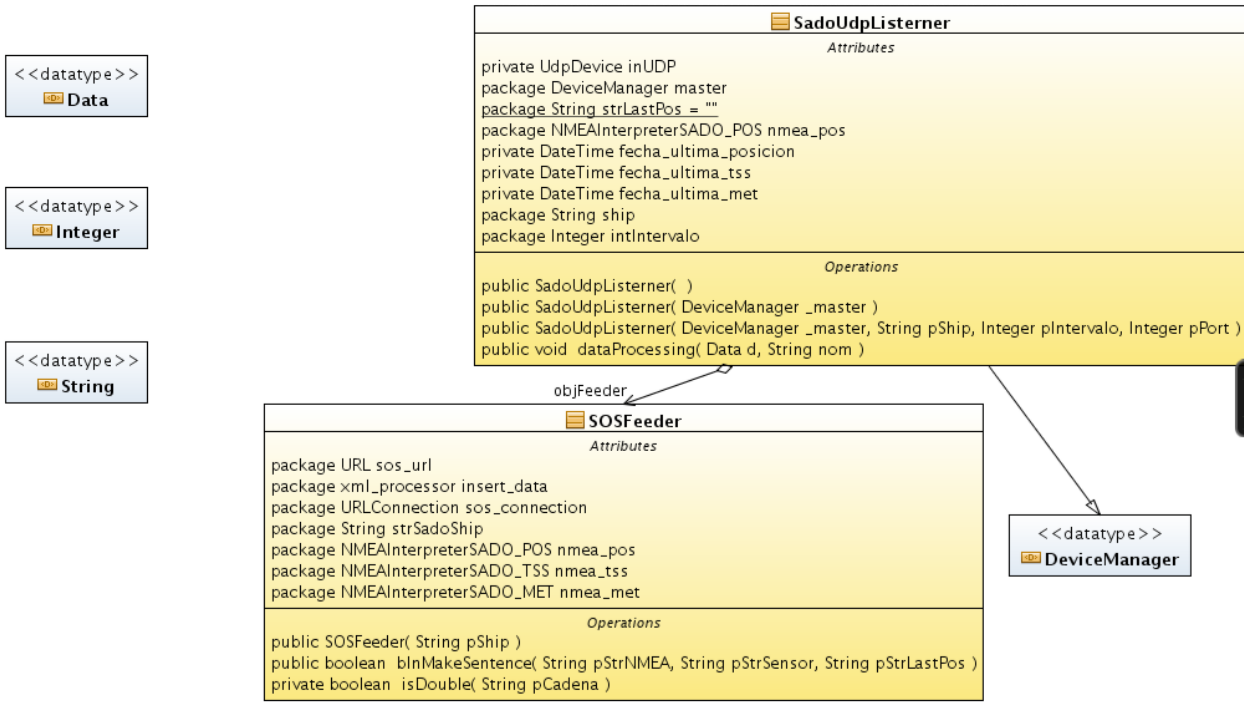

Figura 14: sosFeeder Diagram

Al diagrama de dependències de la figura 14 es pot veure a grans trets el funcionament del sosFeeder:

1. Mitjançant un desenvolupament ja existent: SadoUDPListener, recull les observacions que l'hi arriben a través de la xarxa en forma de paquets UDP en Broadcast.
2. A continuació, sosFeeder, a través d'una connexió http , URLconnection al digrama, es comunica amb el servei de SOS.

3. Interpreta les dades amb l'interpret corresponent a cada trama NMEA:
    1. Trama de posició: nmea_pos.
    2. Trama de meteorologia: nmea_met.
    3. Trama de termosalinitat:nmea_tss.
    4. Mitjançant un desenvolupament ja existent: SadoUDPListener, recull les observacions que l'hi arriben a través de la xarxa en forma de paquets UDP en Broadcast.
4. Codifica les dades per tal de formar un document XML (xml_procesor) amb la petició d'insert que a través de la connexió URL oberta amb el servei SOS farà que les dades s'afegeixin al servei SOS.

III.5-. Disseny de l'aplicació CLIENTSOSUTM

S'ha dissenyat un client SOS per Android, d'ara en endavant SOSUTMCLIENT, amb l'objectiu d'avaluar d'una manera pràctica els avantatges de treballar amb estàndards oberts, tal i com són els casos de SOS i SensorML que permeten tant operar amb dades com amb les pròpies descripcions dels sensors, a través de peticions estàndards, fent possible la integració de tota aquesta informació en entorns oberts i allunyats de les aplicacions i sistemes de representació propietàries de cada fabricant de sensors amb les que ens veuríem abocats a treballar en un altre cas.

