Desarrollo de un sistema de recogida y normalización de información meteorológica de carácter distribuido y social

ITSI Informática UNED (21/05/2015)

Gustavo Rodríguez Castillo

Dirigido por: Dr. D. Sebastián Dormido Canto

Supervisado por: Dr. D. Sebastián Dormido Canto

UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto de Fin de Carrera de Ingeniero Informático

DESARROLLO DE UN SISTEMA DE RECOGIDA Y NORMALIZACIÓN DE INFORMACIÓN METEOROLÓGICA DE CARÁCTER DISTRIBUIDO Y SOCIAL

Gustavo Rodríguez Castillo

Dirigido por: Dr. D. Sebastián Dormido Canto

Supervisado por: Dr. D. Sebastián Dormido Canto

Curso: 2014-2015

Desarrollo de un sistema de recogida y normalización de información meteorológica de carácter distribuido y social

Proyecto de Fin de Carrera de modalidad oferta específica

Realizado por: Gustavo Rodríguez Castillo

Dirigido por: Dr. D. Sebastián Dormido Canto

Supervisado por: Dr. D. Sebastián Dormido Canto

Tribunal calificador:

Presidente: D./D[a]. .....

(firma)

Secretario: D./D[a]. .....

(firma)

Vocal: D./D[a]. .....

*(firma)

Fecha de lectura y defensa: .....

Calificación: .....

RESUMEN

Durante mi carrera profesional me he desempeñado en diferentes áreas funcionales desde el punto de vista de la construcción de software. Una de las áreas que me han resultado siempre de mayor interés es la de los medios telemáticos aplicados a los procesos meteorológicos, desde la obtención de la observación hasta el apoyo a la predicción y la generación y tráfico de productos. He encontrado que las Agencias e Instituciones que diseñan y ejecutan procesos meteorológicos disponen de redes heterogéneas de observación y de herramientas para su normalización. Sin embargo, he encontrado también que existe una comunidad de aficionados a la meteorología que no cuentan con ningún tipo de red de observación, pero que sí trabajan con frecuencia con una o varias estaciones de observación más o menos caseras. Estos aficionados intercambian conocimientos sobre la materia a través de foros y redes sociales, y sin embargo hasta donde conozco no disponen de una plataforma que vaya más allá de los productos específicos de cada estación y que pueda construir una red de estaciones de observación totalmente distribuida a partir de las observaciones de sus estaciones individuales. La idea de construir un germen de dicha herramienta resultaba interesante y por ello se planteó el proyecto en estos términos.

Para la ejecución del mismo se ha procurado realizar un análisis formal de requerimientos, casos de uso y subsistemas de análisis, a partir de lo cual se ha organizado el desarrollo en ciclos de construcción iterativa que permitieran la producción más o menos rápida de productos intermedios y al mismo tiempo la posible revisión del alcance y requisitos entre cada uno de los ciclos. Esta metodología de seguimiento está inspirada en Scrum, si bien el hecho de poder realizar trabajo útil sólo algunos días en horarios alternos y de manera individual distorsiona la filosofía de seguimiento diario de dicha metodología. No obstante, se han utilizado herramientas de seguimiento y estimación propias de Scrum.

El producto final puede verse como un punto inicial a partir del cuál tratar de construir dicha red de observación distribuida. Proporciona funcionalidades de obtención y control de información, cálculos derivados, representaciones de diverso tipo de la información, así como de generación de productos en forma de comentarios en Twitter. De esta forma se ejemplifican un buen número de procesos meteorológicos, si bien todos ellos podrían encontrar ampliación casi hasta el infinito.

ABSTRACT

My professional career has been centered on several functional frames from the point of view of software development. One of those frames has been the application of Information Technology solutions on meteorological processes such as observation, prediction and generation and exchange of multiple products. These solutions have always aroused my interest towards such processes. I have found that both private and Government Agencies count on a number of deployed heterogeneous observation networks along with multivendor software tools that normalize the collected information. However, I have as well found a certain amount of amateur people who are highly interested in meteorology and, while they do not maintain big deployed observation networks, often install and maintain one or several meteorological stations for home use. Such amateurs exchange their knowledge of the subject upon internet forums and social networks, but as far as I know there is no available platform to enable the sharing and normalization of their individually provided observations in order to build up a big, distributed network of observation stations. Rather, they rely on the local software provided by each station, which works isolated and is guided by no exchange standards. The idea of building a small core for such platform was of interest and thus the motivation to define this master work.

