Implementación de un servicio y clientes SOS para gestión inteligente de recursos hidráulicos

Proyecto GeoSmartCity

CARDOSO, Juan Luis; HUARTE, Álvaro; LACUNZA, Garazi; PÉREZ, Iván; CABELLO, María

Este resumen describe la experiencia desarrollada en el marco del proyecto **GeoSmartCity** (http://www.geosmartcity.eu), que ha consistido en la realización de un piloto web para la gestión inteligente de la **red de abastecimiento** de la Mancomunidad de la Comarca de Pamplona (http://www.mcp.es).

El proyecto pretende ayudar a los administradores de la red a contrastar y analizar la información que ofrecen los sensores de la red hidráulica registrados en un sistema **SCADA** y los datos simulados de su correspondiente red **EPANET**.

Esa información es procesada y ofrecida a través de un servicio web conforme a los estándares **SOS** (*Sensor Observation Service*), basado en la tecnología *opensource* de **52°North** SOS (https://github.com/52North/SOS). Para ello ha sido necesario el desarrollo de nueva funcionalidad en el servicio de 52°North SOS, implementando un mecanismo de *plugins* para el acoplamiento al vuelo de nuevas fuentes de datos (Sensores y datos observados). Se han implementado también dos nuevos *plugins* para la integración, respectivamente, de los sensores registrados en el SCADA y de los datos hidráulicos calculados en la simulación de su correspondiente red EPANET. Con esta funcionalidad se permite al usuario el contraste y análisis de la información que ofrecen dichos recursos.

A continuación, y con el objetivo de explotar adecuadamente esa información desde un visualizador web de mapas, se ha implementado una nueva librería en JavaScript que cumple con el estándar SOS 2.0 y que ha sido desarrollada con una arquitectura más modular y agnóstica que la actualmente ofrecida por 52°North (pensada para SOS 1.0 y OpenLayers 2), haciendo sencilla su posible extensión para integrarse con cualquier API de desarrollo web. Además en el marco de este proyecto se ha desarrollado la extensión para su utilización con OpenLayers 3.

El paso final ha consistido en el desarrollo de una aplicación web, utilizando los componentes descritos anteriormente, que permite a los gestores de la Mancomunidad controlar y monitorizar el estado de la red de Abastecimiento y comparar los datos en tiempo real con valores de simulación y valores históricos. Para el desarrollo de esta web se ha utilizado el **API SITNA** (http://sitna.navarra.es/geoportal/recursos/api.aspx API JavaScript basado en OpenLayers 3).

Posteriores desarrollos nos van a permitir la presentación «al vuelo» de nuevas simulaciones basadas en la modificación de los elementos de la red EPANET. Dichos procesos se ejecutarán basándose en peticiones y rutinas de tipo WPS y ofrecerán al usuario una potente herramienta para la simulación y análisis de los recursos hidráulicos de que dispone; simulación de roturas, o fugas de agua, cambios en la calibración de válvulas, etc.

PALABRAS CLAVE

GeoSmartCity, Smart Cities, SOS (Sensor Observation Service), 52°North, API SITNA, OpenLayers, EPANET, SCADA, Utilities, Red de Abastecimiento, Agua

INTRODUCCIÓN

El proyecto **GeoSmartCity** [1] (open geo-data for innovative services and user applications towards Smart Cities) tiene como objetivo establecer una multiplataforma capaz de integrar información geográfica de procedencia diversa (local, europea, crowsourcing,…) mediante estándares abiertos y

en el marco de los escenarios de Smart City:

• **Green Energy**: apoyando la toma de decisiones en el ámbito público relacionado con la energía (consumo energético a nivel de edificios, movilidad,…)

• **Underground**: apoyando la gestión integrada de las infraestructuras subterráneas de servicio público a nivel local.

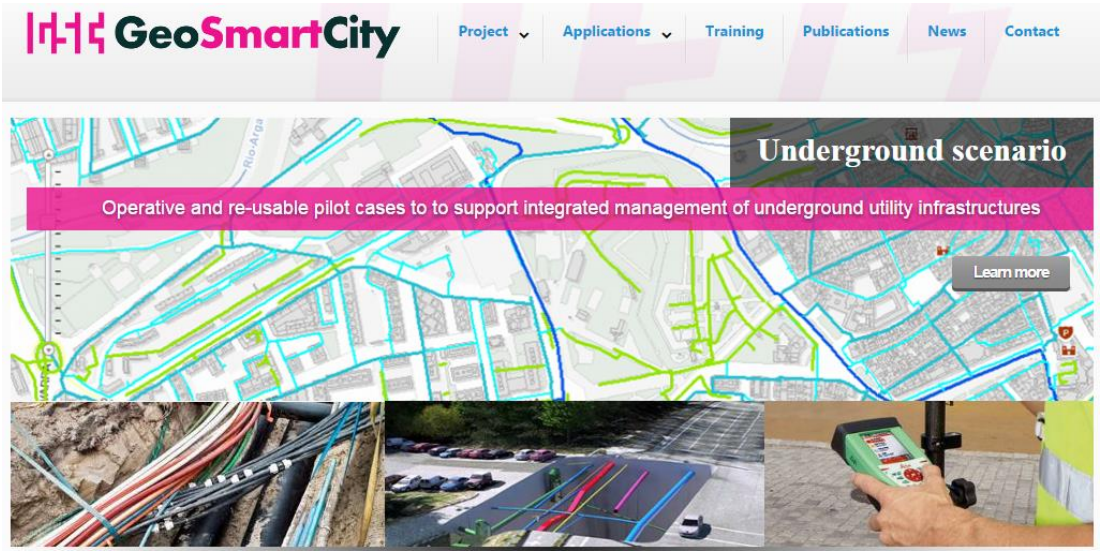

Figura 1: Proyecto GeoSmartCity: www.geosmartcity.eu

GeoSmartCity es un proyecto financiado por el programa 'Competitive and innovation framework – CIP' de la Comisión Europea y pone en contexto y prueba dicha plataforma en el marco de once casos piloto.