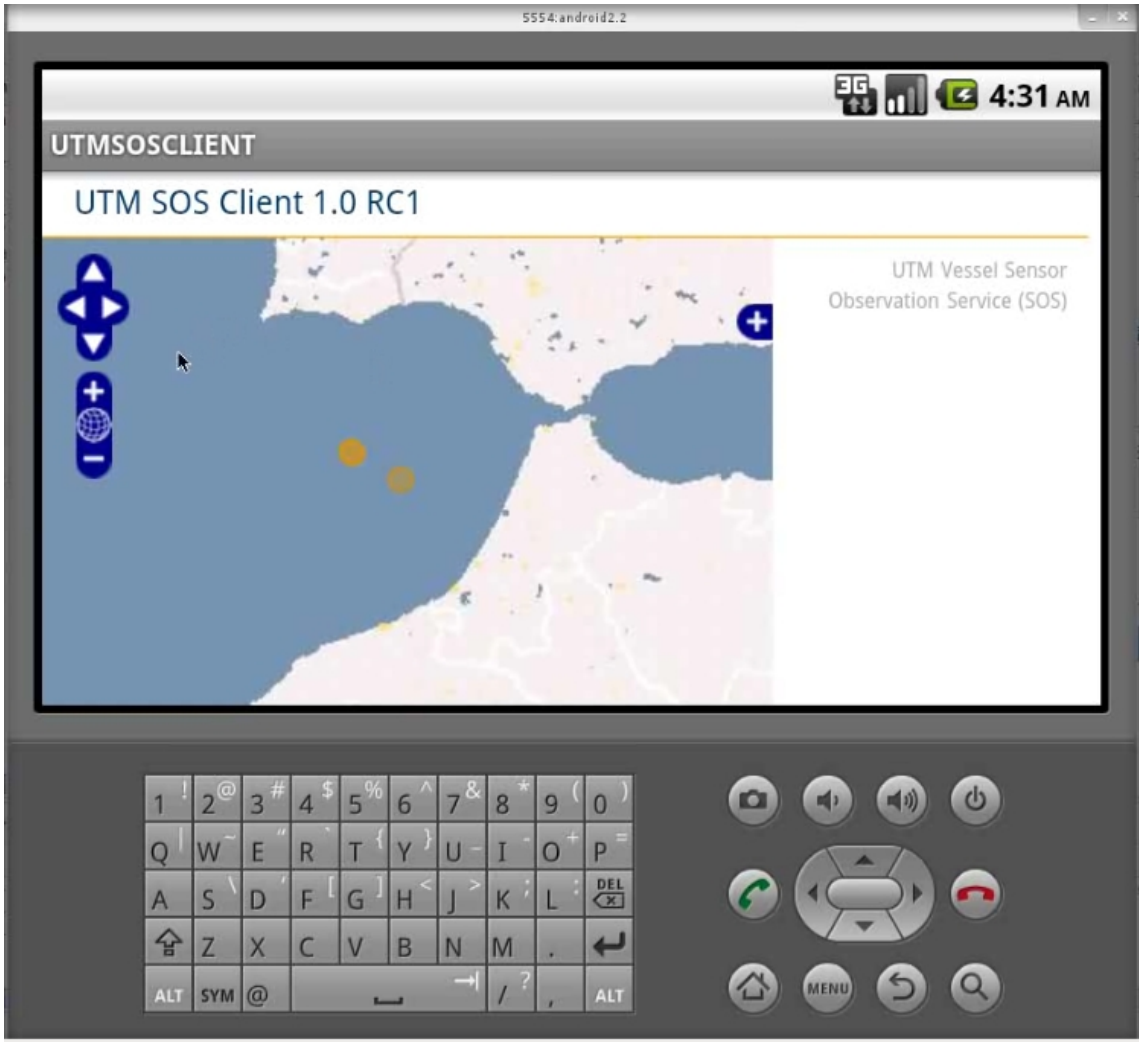

Figura 15: SOSUTMCLIENT

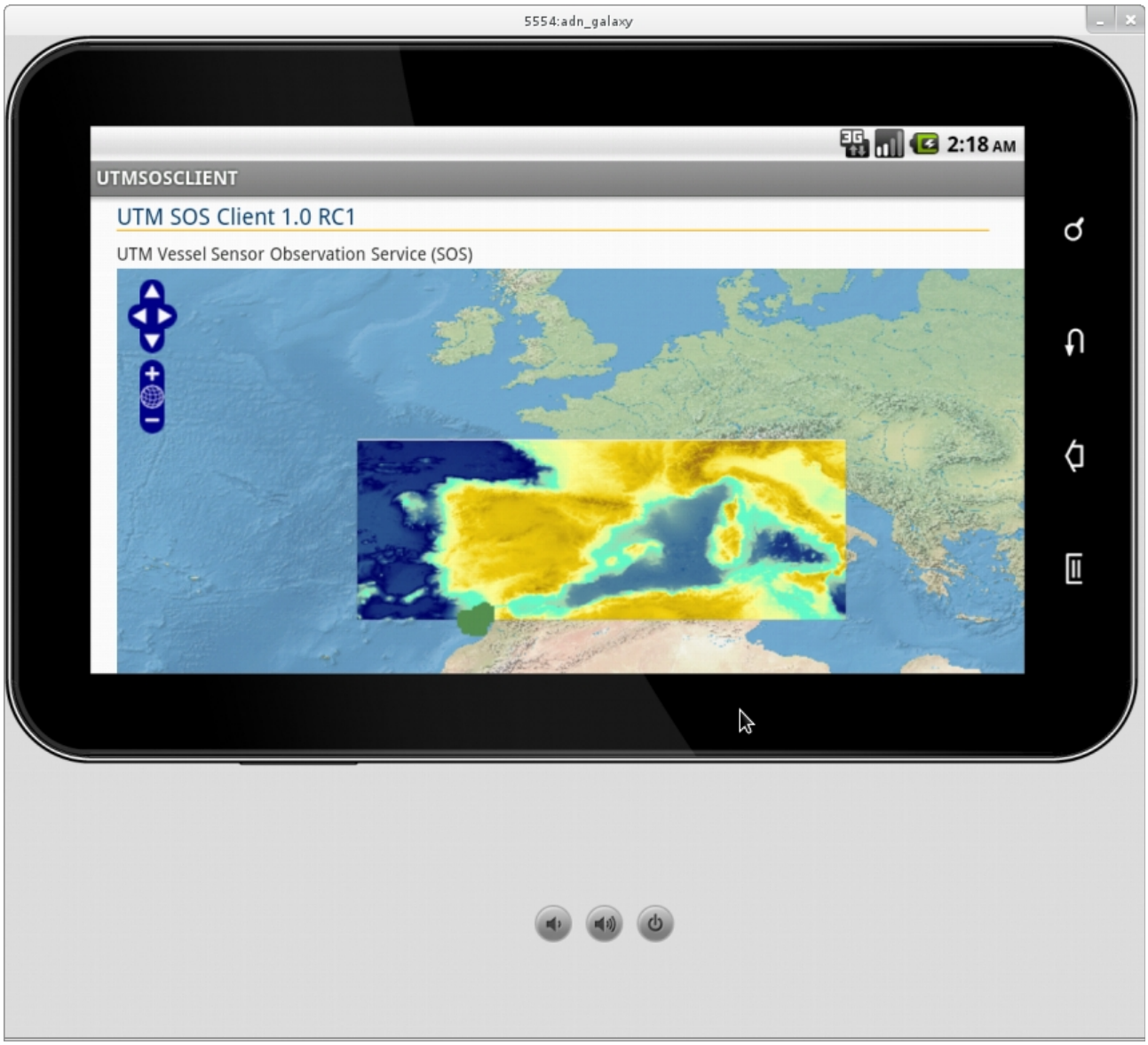

Figura 16: SOSUTMCLIENT Galaxy

A les figures 16 i 17 s'observa l'aplicació SOSUTMCLIENT executant-se en un simulador de dispositiu mòbil. Sobre una capa base apareixen els dos punts que representen dos vaixells al sud de l'estret de Gibraltar.