In order to fulfill the project needs, a formal requirement analysis along with use cases and subsystem design has been accomplished. Based on the results of such analysis, the development itself has been organized into a process of incremental and short development efforts, or sprints. Such sprints have allowed providing intermediate releases as much as the ability to redefine to some extent the scope and requirements of the project between sprints. This methodology was inspired by Scrum, though the fact of not being able to employ fix and constant work times has slightly distorted the approach. However, the follow-up and estimation tools of Scrum have been used.

The final product may actually be regarded as a start-up point on which the distributed observation network can be built. It provides basic features relating the collection, control, derivation and multiple types of display, as much as the product generation in the form of Tweets. A good number of meteorological processes have been so demonstrated, based on which a potentially unlimited number of extensions could appear.

Desarrollo de un Sistema de recogida y normalización de información meteorológica de carácter distribuido y social

PALABRAS CLAVE / KEYWORDS

Ilustración 1 - Tagcloud de palabras clave

Palabras clave: observación, estación, Sistema, calidad, datos, Twitter, usuario, información, red, meteorología.

Keywords: observation, station, system, quality, data, Twitter, user, information, network, meteorology.

ÍNDICE

ÍNDICE DE FIGURAS

INTRODUCCIÓN

OBJETIVOS

El objetivo principal del Sistema es el de crear una **red distribuida de observación meteorológica de propósito social**. Dicho objetivo principal puede descomponerse a su vez en diferentes subobjetivos:

- Elaborar mecanismos de **recopilación de observaciones** meteorológicas desde estaciones o sensores estándares del mercado.
- Construir un sistema de **normalización y control** de información observada, que permita la composición de reglas de calidad de la información generada y que a su vez sea capaz de calcular **medidas agregadas o derivadas** de las anteriores.
- Consolidación de la información recopilada de manera anónima en un sistema **web geolocalizado**
- Proporcionar mecanismos de **publicación de comentarios** en redes sociales existentes (**Twitter**).

El público objetivo del Sistema serían aquellos **aficionados** a la Meteorología que disponen de estaciones más o menos **caseras** y que difícilmente pueden compartir información normalizada. El objetivo sería homogeneizar el uso individualizado de estaciones meteorológicas de modelos heterogéneos a través de un sistema estandarizado que ofrezca herramientas comunes de explotación y de calidad datos en un ámbito social.

PLANIFICACIÓN

La construcción del Sistema se ha planteado mediante una serie de iteraciones cortas o sprints del tipo de las planteadas por un equipo de trabajo Agile. En concreto se definieron **3 iteraciones**, con las dos primeras enfocadas a producir el Sistema plenamente funcional mientras la última sumaba las funcionalidades de configuración del mismo. El esfuerzo global se calculó en días naturales, estimando una

disponibilidad media de 1,5 horas de trabajo real en cada día natural. La estimación de esfuerzo combinado de las 3 iteraciones se estimó en 114 días naturales, es decir, unas 171 horas de trabajo real. La ejecución real abarcó desde el 12/12/2014 hasta el 30/04/2015, lo que supone 169 días naturales. Sin embargo, el flujo de trabajo no fue constante y es difícil conocer el esfuerzo real en horas producido.