El caso piloto "Comarca de Pamplona" implementa el espíritu de dicha plataforma en la entidad local "**Mancomunidad de la Comarca de Pamplona**" [2], en adelante **MCP**, mediante la explotación de diversos recursos hídricos de esta organización.

Mancomunidad
Comarca de Pamplona
Iruñerriko
Mankomunitatea

Figura 2: Logo de la Mancomunidad de la Comarca de Pamplona

La **MCP** (www.mcp.es) es una entidad local compuesta por 50 municipios de Navarra (España), el mayor de los cuales es Pamplona, que es también el centro geográfico de la mancomunidad. La mayor parte de sus integrantes son considerados parte del Área metropolitana de Pamplona.

Este caso piloto implementa un servicio web tipo **SOS (Sensor Observation Service)** [3] para facilitar la gestión inteligente de la red hidráulica de abastecimiento de la **MCP**.

El reto del proyecto es integrar los sensores físicos existentes en la red hidráulica, los datos obtenidos en la simulación de la red con modelos matemáticos, y geoinformación específica de la organización. Y todo ello conforme a los estándares. El usuario con este despliegue podrá analizar de forma ágil el estado actual de los recursos hídricos, la validez de sus simulaciones matemáticas y el comportamiento de la red ante cambios en su configuración operacional (nuevos consumos, nuevos elementos, fallos/roturas,…). Para ello se ha desarrollo una aplicación piloto Web que permite que permite a los gestores de la Mancomunidad controlar y monitorizar el estado de la red de Abastecimiento y comparar los datos en tiempo real con valores de simulación y valores históricos.

ESTÁNDAR SOS – *SENSOR OBSERVATION SERVICE*

La especificación oficial define el propósito de este estándar como:

«The SOS standard is applicable to use cases in which sensor data needs to be managed in an interoperable way. This standard defines a Web service interface which allows querying observations, sensor metadata, as well as representations of observed features. Further, this standard defines means to register new sensors and to remove existing ones. Also, it defines operations to insert new sensor observations. This standard defines this functionality in a binding independent way; two bindings are specified in this document: a KVP binding and a SOAP binding».

El estándar especifica el modelo de datos de objetos "sensor" que proveen valores de una determinada magnitud, y de los metadatos que describen las observaciones realizadas. Los sensores suelen monitorizar en el tiempo los datos observados.

El objetivo de **SOS** es proveer acceso a observaciones realizadas por sensores y sistemas de sensores de una forma estándar, incluyendo sensores remotos, in-situ, fijos y móviles. **SOS** utiliza la especificación "Observation and Measurements" (O&M) para modelar las observaciones de los sensores y "TransducerML" y "SensorML" para modelar los sensores y sistemas de sensores.

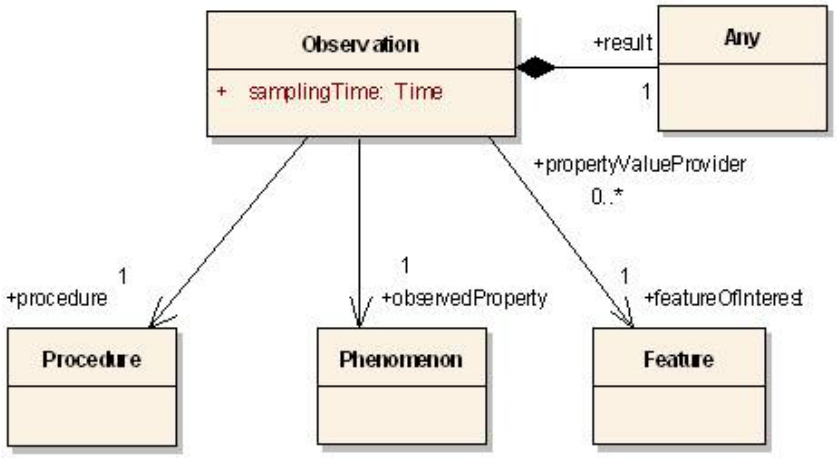

Figura 3: Modelo básico de una observación

Una observación podría ser definida como el acto de observar un fenómeno. Una medida es una observación especializada cuyo resultado es un valor numérico.

El modelo básico de observación contiene cinco elementos. El objeto "procedure" apuntaría a un procedimiento (normalmente un sensor) que produce el valor de la observación. El elemento "observedProperty" hace referencia al fenómeno que fue observado. La entidad "featureOfInterest" se refiere al objeto del mundo real al que pertenece la observación. El atributo "samplingTime" indica el momento en que se efectuó la observación. Finalmente el valor de la observación está contenido en el elemento "result".

La observación actúa como un proveedor de valores de una propiedad: provee un valor (p. ej. 27º Celsius) para una determinada propiedad (p. ej. Temperatura) en un determinado "featureOfInterest" (p. ej. Estación meteorológica) y para un cierto intervalo de tiempo. La localización a la que pertenece la observación está referenciada indirectamente por la geometría de la "featureOfInterest".

Los sensores pueden monitorizar magnitudes físicas como la temperatura, caudal, apertura de una válvula… o incluso eventos como el registro de una incidencia desde un teléfono móvil, el estado operacional de un vehículo dentro de una flota, la cantidad disponible de un producto...

Al consumidor de datos de sensores le interesa obtener las observaciones de uno o más sensores. Esto puede realizarse desde dos puntos de vista. Desde el punto de vista centrado en el sensor,

donde el consumidor ya conoce la existencia de una serie de sensores y desea obtener sus observaciones. O desde el punto de vista centrado en la observación, donde al consumidor le interesa obtener las observaciones de un área geográfica particular sin que, a priori, tenga importancia de qué sensores en particular provienen los datos.