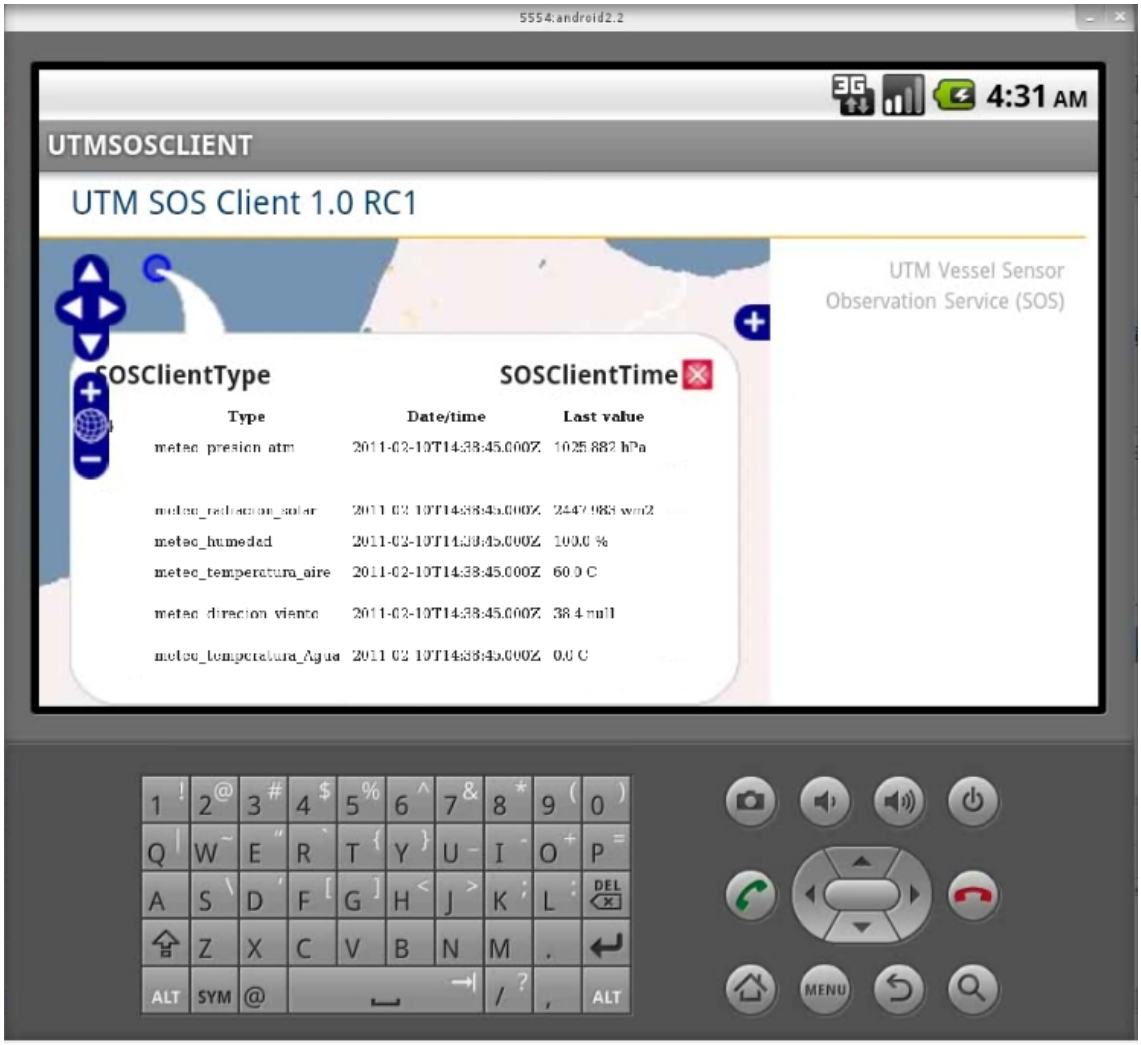

Figura 17: SOSUTMCLIENT data

A la figura 17 es pot veure com al clickar un dels punts apareix la informació més recent del termosalinometre associada a aquest punt, que s'obté a través d'una petició al servei SOS.

IV-. Conclusions

S'han detectat tot un seguit d'avantatges i de mancances de SWE aplicat a observacions de sensors marins.

La conclusió principal del present treball és que és molt beneficiosa la incorporació de tecnologia SWE per l'àmbit d'observacions de sensors marins.

Dins d'una plataforma d'observació oceanogràfica conviuen sensors que poden ser molt diversos: sensors de dades meteorològiques, de dades de l'aigua en superfície, dades de l'aigua en profunditat, dades geològiques del fons marí, dades de velocitat i posició de la pròpia plataforma i moltes d'altres. Aquest conjunt de sensors està molt lluny de mantenir-se constant en el temps i pot ser modificat en funció de les necessitats del que es vulgui observar en cada moment.

Donada la gran heterogeneïtat de sensors, fabricants i interfícies per accedir a les observacions dels sensors, resulta molt interessant la creació d'un servei que permeti l'accés i consulta a les dades obtingudes pels sensors d'una forma unificada.

Amb SWE aquest accés permet poder filtrar el conjunt de dades retornat al client seguint diferents criteris, com pot ser els referits a un marc temporals determinant: quines són les dades més recents, quines dades hi han entre aquestes dues dates; referides a un marc geogràfic: quines dades hi ha dins d'aquesta "caixa de coordenades" ("bounding box"). El disseny d'un client SOS per a Android permet veure com l'ús de SWE fa possible l'accés i la representació de dades oceanogràfiques en dispositius i plataformes molt generalistes com és el cas d'Android, d'una forma senzilla i molt flexible.

De la mateixa manera que resulta molt interessant poder accedir a les dades d'una forma estandarditzada, la conclusió del present treball és que per les mateixes raons anteriorment exposades, la possibilitat que proporciona SWE mitjançant SensorML de

descriure el propi sensor, els processos que hi tenen lloc i informació que resulta clau com pot ser el calibratge de l'instrument, permet proporcionar d'una manera molt operativa i ordenada totes les metadades que descriuen d'una forma el més completa possible com s'ha obtingut les observacions. El disseny de sosFeeder permet veure com l'ús de SWE permet la integració de dades de sensors que no tenen res a veure, ni en la seva funció (dades meteorologies, dades de termosalinitat), ni en el seu fabricant, entre d'altres característiques que en un altre marc de treball obligarien a desenvolupar protocols de treball específics per a cadascun d'ells.