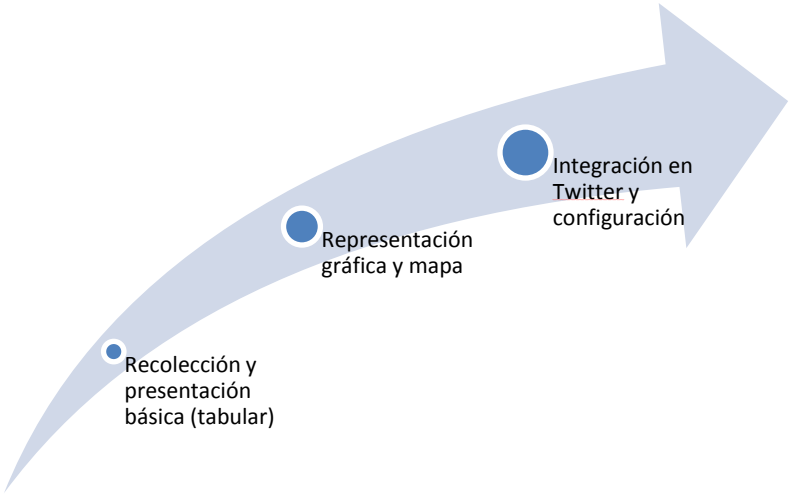

Ilustración 2 - Planificación incremental

ITERACIÓN 1: RECOLECCIÓN Y PRESENTACIÓN BÁSICA (TABULAR)

El objetivo de la iteración 1 es construir:

- Los **elementos comunes** del proyecto como: modelo de datos, elementos de persistencia, clases básicas de interfaz de usuario, las interfaces de los plugins de recolección, etc.
- El **núcleo de recolección** casi completo, incluyendo:
  - Núcleo de lectura y almacenamiento de observaciones
  - Plugin de estación de ejemplo
  - Control de calidad
- Función básica de **consulta de observaciones en formato tabular**

El esfuerzo para completar la iteración se estimó en **42** días naturales.

ITERACIÓN 2: REPRESENTACIÓN GRÁFICA Y MAPA

El objetivo de la iteración 2 es construir:

- El núcleo de recolección en su completitud, añadiendo la función de **cálculo de observaciones derivadas**.
- Las funciones avanzadas de representación de observaciones, incluyendo:
    - Representación **gráfica de observaciones**
    - Representación **tabular de observaciones derivadas**
    - Representación **gráfica de observaciones derivadas**
    - **Mapa de estaciones**
    - **Búsqueda de estaciones**

El esfuerzo para completar la iteración se estimó en **50** días naturales.

ITERACIÓN 3: INTEGRACIÓN EN TWITTER Y CONFIGURACIÓN

El objetivo de la iteración 3 es completar el Sistema implementando las últimas funcionalidades, a saber:

- Generación de **comentarios** e integración con **Twitter**
- **Configuración de la estación**
- **Configuración del perfil de usuario**

El esfuerzo para completar la iteración se estimó en **22** días naturales.

CONTEXTO DE LA OBSERVACIÓN METEOROLÓGICA

AGENCIAS E INSTITUCIONES

Los procesos de observación, control, predicción y análisis meteorológico y climatológico son en general competencia de Agencias u Organismos estatales. Originalmente dichas Agencias solían estar ligadas al Ministerio de Defensa, de tal forma que aún en la actualidad en algunos países se encuentran en dicha situación. En la mayoría de los casos sin embargo, si bien pueden existir Organismos aún ligados al Ministerio de Defensa, las Agencias Estatales de Meteorología han pasado a tener entidad propia o a pertenecer a otros Ministerios, como el de Agricultura y Medio Ambiente en el caso español.

En España de forma paralela o adicional pueden existir Agencias ligadas a ciertas Comunidades Autónomas, Agencias de objetivo específico, como las Cuencas Hidrográficas y los Puertos del Estado, o Agencias privadas que proveen servicios meteorológicos a compañías para las que dichos servicios resultan vitales, como pueden ser las empresas energéticas que explotan energías renovables.

Por último existe un intercambio de información meteorológica de carácter específicamente aeronáutico con aeropuertos y aeródromos y todo un catálogo de información muy variada que se trafica con Agencias Meteorológicas internacionales.

Cada una de estas Instituciones puede disponer de una o varias **redes de estaciones de observación** de muy diferente carácter, en localizaciones geográficas variadas, que forman la base de los posteriores procesos meteorológicos y climatológicos. Las estaciones de observación (millares) serán a su vez provistas por muy **diversos fabricantes**, serán de **diversos modelos** y estarán compuestas por **diversos sensores**. Esto introduce una complejidad en el tratamiento y normalización de la información que es necesario resolver a través de herramientas centralizadas.