La siguiente figura muestra el consumidor de datos de sensor en un contexto operacional con catálogos OGC CS-W para la búsqueda de instancias SOS. Además se muestran instancias de servicios SOS con ofertas de sensores y observaciones. Los servicios se pueden organizar en topologías complejas usando la agregación u otras técnicas, pero todo esto es transparente para el consumidor. El consumidor solo necesita tratar con registros e interfaces de servicio.

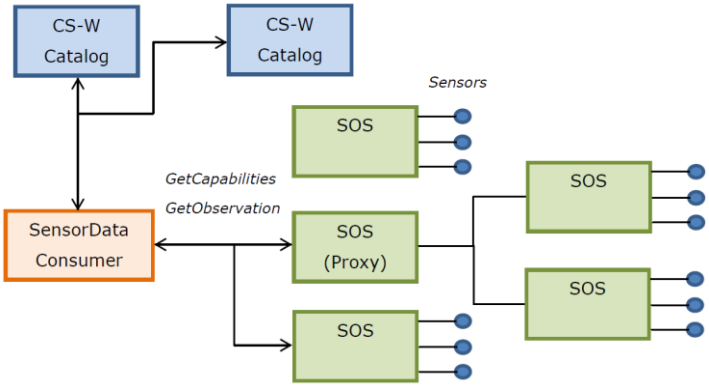

Figura 4: Consumidor de datos de un sensor

En cualquier caso, el consumidor realizaría una búsqueda, normalmente mediante un servicio de catálogo, para descubrir las instancias de los servicios SOS que pueden proveer las observaciones deseadas. Después de esta primera búsqueda el consumidor podría obtener las observaciones directamente desde los servicios o podría realizar nuevas búsquedas a nivel de servicio para obtener los metadatos de los sensores antes de obtener las observaciones de éstos. Las búsquedas a nivel de servicio suponen la invocación de la operación "GetCapabilities" que devuelve información de las ofertas de cada servicio SOS. La obtención de los metadatos de un sensor precisa de la invocación de la operación "DescribeSensor". Finalmente, el consumidor invoca la operación "GetObservation" para obtener los valores de las observaciones deseadas.

INFRAESTRUCTURA HIDRAULICA DEL PROYECTO

La Mancomunidad de la Comarca de Pamplona (**MCP**) gestiona los servicios públicos del agua (abastecimiento y saneamiento), recogida y tratamiento de los residuos urbanos, el transporte urbano comarcal, el servicio de taxi y el Parque Fluvial de la Comarca, atendiendo a una población de aproximadamente 350.000 habitantes.

La **MCP** dispone de un sistema de sensores físicos que monitorizan el estado de sus redes hidráulicas. Estos sensores están registrados en una plataforma **SCADA** [4] donde se almacenan los valores observados y el instante en el que se registraron.

La plataforma **SCADA** gestiona el histórico de los datos enviados por los sensores, añadiendo nuevos valores según su correspondiente frecuencia de actualización de datos. En nuestro caso los sensores integrados envían nuevos valores cada aproximadamente cinco minutos creciendo el volumen de datos de forma lineal con el tiempo. Como se puede entrever, el volumen de registros almacenados es muy elevado y es por ello del uso de estas plataformas **SCADA** para su gestión. Este hecho ha sido determinante en la concepción de la arquitectura software a implementar y que se describirá posteriormente.

La organización tiene instalados en la red de abastecimiento un completo conjunto de sensores pero para nuestro caso piloto sólo se han considerado los de tipo "*Clorímetro*" y "*Caudalímetro*" que

miden respectivamente la concentración de cloro y el caudal de agua de los puntos de la red donde estos elementos se encuentran instalados.

De forma paralela, la **MCP** ha desarrollado varios modelos **EPANET** [5] que describen matemáticamente sus redes hidráulicas. Estos modelos definen todos los elementos físicos de la red (tuberías, válvulas, bombas, depósitos,…) y sus respectivas configuraciones operacionales (consumos, niveles de depósitos, curvas de bombeo, calibraciones de válvulas,…). Para este caso piloto sólo se ha tenido en cuenta un modelo simplificado con alrededor de 15.000 elementos y con una configuración operacional típica de los meses de verano.

El reto del proyecto ha sido integrar los sensores registrados en la plataforma **SCADA**, los datos simulados de la red **EPANET**, y geoinformación específica de la organización. Y todo ello conforme a los estándares. El técnico puede analizar en sus herramientas cliente el estado actual de los recursos hídricos, la validez de sus simulaciones matemáticas y el comportamiento de la red ante cambios en su configuración operacional (nuevos consumos, nuevos elementos, fallos/roturas,…).

ARQUITECTURA *SOFTWARE* DEL PROYECTO

El proyecto utiliza como *software back-end* un servicio web **SOS** basado en tecnología ***opensource*** de **52°North. (**http://52north.org/communities/sensorweb/sos/index.html)

La siguiente figura muestra la arquitectura adoptada:

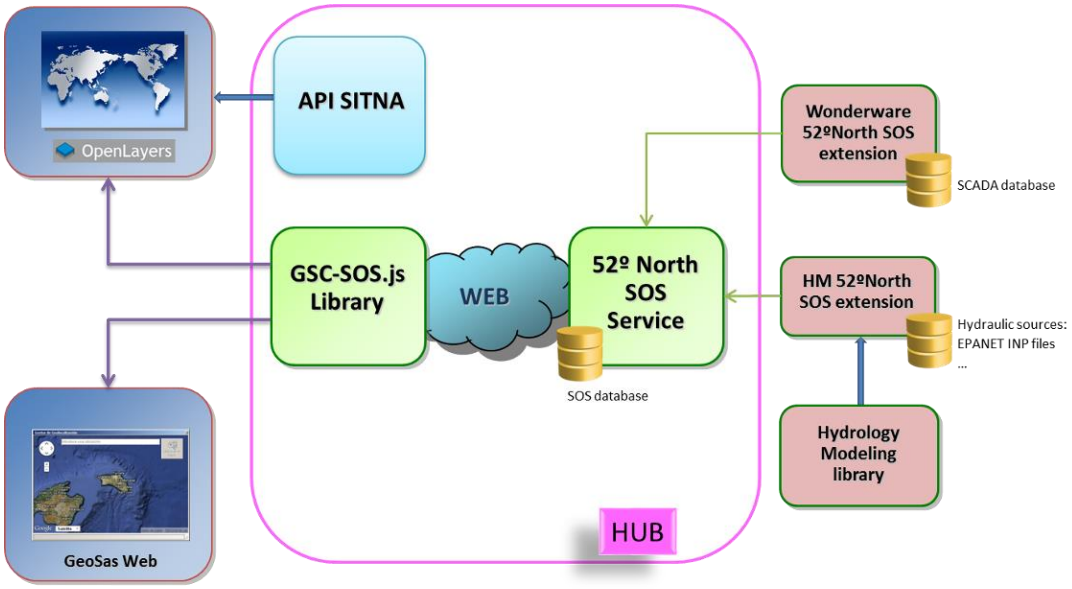

Figura 5: Diagrama de arquitectura del proyecto

52°North [6] es una organización que aglutina una red de *partners* desde ámbitos relacionados con la investigación, industria y administraciones públicas y cuyo principal propósito es el fomento de la innovación en el campo de la geo-información.

La implementación del servicio **SOS** ha sido desarrollada bajo el marco de esta organización y su código fuente se provee gratuitamente [7] bajo licencia **GNU** [8].

En internet hay disponible un servicio **SOS** demo para familiarizarse con la tecnología:

http://sensorweb.demo.52north.org/sensorwebtestbed/

Invocando su "*GetCapabilities*"…

http://sensorweb.demo.52north.org/sensorwebtestbed/sos?service=SOS&request=GetCapabilities

… se pueden observar los metadadatos ("procedure", "featureOfInterest", "offering", "observedProperty",…) de los objetos sensor y observación que el servicio provee. También se indican las capacidades de filtrado de datos en base a criterios espaciales, temporales, por identificador...

INTEGRACIÓN DE DATOS SCADA Y EPANET EN 52ºNORTH SOS

El núcleo de la solución es un servicio **SOS** que implementa un mecanismo de "*plugins*" para integrar los sensores de la plataforma **SCADA**, y los resultados de simulación de modelos **EPANET**.

El servicio **SOS** de **52ºNorth** almacena toda la información de los sensores y observaciones en una base de datos propia y con un esquema de datos específico. En nuestro caso piloto la base de datos de configuración es una PostgreSQL/PostGIS pero están soportados otros proveedores (SQL Server y Oracle). El proceso de registrar los sensores, las observaciones, y las medidas de esas observaciones puede ser invocado con el interfaz **REST** que el servicio ofrece.

Como se ha comentado anteriormente, el volumen de datos almacenado en una plataforma **SCADA** puede ser, y lo es, muy elevado. El sistema almacena cientos de miles de registros; cada sensor emite un par "valor-instante" con una frecuencia determinada (p. ej. Cinco minutos) que se graba a la base de datos interna (SQL Server en nuestro caso). El volumen de información por tanto va creciendo con el tiempo y tarde o temprano nos tendremos que pelear con ese buen "*big-set*" de datos.

Por otra parte, el servicio cachea en **RAM** mucha de la información registrada en la base de datos para de esta forma optimizar la respuesta a las peticiones web solicitadas. El servicio implementa internamente unos procesos de sincronización automática entre la base de datos y esa caché. Ante un gran conjunto de objetos (las redes **EPANET** pueden aportar cientos de miles de ellos) y/o objetos con millones de observaciones (el caso de la plataforma **SCADA**), nos damos cuenta de los requerimientos hardware-RAM que se establecerían en un despliegue en producción.

La réplica de datos no es un problema grave, pero realmente la organización ya costea el alojamiento de esa información en una base de datos existente. No parece ser la mejor idea replicar de nuevo la información, y mantenerla sincronizada, si ello no es estrictamente necesario. El segundo aspecto es más problemático, la gestión de la caché sí que puede provocar la necesidad de grandes recursos hardware, y por tanto costes, o incluso llegar a ser inabordables.

El proyecto finalmente adopta una solución innovadora que fije los dos problemas: ***Implementar en el software original la capacidad de inyectar datos "al vuelo" desde "plugins" externos al núcleo del servicio***. Este desarrollo ha requerido cambiar el comportamiento actual de la caché interna de datos, de definir un conjunto de interfaces que se integren en los procesos que maneja la caché, y del desarrollo de "*plugins*" implementando estos interfaces para la inyección de sus propios datos. Y todo transparente al comportamiento actual del servicio y por supuesto a las aplicaciones cliente.

Es importante resaltar que la inyección complementa los datos registrados en el servicio de forma habitual, el resultado puede contener registros de la base de datos central, y registros de cada uno de los "*plugins*" instalados.

El proyecto siempre ha pretendido respetar la filosofía colaborativa de **GeoSmartCity** y **52ºNorth** y todo el desarrollo implementado ha querido ser ofrecido a la comunidad de forma gratuita. El código finalmente ha sido integrado en una rama específica del repositorio del proyecto de **52ºNorth SOS**:

https://github.com/52North/SOS/tree/feature/dynamic_observable_objects

En la siguiente imagen se muestra el "*pull request*" al repositorio de **52ºNorth SOS** con todo el código implementado:

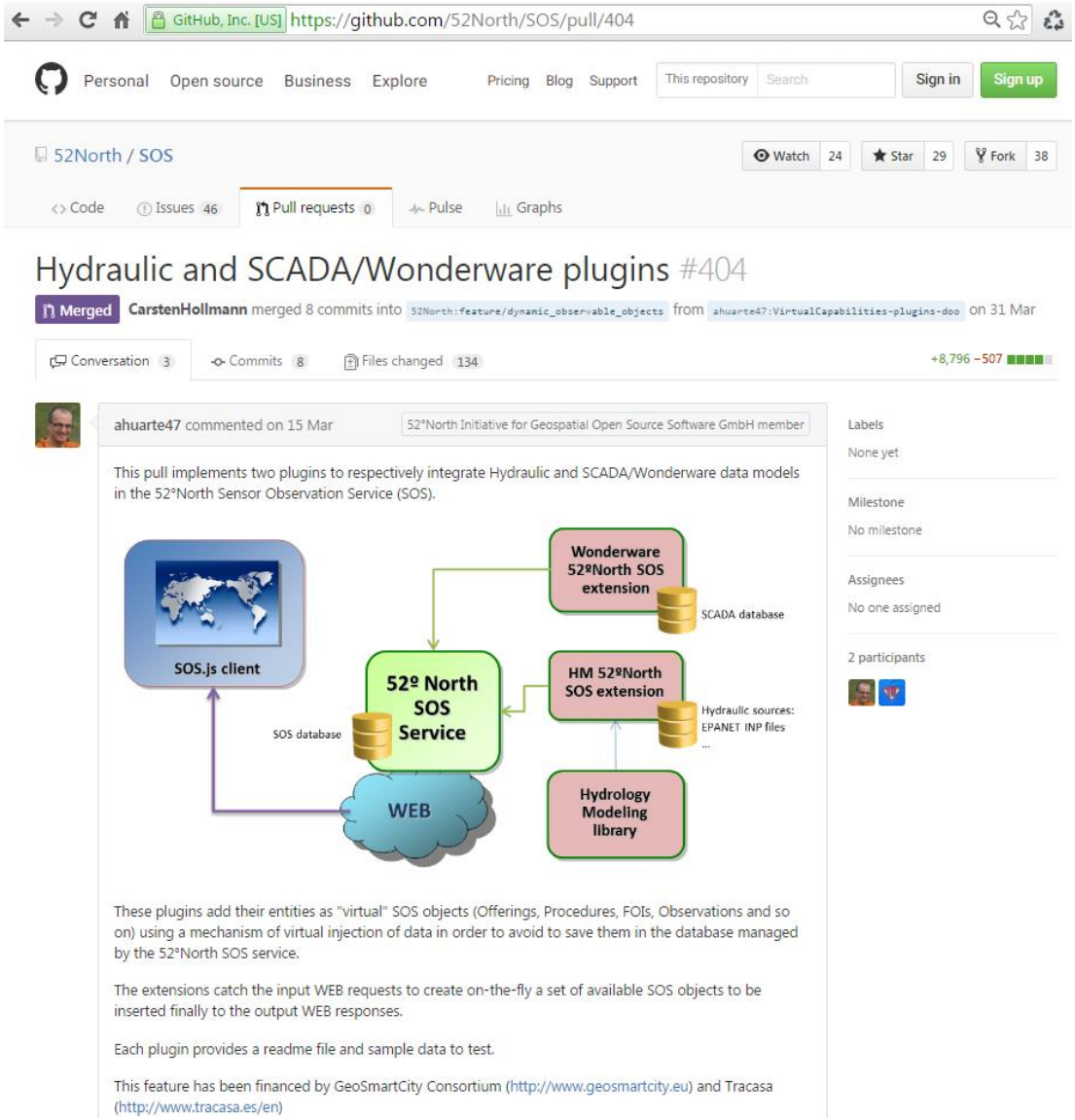

Figura 6: Pull merged: https://github.com/52North/SOS/tree/feature/dynamic_observable_objects

El desarrollo se puede dividir en dos partes complementarias. Una primera parte con las modificaciones necesarias en el core del servicio para soportar la inyección al vuelo de datos. Y un segundo paso, el desarrollo de los "plugins" en sí mismos, uno para cada fuente datos a integrar (**SCADA** y **EPANET**), que inyectan temporalmente su propia información.

El software define un nuevo interfaz que ofrece la posibilidad de invocar esta inyección de datos y que puede ser de nuevo reutilizado en la implementación de otras fuentes de datos. Las extensiones de **SCADA** y **EPANET** son simples casos de uso. También se ha desarrollado un paquete java con clases de utilidad para facilitar esta integración y que las dos nuevas extensiones utilizan.

Extensión SCADA para 52°North SOS

Este "*plugin*" es el responsable de inyectar los datos que la plataforma **SCADA** gestiona. En nuestro caso piloto la plataforma es una solución ***Wonderware*** [9] con su propio esquema de datos y que corre contra una base de datos SQL Server.

La extensión es sencilla, ejecuta consultas SQL a las tablas de la base de datos conforme a los criterios especificados en la petición web original. El componente integra los registros que cumplen

esos filtros de entrada como objetos normales **SOS**. El resultado respeta el comportamiento actual del servicio, pero inyecta de forma temporal sólo lo requerido por la petición web de entrada.

El mapeo del modelo de datos **Wonderware** a objetos conforme al estándar **SOS** es sencillo; cada sensor físico registrado se corresponde con un objeto "*offering*" con un método o "*procedure*" que contiene una determinada "*observedProperty*". El objeto "*featureOfInterest*" que da la componente geográfica al sensor se establece por configuración en un **shapefile** [10] con sus geometrías relacionadas por clave. Es posible el uso de otros formatos geográficos pues el software utiliza la librería opensource **GeoTools** [11] que soporta muchos otros.