Actualment s'està realitzant un esforç molt important per unificar els diferents models que conviuen dins de SWE: SOS, SAS, SPS.

"A major change in the design of the new generation SWE is a common model for SWE services. Many aspects of SWE service specifications can be commonly defined. This includes service operations and exceptions, among others. To harmonize these aspects the SWE Service Model (SWES) standard [51] has been developed." (New Generation Sensor Web Enablement, 2011 [2]). El concepte cabdal és el de SWE Service Model (SWES), que permet treballar als diferents serveis dins d'un marc comú, tal i com és el cas de SOS 2.0 i de SPS 2.0

També cal incidir en un altre dels beneficis d'integrar les dades provinents dels sensors en un sistema SWE-SOS. Els serveis de catàleg (figura 18) típicament associats com a porta d'entrada als serveis web d'informació geogràfica permeten que el client faci una cerca, tant de les dades com dels serveis disponibles, ja que els serveis de catàleg serveixen dos tipus de dades, o més concretament metadades:

- Metadades que descriuen un set de dades.
- Metadades que descriuen un servei de dades, en el món d'informació geogràfica seria un servei de mapes (WMS) o un servei de cobertures (WCS) per exemple. La descripció del servei la recull el CWS a través de peticions estàndards als diferents serveis WMS, WCS o WFS (*harvesting*). A l'integrar les observacions dels sensors en un servei SWE-SOS es possibilita que les metadades del servei SOS i dels propis sets de dades estiguin disponibles a

través d'un servei de catàleg, millorant en gran mesura la interoperabilitat.

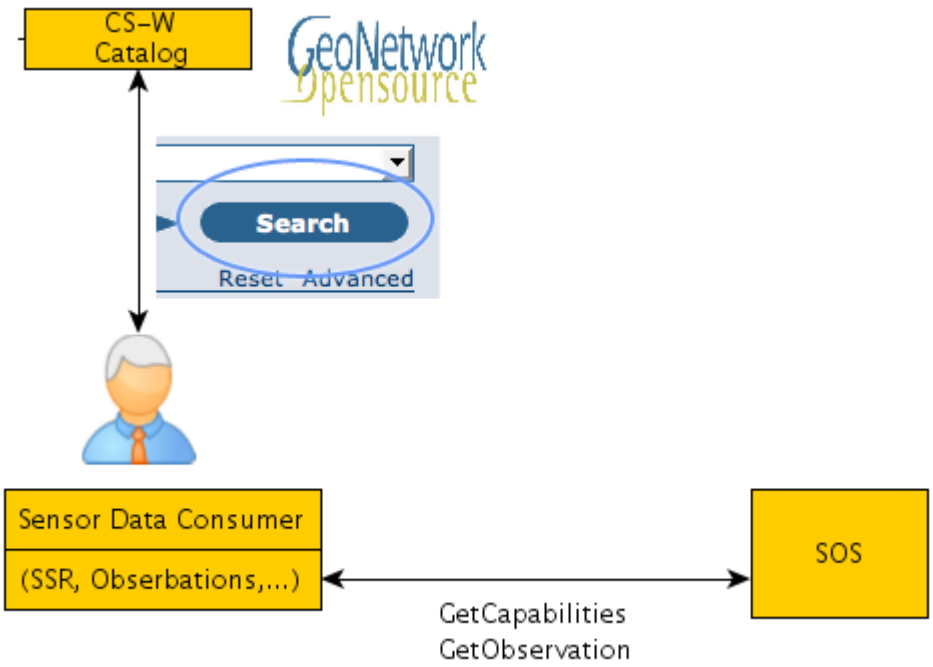

Figura 18: Client-SOS-CWS

La necessitat de millorar algunes interfícies de l'especificació 1.0 de SWE ja ha quedat palesa anteriorment en aquest mateix treball, especialment en el cas de sensors mòbils:

"*To be able to describe mobile sensor data the current data model defined by the OGC needs to be adjusted. Thereby, the essential extension of the model is to endow an observation with information about the location where it took place as well as with the geometric description of superior features to which it belongs (e.g. the area within a mobile sensor is moving)" (Providing Mobile Sensor Data in a Standardized Way - The SOSmobile Web Service Interface, 2008* [8])

A continuació es detallen les que s'han considerat les mancances més remarcables i possibles solucions que es podrien adoptar per superar les limitacions de l'estàndard actual.

A l'hora de determinar quina és la millor forma per aconseguir donar resposta a les limitacions detectades en SWE per dades marines de sensors remots, s'han avaluat tres aproximacions:

1. Esperar que les millores es produeixin dins de la especificació de l'estàndard:. esperar que a la nova especificació de SWE 2.0 incorpori les millores desitjades, veure esborrany SWE 2.0 [31]. Tot i que com a membres OGC i partint de la base de que són d'estàndards oberts, es pot argumentar i fer peticions per tal d'incorporar a l'especificació peticions que responguin a les necessitats detectades. Si aquestes necessitats són prou generals podrien passar a formar part de l'estàndard.
2. Modificació del codi font del SOS: per aconseguir una integració el més eficient a cada problemàtica concreta. Al tractar-se de programari lliure, tal i com és natural. Es pot descarregar tot el codi font del projecte SOS que es consideri convenient. Seria el cas dels fonts de la implementació de referència de 52North que s'ha emprat al present treball i que és accessible des del gestor de "subversión" (SVN). En general no es considera que adaptar els fonts d'un projecte de forma unilateral ("Fork") sigui la pràctica més aconsellable, la raó són les dificultats a l'hora d'actualitzar a noves versions del programari que s'ha fet servir com a base del "Fork".
3. "Middelware Web Service": es tracta d'un servei web que faci de capa intermitja entre el propi servei SOS i la resta de serveis d'inserció i consulta de dades. En l'arquitectura orientada a serveis amb la que es treballa es considera que el disseny i desenvolupament d'un servei web és l'aproximació més eficient i que permet separar per una banda les funcionalitats més específiques dins del servei web propi i que els clients estàndard ataquin directament el servei web.