Es necesario tener en cuenta que cada estación meteorológica es básicamente una composición de los siguientes elementos:

- **Una serie de sensores** como termómetros, anemómetros, barómetros, pluviómetros, etc. Los mencionados son los más comunes sin duda, pero hay que tener en cuenta que según la geografía existirán también sensores de la capa de nieve, de la luz reflejada o del nivel y velocidad de las mareas (en el caso de las boyas).
- **Un datalogger**, que es el firmware que permite la conexión de los sensores y coordina la obtención de información de los mismos y la almacena internamente. El datalogger almacena información instantánea, por ejemplo cada segundo, de todos los sensores conectados a la estación. Partiendo de la información instantánea, el datalogger va a generar información resumida (medias) en un período que según el fabricante será normalmente de 10 o 15 minutos, si bien no existe un estándar al respecto. Finalmente este datalogger ofrece mecanismos para acceder y leer esta información o bien para enviarla por medios heterogéneos, como FTP, al destino configurado.
- **Un software de operación y consulta** de la estación, cuyas funcionalidades vienen determinadas por el fabricante, y que está completamente asociado a la estación (más exactamente al datalogger) para el que se configura en una relación 1:1.

Descrita la situación, es posible imaginar el problema derivado de que diferentes dataloggers proporcionen información de sensores distintos, con **formatos heterogéneos** (XML, CSV, etc.) y a través de **protocolos de transporte diferentes** (FTP, TCP, etc.).

Las Agencias Meteorológicas disponen de herramientas multifabricante que son capaces de obtener o recopilar la información de sus diversas redes de estaciones, analizar dicha información en base a diferentes algoritmos de calidad, almacenarla y publicarla.

Además de las redes institucionales, que por supuesto están compuestas por estaciones de alta calidad y de costes muy elevados, existe una **red de aficionados** que disponen de modelos de estaciones caseros muy variados y que utilizan con fines exclusivamente personales. Generalmente dicha información será presentada al usuario a través del software de la propia estación y no siempre será extraíble por un software externo. Sin embargo, muchas estaciones modernas disponen de mecanismos de interrogación basados en protocolos propios sobre conexiones USB, Bluetooth o incluso modelos de datos relacionales.

La idea sería permitir a los usuarios unirse a una red de aficionados donde cada uno aporta **la información producida por su propia estación**, siempre que ésta sea integrable de alguna manera, y a su vez recibir como contraprestación la posibilidad de explotar la información de todos los demás usuarios del Sistema. En este escenario es vital por un lado diseñar un Sistema **que pueda escalar tanto en volumen de usuarios y estaciones, como en heterogeneidad de las propias estaciones**. Especialmente importante es que el diseño permita un **esfuerzo mínimo a la hora de integrar un modelo nuevo de estación**, donde exista un pequeño elemento capaz de comunicarse con la estación y parsear la información, y todo lo demás sea común y completamente independiente de fabricantes, modelos y sensores instalados.

PROCESOS METEOROLÓGICOS Y CLIMATOLÓGICOS

La observación meteorológica es sólo el primer paso de lo que podemos denominar el **dato meteorológico**. Sin embargo, se trata de una información vital para el posterior proceso asociado.

A grandes rasgos, se pueden describir los procesos de análisis meteorológico agrupándolos según el ciclo de vida del dato:

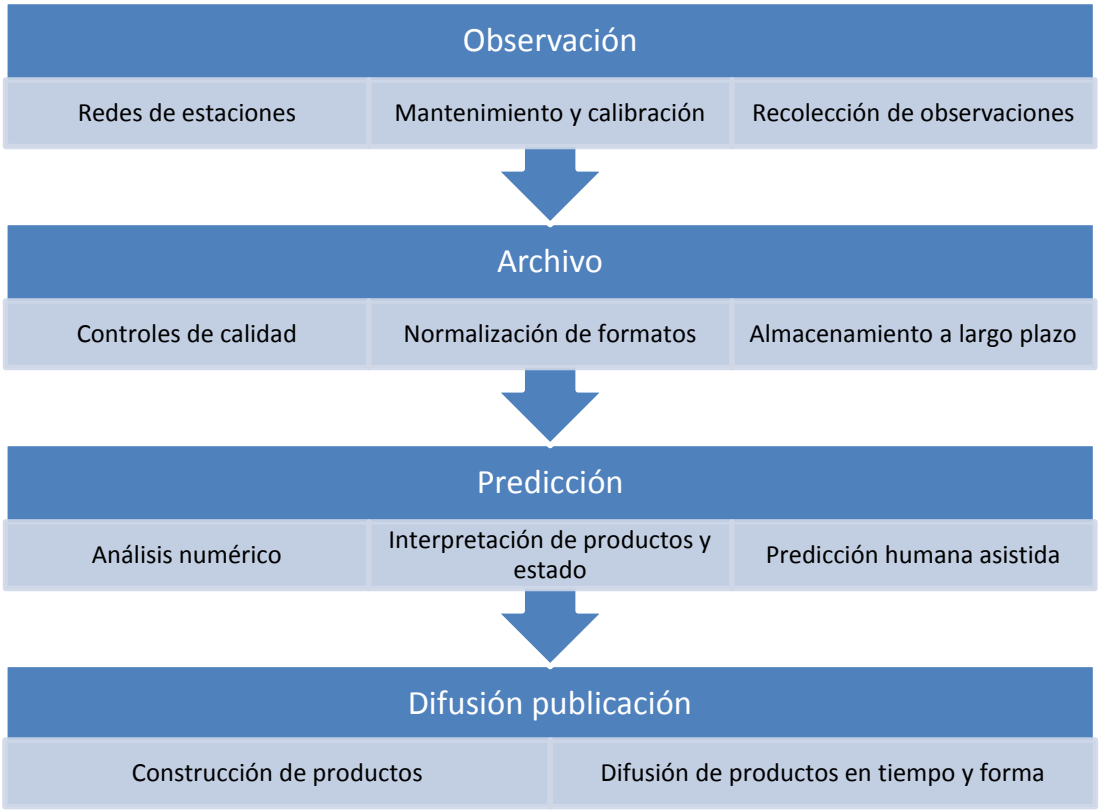

Ilustración 3 - Procesos meteorológicos

OBSERVACIÓN

Son procesos centrados en la **recolección puntual de un dato observado de calidad**. Ello implica no sólo el desarrollo e implantación de redes grandes y variadas de estaciones, sino también el correcto mantenimiento y calibración de todos los elementos involucrados en el proceso (desde las estaciones hasta las redes de comunicaciones).

En lo tocante al Sistema aquí planteado, se sitúa como una fuente de información más dentro de dicho proceso de observación. Puesto que el Sistema cuenta con un análisis de calidad de los datos, si bien en un estado básico, se puede entender asimismo como una herramienta de apoyo para la detección de problemas en sensores o datalogger (medidas erróneas o inexistentes) o en los mecanismos de comunicación con la estación (ausencia de datos, errores de conexión).

ARCHIVO

Una vez recepcionada la información observada, se realizan sobre ella **controles de calidad de diverso tipo** (control de límites físicos, control de variabilidad temporal, control de consistencia espacial, etc.). En adelante el dato se considera siempre como el propio dato más su flag de calidad, que será fundamental para el tratamiento posterior del dato.

La información ya liberada de la heterogeneidad original se normaliza de acuerdo con los diferentes estándares de formato y se almacena en crudo sin límite temporal alguno.

El Sistema objeto del proyecto aplicará un control de calidad de límites configurables. En cuanto a su almacenamiento, está planteado como un estado temporal del dato que puede abarcar varios meses o incluso años pero que debería apoyarse en un archivo masivo en medios especializados más a largo plazo.