Por ejemplo, ante una petición "*GetObservation*" de un objeto sensor SCADA:

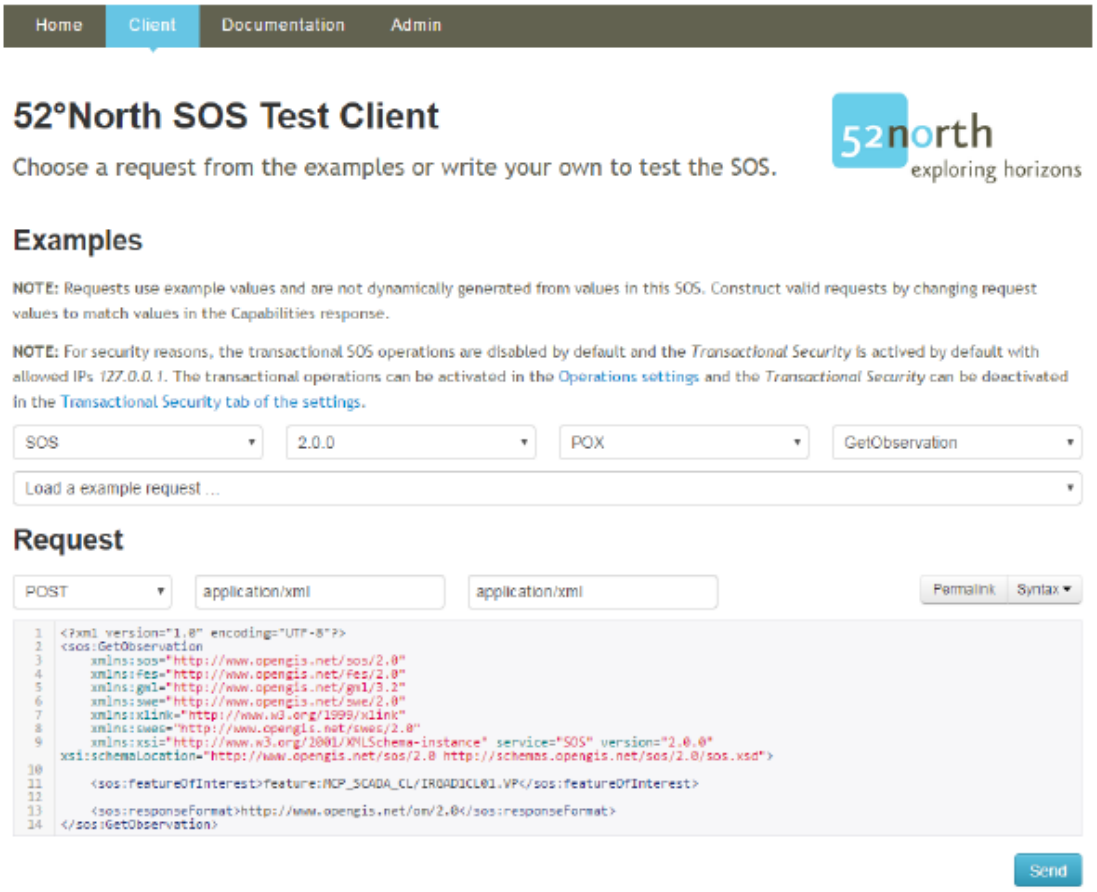

Figura 7: Petición "*GetObservation*" de un objeto sensor SCADA

La respuesta que se obtiene cumple como se espera el estándar SOS:

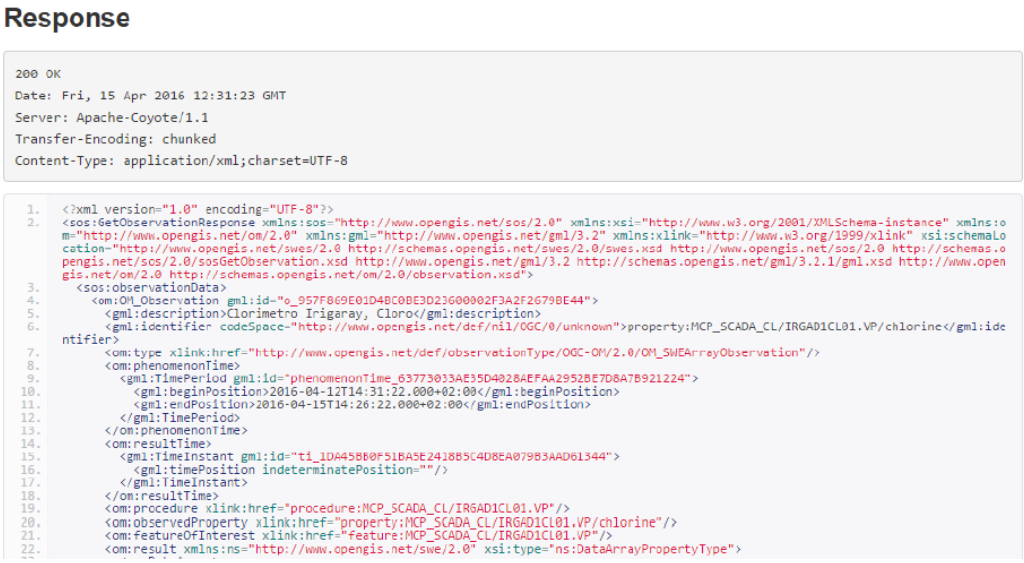

Figura 8: Parte de la respuesta del servicio SOS al "*GetObservation*" de un objeto

Las observaciones del sensor físico son mapeadas como series temporales instante+valor y contienen los registros "históricos" y el último valor o "*live*" de la propiedad observada. Como la extensión utiliza los filtros espaciales y temporales de la petición web original, se puede dar el caso de que el dato "*live*" no se incluya en la respuesta, o incluso no haya datos que devolver.

Para evitar que la petición cree una respuesta web con demasiados registros, es posible establecer por configuración un valor máximo de entidades a devolver. Hay que tener en cuenta que la aplicación cliente podría pedir todas las observaciones de un sensor, o especificar un filtro muy vago de tiempo, y el servicio podría tener que devolver cientos de miles de registros en una respuesta XML. No es necesario decir qué sucede con eso.

Extensión EPANET para 52ºNorth SOS

Este "plugin" es el responsable de inyectar una red hidráulica modelizada en **EPANET** y los resultados de su simulación mediante los algoritmos matemáticos de este modelo. La extensión soporta tantas redes como se desee y el punto de partida de cada uno de ellas es el conocido fichero **INP** con todos los datos geométricos y operacionales de los objetos de la red.