Cadascuna d'aquestes té avantatges i inconvenients. S'ha arribat a la conclusió de que la tercera opció, "Middelware Web Service", és la que proporciona més flexibilitat i la que en resum és la més eficient, tot i que afegeix un factor de complexitat en l'arquitectura del sistema a l'incorporar un element més a tot el sistema. En tot cas, si es pot evitar i no hi han raons de pes que ho justifiquin sempre seria més convenient treballar atacant directament el servei SOS. L'aproximació del servei intermedi pretén donar resposta únicament a problemàtiques molt concretes d'una forma eficient i sense comprometre l'estàndard.

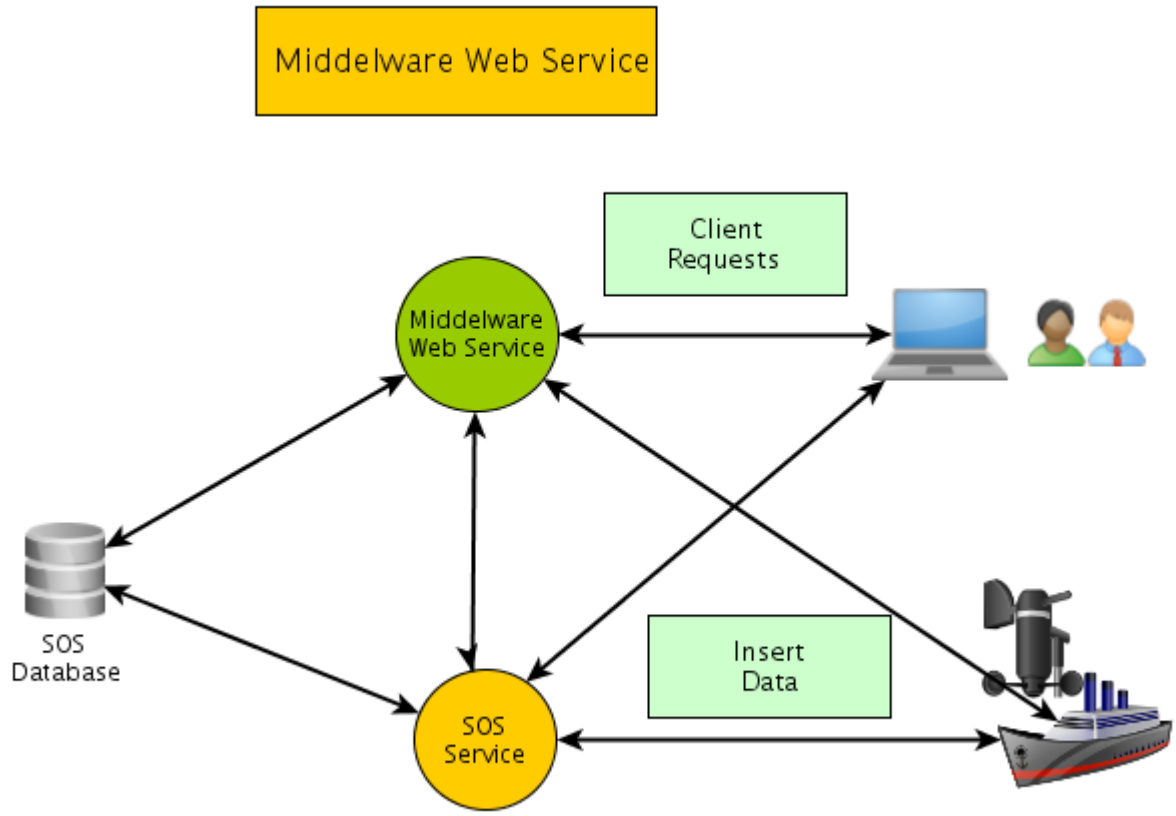

Figura 19: Middelware Web Service

A la figura anterior (figura 19), a la part superior del diagrama es pot veure com les aplicacions de consulta de dades poden atacar directament en servei SOS (clients estàndards) o bé, atacar el servei intermedi a través de peticions no estàndards, sent en aquest cas el servei intermedi l'encarregat de processar aquestes peticions, ja sigui a través de varis peticions al servei SOS o atacant directament la base de dades de SOS. Per altra banda els serveis d'inserció de dades. En tots els casos que es fa servir el servei SOS intermedi aquest retornaria a l'aplicació amb la que interactuï o bé les dades requerides o una resposta indicant si la operació s'ha realitzat correctament l'operació o no. Aplicada a la problemàtica del present projecte amb observacions marines en plataformes remotes amb enllaços satèl·lit, l'ús del servei intermedi es pot utilitzar per minimitzar al màxim el tràfic de metadades en les peticions d'inserció d'observacions. Un altre exemple seria el cas d'insercions massives de dades en mode "batch", en el que un set de dades és recuperat pel servei intermedi a través d'un enllaç no permanent (connexió ftp puntual, scp) per procedir posteriorment a incorporar el conjunt de dades a servei SOS pròpiament dit. En aquest últim cas el servei intermedi estaria situat a terra.

Per altra banda és especialment rellevant la necessitat d'incloure peticions de tipus: ***UpdateSensor*** a l'estàndard OGC de SOS com les que ja implementa 52North SOS a la seva implementació (Figura 3), però que en la versió actual de SOS (SOS 1.01), no formen part de l'estàndard OGC.

Les peticions UpdateSensor resulten clau per tal d'aconseguir un millor encaix de sensors mòbils, com és el cas dels sensors objectes del projecte, ja que a través de peticions *UpdateSensor* es pot actualitzar l'estat del sensor (actiu o no actiu) i el que és més important, la seva posició (coordenades i alçada o profunditat del sensor).

En aquest punt resulta cabdal separar el que és la posició del sistema de sensors en un moment donat (Figura 20 anotació A) i la posició de l'instrument dins del propi sistema (Figura 17 anotació B i C).