PREDICCIÓN

Los datos de observación sirven como base para la ejecución de **procesos de análisis numérico complejos** sobre los mismos. Estos modelos numéricos son diseñados y ejecutados por Agencias Nacionales y Supranacionales en supercomputadores dedicados a dicho análisis numérico. Como resultado se generarán una serie de productos que permiten a los Departamentos de Predicción realizar las predicción de condiciones de muy diversa índole (avisos marítimos, avisos de tierra, alarmas) y referencia temporal.

DIFUSIÓN/PUBLICACIÓN

Como resultado de los diferentes procesos **se generarán productos** en formatos generalmente (pero no siempre) definidos por estándares internacionales de la WMO o la ICAO, que finalmente serán **intercambiados entre las diferentes Agencias e Instituciones por medios telemáticos**.

DEFINICIÓN DE ALCANCE

Ya se ha indicado anteriormente el objetivo principal: crear una **red distribuida de observación meteorológica de propósito social**.

Y los subobjetivos en los que se descompone:

- **Recopilación** de observaciones meteorológicas.
- Control de la **calidad** de la observación **y cálculo** de variables derivadas.
- **Presentación** de la información geolocalizada.
- Publicación de comentarios en **Twitter** como ejercicio social.

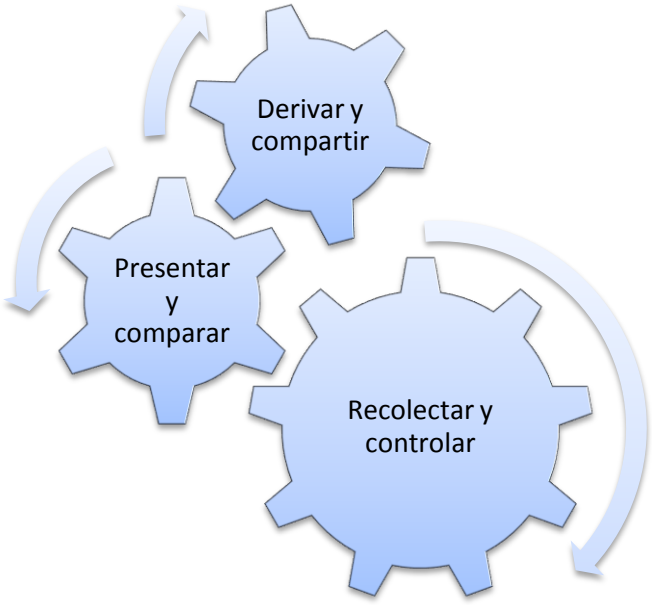

Ilustración 4 - Objetivos engranados

Para alcanzar dichos objetivos, el Sistema debe cubrir una serie de funcionalidades y seguir unas pautas de diseño que en definitiva constituyen el alcance del Sistema que se va a desarrollar. El alcance se descompone según grandes bloques funcionales como se detalla a continuación:

RECOPILACIÓN DE OBSERVACIONES

El hecho de obtener las observaciones de las diferentes estaciones meteorológicas supone la necesidad de implementar una integración con distintos modelos de las mismas que permita conectarse a sus dataloggers y extraer, periódicamente según configuración, las lecturas provenientes de sus sensores que han quedado almacenadas en dichos dataloggers o medios adyacentes.

Es importante destacar respecto a esta capacidad de obtener periódicamente observaciones de las diferentes estaciones existentes, los siguientes aspectos:

- El **núcleo** o elemento de recolección de observaciones debe ser en su mayor parte totalmente **genérico e independiente del modelo** final de estación a conectar. De esta forma se reducirá el esfuerzo de la integración de un nuevo modelo de estación a la implementación más mínima posible, típicamente a los elementos de comunicación con la estación y de parseo de la información obtenida de la misma.
- El puente entre este núcleo común y la especificidad de la estación debe cumplir lógicamente una interfaz propuesta dentro del propio Sistema que permita la conexión de nuevos modelos con **un sistema de "plugins"**.
- **No está dentro del alcance** de este proyecto la elaboración de ninguna integración con ningún modelo de mercado, puesto que el foco está localizado en la herramienta general de tratamiento posterior de la información.
- **Sí se considera incluido en el alcance** del proyecto la implementación de un caso de ejemplo de integración con una estación de simulación, es decir, con un formato de datos y de comunicación ficticio, si bien se procurará que sea acorde a lo que se pueda encontrar en el mercado real.
- Cada usuario podrá conectar su propia estación y para ello deberá caracterizar:
    - El **modelo** de estación **y el plugin** que la maneja
    - A partir de dicho plugin, será posible obtener del mismo los **parámetros que es preciso configurar** para la lograr la integración de la estación. Por ejemplo, el plugin podría indicar que es necesario configurar un host, puerto, usuario y password para una conexión FTP.