El componente carga y resuelve de forma automática las redes que se configuran integrando sus datos y resultados como un conjunto de sensores-observaciones **SOS**. El proceso genera por cada modelo **INP** de entrada una base de datos interna en formato **SpatiaLite** [12] con los datos a inyectar finalmente en el servicio.

Cada uno de los elementos de la red se mapea a un grupo de objetos "*offering*", "*procedure*", "*featureOfInterest*" y un conjunto de "*observedProperties*". Por ejemplo, una tubería se mapea como un sensor "virtual" que aporta sus variables hidráulicas resueltas de la simulación (presión, caudal, calidad de agua, pérdidas unitarias,…) como "*observedProperties*". Los demás tipos de objetos aportan sus propios atributos y se puede consultar la documentación de EPANET para averiguar cuáles son.

LIBRERÍA GSC-SOS.JS

Una vez creados y configurados los servicios que nos facilitan el acceso a los datos a través del estándar SOS. El siguiente paso es el de explotar adecuadamente esa información desde un visualizador genérico web de mapas. Para ello, en el marco del proyecto se ha implementado una nueva librería en JavaScript que cumple con el estándar SOS 2.0 y que ha sido desarrollada con una arquitectura más modular y agnóstica que la actualmente ofrecida por 52°North (pensada para SOS 1.0 y **OpenLayers** 2 [15]), haciendo sencilla su posible extensión para integrarse con cualquier API de

desarrollo web.

Además en el marco de este proyecto se ha desarrollado la extensión para su utilización con OpenLayers 3 [15].

Tanto el código fuente y el acceso a los test están disponibles en https://github.com/GeoSmartCity-CIP/gsc-sos.js

PILOTO WEB

El paso final del proyecto ha consistido en el desarrollo de una aplicación web, utilizando todos los componentes descritos anteriormente, que permite a los gestores de la Mancomunidad controlar y monitorizar el estado de la red de Abastecimiento y comparar los datos en tiempo real con valores de simulación y valores históricos.

Para la realización de esta web se ha utilizado el **API SITNA** [16], librería JavaScript basada en OpenLayers 3 que facilita el desarrollo de clientes web de mapas.

El piloto muestra un control de capas de información geográfica que permite visualizar o no la siguiente información:

- Red de sensores SCADA, compuesta de clorímetros y caudalímetros
- Red EPANET, red de abastecimiento de la MCP
- Red de Abastecimiento y Saneamiento según el modelo INSPIRE para el proyecto GeoSmartCity.

A continuación se muestran dos imágenes con ejemplos de la funcionalidad más destacada.

En el primer caso podemos elegir la propiedad observada, seleccionando uno de los valores en el desplegable "*sensor observation*". En este caso se ha seleccionado el flujo o caudal de agua que pasa por un punto. Como resultado la aplicación nos muestra dicha información en el mapa para todos los sensores de la red que miden la propiedad seleccionada.

Otro tipo de consultas que se pueden realizar es clicando en un punto del mapa y obteniendo toda la información que miden en ese punto. Esta consulta es parametrizable entre dos fechas y nos devuelve los valores recogidos cada hora. La información resultante es mostrada en modo tabla y modo gráfica, siendo además descargable en formato Excel.

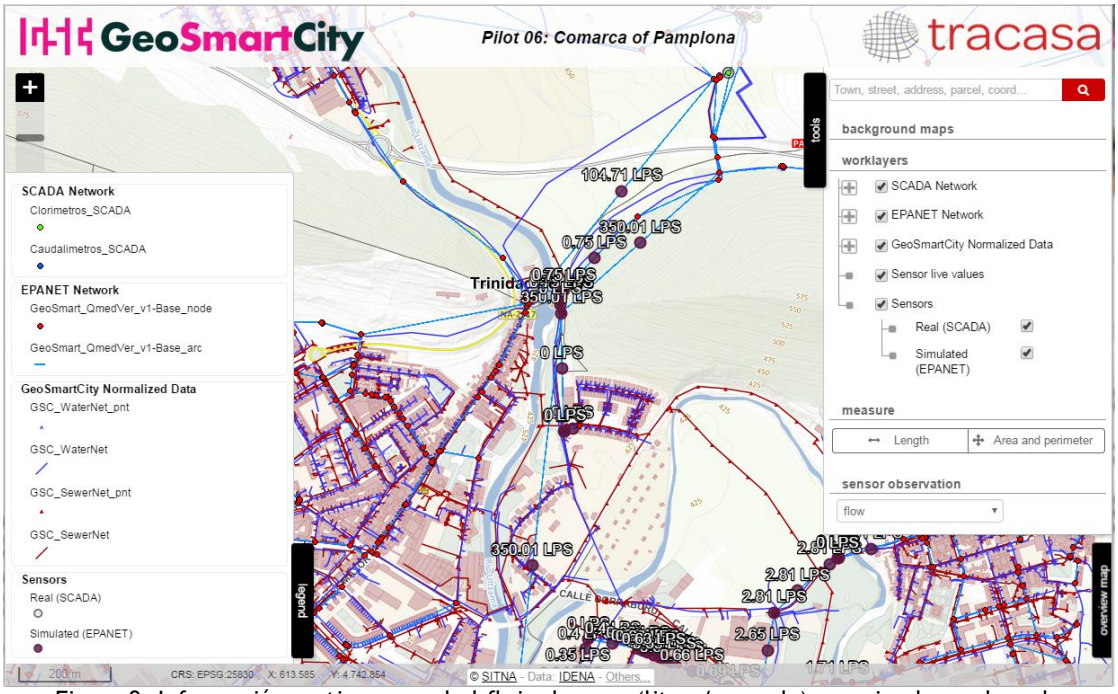

Figura 9: Información en tiempo real el flujo de agua (litros/segundo) que circula por la red