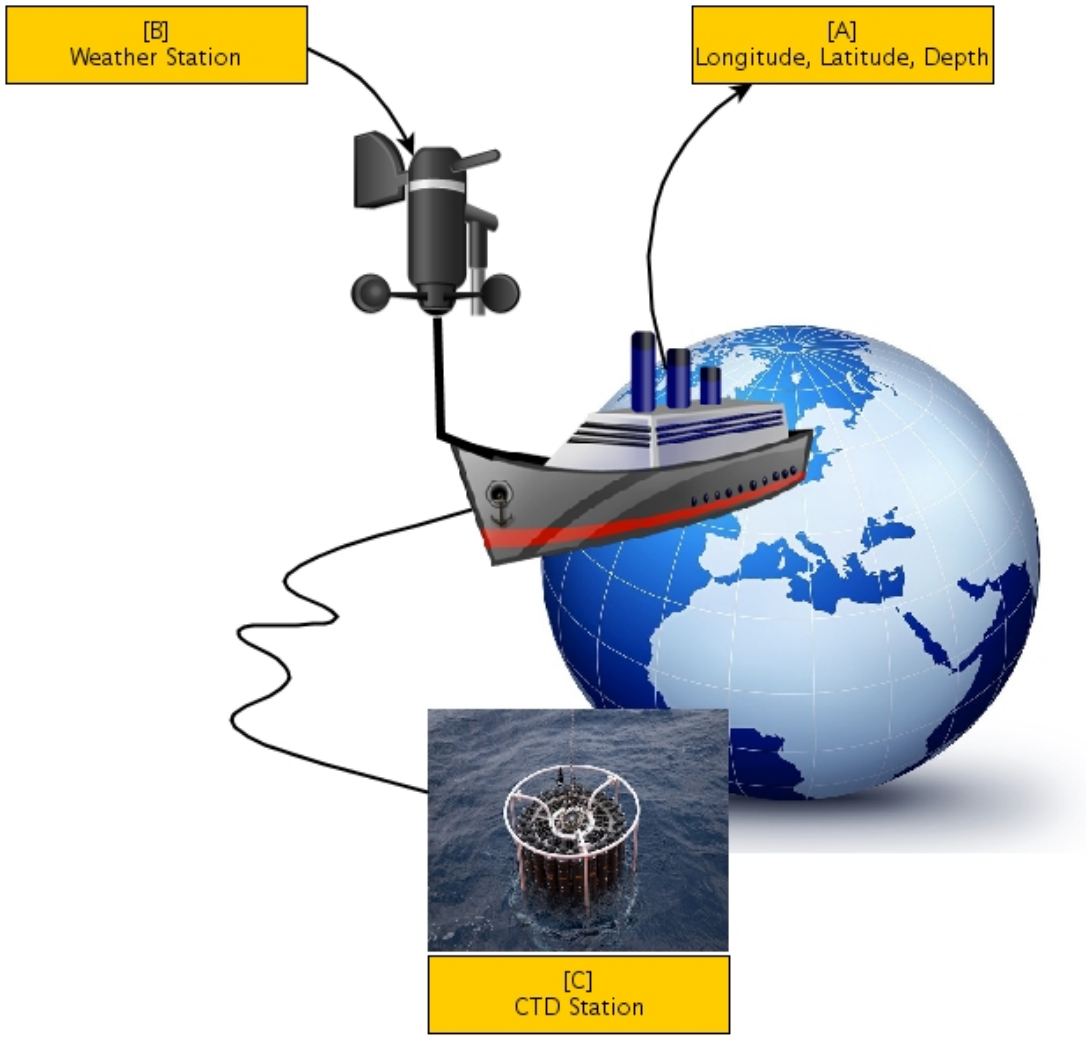

Figura 20: Sensor Positions

És remarcable la gran sobrecàrrega de metadades que pateix el sistema SOS en les peticions de recuperació de dades i a les peticions d'inserció, ja que en cadascuna d'aquestes peticions SOAP, la petició XML porta metadades dels camps de dades que hi intervenen, el que fa que aquestes peticions siguin innecessàriament grans en la majoria dels casos. En l'escenari marí objecte del present treball, aquestes comunicacions es realitzen en moltes ocasions a través d'un enllaç satèl·lit i això té unes fortes implicacions a nivell de costos econòmics i de velocitats de connexió econòmiques importants lligades al volum de dades transmeses en la comunicació.

Es considera que podria resultar interessant, per tal d'evitar aquest entrebanc la incorporació a SOS d'una petició SOAP **InsertResult** (figura 21) de l'estil de la ja existent a l'estàndard **getResult** (figura 22).

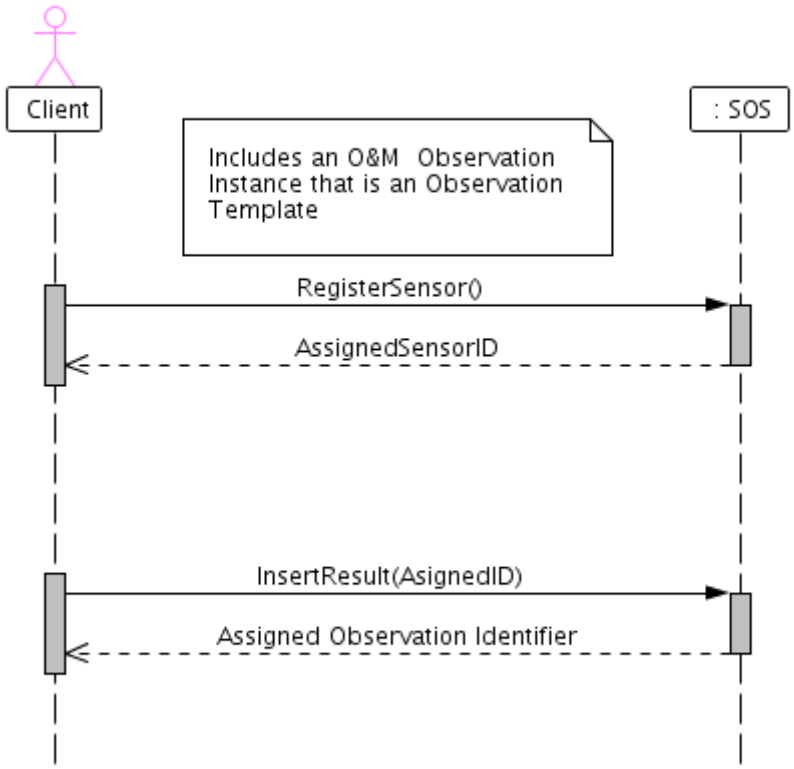

Figura 21: InsertResult

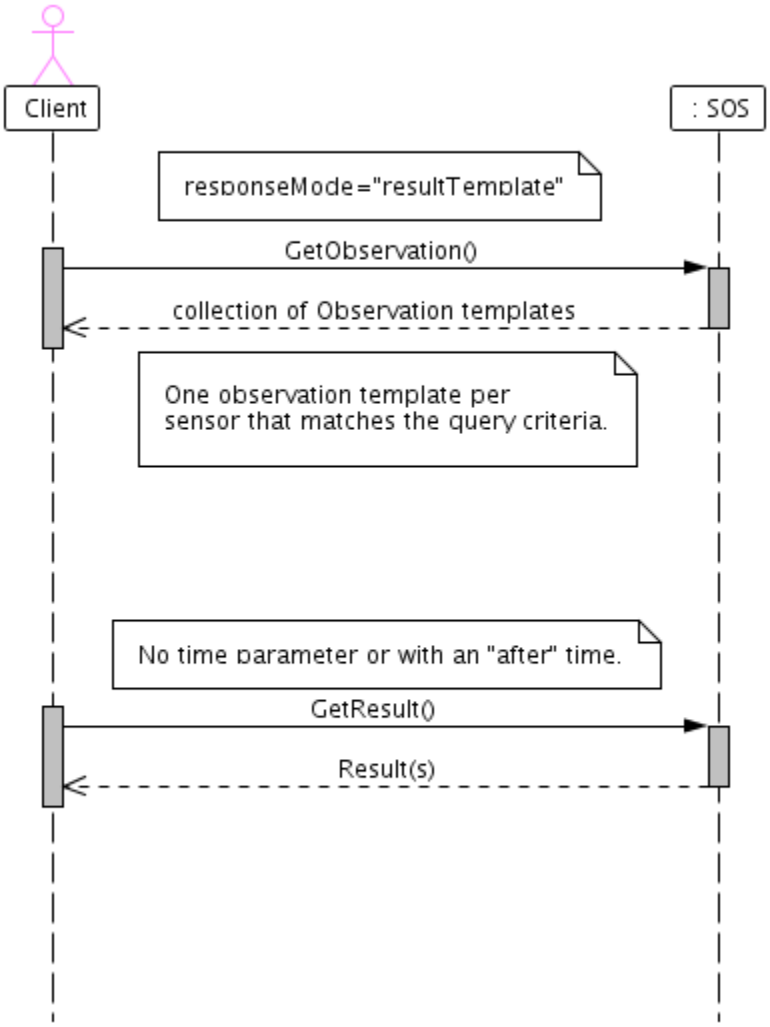

Figura 22: getResult

A la següent figura (figura 23) es pot veure de forma esquemàtica l'evolució dels diferents components que conformen SWE. Així veiem com *transducer markup language (*TML) no seguirà a SWE 2.0. TML suportava la codificació tant de les dades com de les metadades dels sensors, focalitzat en el *streaming* de dades, però a la pràctica el seu ús ha estat molt poc freqüent i segons sembla pel moment no hi ha una demanda per incorporar-lo.

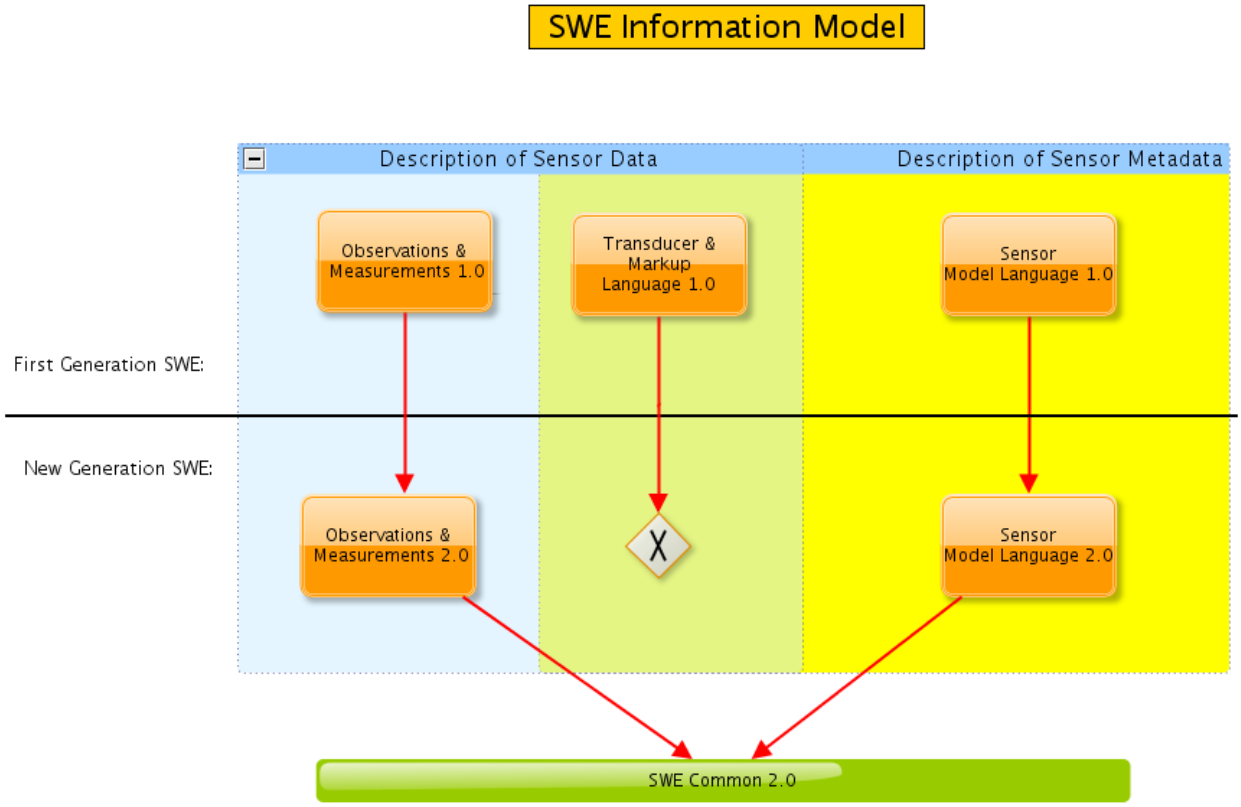

Figura 23: SWE 2.0

V-. Bibliografia

[1] Grigori Babitski, Simon Bergweiler, J¨ rg Hoffmann, Daniel Sch¨ n, Christoph Stasch, and Alexander C. Walkowski. "Ontology-based Integration of Sensor Web Services in Disaster Management". GeoS '09 Proceedings of the 3rd International Conference on GeoSpatial Semantics; 2009.

[2] Arne Brö ring, Johannes Echterhoff, Simon Jirka, Ingo Simonis, Thomas Everding, Christoph Stasch, Steve Liang and Rob Lemmens."New Generation Sensor Web Enablement". Sensors 2011, 11(3), 2652-2699, 2011.