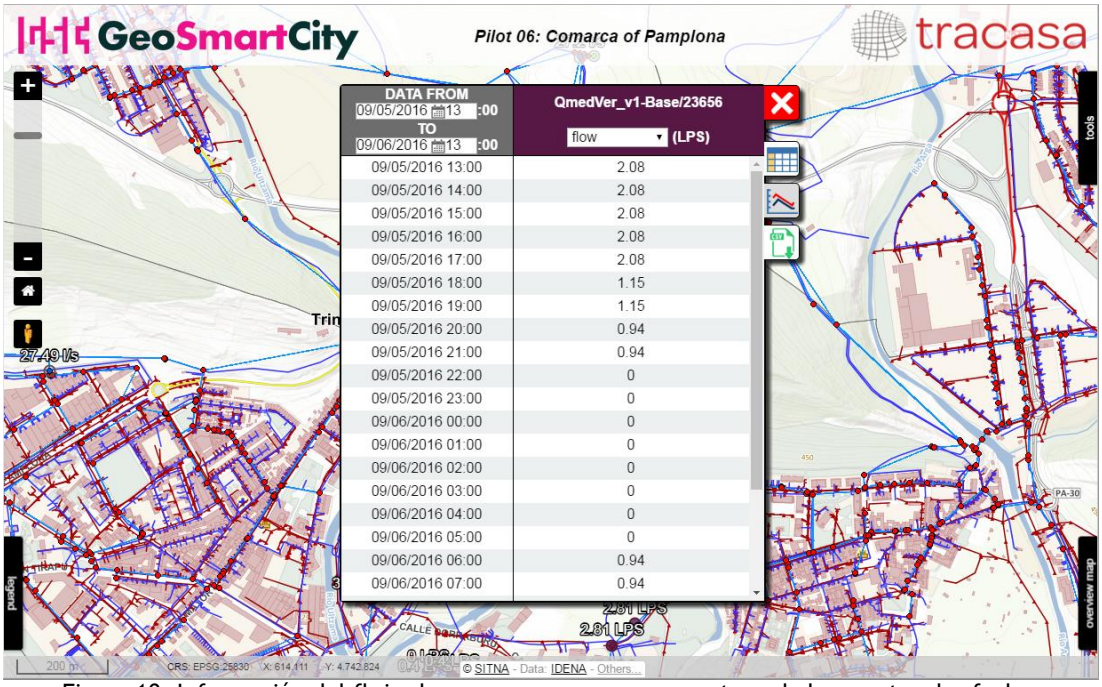

Figura 10: Información del flujo de agua que pasa por un punto cada hora entre dos fechas

FUTUROS TRABAJOS

Queda pendiente la implementación de mecanismos para que el usuario pueda notificar cambios en la red hidráulica (cambios en las calibraciones de los elementos, roturas o cortes, alta de nuevos objetos) y presentar en la aplicación cliente un nuevo conjunto de resultados de simulación de la red **EPANET**.

Es posible también la integración de otros tipos de modelos hidráulicos, **HEC-RAS** [13] o **SWMM** [14] que proporcionarían un completo "framework" para la gestión de recursos hídricos conforme al

estándar **SOS**.

También podría ser interesante implementar un mecanismo de alertas para notificar eventos de riesgo en la red.

AGRADECIMIENTOS

A **GeoSmartCity** por dar soporte y financiación a la iniciativa.

A la **Mancomunidad de la Comarca de Pamplona** por colaborar activamente en el desarrollo del proyecto piloto. Por proporcionar las redes y modelos hidráulicos, y al soporte para encaminarnos correctamente por estas tecnologías. Especial agradecimiento a Javier Lerga y Jokin Egüés.

A **52º North** por el software desarrollado y por ofrecerlo gratuitamente a la comunidad. Y al apoyo de sus técnicos para la integración final de este desarrollo en el proyecto.

A **Tracasa**, y los compañeros y colaboradores en el proyecto; Álvaro Huarte, Carlos Sabando, María Cabello, Raúl Orduna, Javier Ruiz, Jose Luis Hernández, Garazi Lacunza e Iván Pérez. Sin ellos no hubiera sido posible la integración final de tantas partes heterogéneas del proyecto.

Y a todos ellos por su apoyo a la innovación y a su impulso para la creación de proyectos para beneficio común de la sociedad.

REFERENCIAS

[1] GeoSmartCity: http://www.geosmartcity.eu/

[2] MCP: http://www.mcp.es/

[3] Sensor Observation Service: http://www.opengeospatial.org/standards/sos

[4] SCADA: https://es.wikipedia.org/wiki/SCADA

[5] EPANET: https://www.epa.gov/water-research/epanet

[6] 52°North: http://52north.org/

[7] Repo github del servicio 52°North SOS: https://github.com/52North/SOS

[8] Licencia GNU: https://es.wikipedia.org/wiki/GNU_General_Public_License

[9] Wonderware: http://software.schneider-electric.com/wonderware/

[10] Shapefile: https://es.wikipedia.org/wiki/Shapefile

[11] GeoTools: http://www.geotools.org/

[12] SpatiaLite: http://www.gaia-gis.it/gaia-sins/

[13] HEC-RAS: http://www.hec.usace.army.mil/software/hec-ras/

[14] SWMM: https://www.epa.gov/water-research/storm-water-management-model-swmm

[15] OpenLayers http://openlayers.org/

[16] API SITNA: http://sitna.navarra.es/geoportal/recursos/api.aspx

AUTORES

Juan Luis CARDOSO
jlcardoso@tracasa.es
Tracasa
Sistemas de Información Territorial

Álvaro HUARTE
ahuarte@tracasa.es
Tracasa
Sistemas de Información Territorial

Garazi LACUNZA
glacunza@itracasa.es
Tracasa Instrumental
Sistemas de Información Territorial

Iván PÉREZ
iperez@tracasa.es
Tracasa
Sistemas de Información Territorial

María CABELLO
mcabello@tracasa.es
Tracasa
Área Comercial y Consultoría