[3] Arne Bröring, Eike Hinderk Jürrens, Simon Jirka, Christoph Stasch. "Development of Sensor Web Applications with Open Source Software". In OpenSoruce GIS 2009, 22 June 2009.

[4] Carlos Granell Canut. "Servicios OGC". FUOC, 2011. 2011.

[5] Hankin, S. & Co-Authors. "Data Management for the Ocean Sciences – the Next Decade". Proceedings of OceanObs'09: Sustained Ocean Observations and Information for Society (Vol. 1), Venice, Italy, 21-25 September 2009, Hall, J., Harrison, D.E. & Stammer, D., Eds., ESA Publication WPP-306, doi:10.5270/OceanObs09.pp.21, 2010.

[6] Cory Henson, Holger Neuhaus, Amit Sheth,Krishnaprasad Thirunarayan, Rajkumar Buyya. "An Ontological Representation of Time Series Observations on the Semantic Sensor Web ". 1st International Workshop on the Semantic Sensor Web (SemSensWeb 2009)June 1st, 2009 Heraklion, Crete, Greece, 2009.

[7] Holger NEUHAUS, Michael COMPTON. "The Semantic Sensor Network Ontology: A Generic Language to Describe Sensor Assets ". AGILE Workshop Challenges in Geospatial Data Harmonisation (2009), 2009.

[8] Christoph Stasch, Arne Bröring, Alexander C. Walkowski. "Providing Mobile Sensor Data in a Standardized Way - The SOSmobile Web Service Interface ". IfGI, Institute for Geoinformatics Robert-Koch-Str. 26-28, Muenster 48149, Germany {staschc, arneb, walkowski}@uni-muenster.de, 2008.

[9] Krzysztof Janowicz, Michael Compton. "The Stimulus-Sensor-Observation Ontology Design Pattern and its Integration into the Semantic Sensor Network Ontology ". The 3rd International workshop on Semantic Sensor Networks 2010

(SSN10) in conjunction with the 9th International Semantic Web Conference (ISWC 2010), 7-11 November 2010, Shanghai,China. 2010, 2010.

[10] Payam Barnaghi, Michael Compton, Oscar Corcho, Raúl García Castro,

John Graybeal, Arthur Herzog, Fraunhofer Gesellschaft, Krzysztof Janowicz, Holger Neuhaus, Andriy Nikolov, Kevin Page, Kerry Taylor. "Semantics and Sensors ". http://www.w3.org/2005/Incubator/ssn/wiki/Incubator_Report, 2011.

[11] Marine Metadata Interoperability. http://marinemetadata.org/mmiswinfo/. 2011.

[12] Open Geospatial Consortium Standards. http://www.opengeospatial.org/.

[13] Open Geospatial Consortium sensor web enablement web group. http://www.opengeospatial.org/projects/groups/sensorweb.

[14] 52 North sensorweb comunity. http://52north.org/communities/sensorweb/.

[15] oostethys sensorweb comunity. http://www.oostethys.org/.

[16] gvSIG i SWE. http://gvsig-desktop.forge.osor.eu/downloads/pub/events/jornadas-Brasileiras/2010/reports/gvSIG_Sensores.pdf.

[17] Map Server SOS. http://mapserver.org/ogc/sos_server.html.

[18] VAST SWE i SensorML inititives. http://www.botts-inc.net/vast.html.

[19] D. Afonso, E. Arilla, O. Garcia, A. Hernández, J. Olivé, J. L. Ruiz, X. Romero, A. Sandoval J. A. Serrano, J. Sorribas, E. Trullols, J. del Río, (2009). "Satellite Communication Systems On-Board Spanish Oceanographic Vessels: Performance and Optimization Analysis," SPACOMM 2009 First International Conference on Advances in Satellite and Space Communications, 2009. pp.180-186.

[20] Nogueras, M.; del Rio, J.; Cadena, J.; Sorribas, J.; Artero C.; Dañ obeitia, J.; Mà nuel, A. OBSEA an oceanographic seafloor observatory. In Proceedings of the IEEE-ISIE, Bari, Italy, 4–7 November 2010; pp. 488-492.

[21] Giovanni Aloisio, Dario Conte, Cosimo Elefante, Italo Epicoco, Gian Paolo Marra, Giangiuseppe Mastrantonio and Gianvito Quarta. "SensorML for Grid Sensor Networks".

[22] Michl, R. Klipp and M. Mothes, "Hydrological Sensor Networks in Germany – Introducing SensorWeb-WSV ". 18th World IMACS / MODSIM Congress, Cairns, Australia 13-17 July 2009 http://mssanz.org.au/modsim09.

[23] Simon Cox . "Geographic Information: Observations and Measurements OGC Abstract Specification" Topic 20 *(OGC 10-004r3 and ISO 19156)*. Open Geospatial Consortium Inc. 2010-11-10.

[24] Scott Fairgrieve. *OWS-7* "CCSI-SWE Best Practices Engineering" *(OGC 10-073r1).* Open Geospatial Consortium Inc. 2010-06-30.

[25] Arthur Na, Mark Priest. "*Sensor Observation Service "(OGC 06-009r6)*, Open Geospatial Consortium Inc. 2007-10-26.

[26] Geonetwork. http://geonetwork-opensource.org/.

[27] Web Enforcement Service (WSS). http://52north.org/communities/security/wss/2.0/.

[28] Arne Brö ring, Johannes Echterhoff, Simon Jirka, Ingo Simonis, Thomas Everding, Christoph Stasch, Steve Liang and Rob Lemmens. "New Generation Sensor Web Enablement" . Sensors 2011, 11, 2652-2699; doi:10.3390/s110302652.

[29] Mike Botts . "Sensor Model Language (SensorML) for In-situ and Remote Sensors". *(OGC 04-019r2)*. Open Geospatial Consortium Inc. 2004-11-01.

[30] Robert Arko, Carlos Rueda, Nan Galbraith, Robert Morris, Luis Bermudez, John Graybeal. "The MMI Device Ontology: Enabling Sensor Integration Use Case: Multibeam Sonars". http://mmisw.org/orr.

[31] Arne Bröring, Christoph Stasch, Johannes Echterhoff. "SOS_2.0_final_draft". (OGC 10-037). Open Geospatial Consortium Inc. 2012-01-06.