- El usuario tendrá la capacidad de **configurar los parámetros** provistos por el plugin.
- Asimismo el plugin proporcionará la **lista de variables** que mide el modelo de estación (que en definitiva se deriva de los sensores instalados).
- El usuario podrá **configurar los valores umbrales** para cada una de dichas variables que serán considerados para la ejecución de los controles de calidad durante el almacenamiento (ver Almacenamiento y estandarización de observaciones)

ALMACENAMIENTO Y ESTANDARIZACIÓN DE OBSERVACIONES

El Sistema, independientemente de modelos de estación e incluso de las propias variables recuperadas (temperatura, humedad, etc.), presentará un modelo de almacenamiento de las mismas **único, comprensible y escalable**, que permita su crecimiento:

- En **volumen**: puesto que está dirigido a una comunidad de aficionados a la meteorología potencialmente grande, aunque nunca masiva, se debe diseñar de manera que una cantidad grande de usuarios, estaciones y, especialmente, observaciones, no suponga una degradación significativa de tiempos de respuesta. Por supuesto, puesto que las observaciones pueden tener un tiempo de vida potencialmente infinito, es muy posible que llegado un punto, en un hipotético despliegue real, fuera necesario combinar estrategias de almacenamiento offline con cargas de datos antiguos bajo petición. El alcance no llega hasta dichas estrategias, pero sí es preciso que el diseño tenga en consideración esta posibilidad.
- En **tipología**: la mayoría de las estaciones meteorológicas cuentan con unos determinados sensores básicos para variables como temperatura, presión, humedad, precipitaciones, dirección y velocidad del viento, etc. Sin embargo la lista de variables es muy variable y susceptible de crecer a través de avances técnicos o científicos. El diseño debe considerar por tanto el hecho de que las

variables a almacenar pueden variar, hecho que no debe impactar en absoluto en el funcionamiento del Sistema.

Como parte del proceso de **post-almacenamiento** encontramos:

- La ejecución de **controles de calidad del dato**. Dentro del alcance se contempla la implementación de un control simple de límites umbrales, es decir, un control donde se compruebe únicamente que una determinada observación (relativa a una variable) no sea nunca inferior a un mínimo o superior a un máximo configurado. Es importante considerar que los **límites climatológicos no son fijos**. Por el contrario son muy variables según las zonas donde está ubicada la estación, no sólo en diferentes países, regiones, etc., sino incluso dentro de una misma región (son diferentes los valores que se pueden producir en entornos de montaña de los que se producen a pocos kilómetros, pero en la falda de la montaña). El diseño del Sistema debe asegurar la configurabilidad de dichos valores umbrales por cada usuario para su propia estación. Además de dichos umbrales configurables podrán desde luego existir valores umbrales globales (por ejemplo, para la dirección del viento el umbral debe estar entre 0 y 360 en todos los casos). Controles de calidad más avanzados, como los de consistencia temporal (la variabilidad de una variable no puede superar un umbral; o por el contrario, debe ser siempre superior a un umbral) o los de consistencia espacial (estaciones cercanas o en zonas comparables deben producir datos coherentes) quedan fuera del alcance inicial.
- El **cálculo de variables derivadas** a partir de la información recibida. A partir de determinadas variables es posible calcular otras (por ejemplo es típico reducir la presión al nivel del mar o calcular la temperatura del punto de rocío). Este tipo de cálculo queda fuera del alcance. Sin embargo sí se plantea la inclusión de cálculos de valores **mínimos, medios y máximos** de las diferentes variables recolectadas **para períodos de tiempo definidos**. Dichos períodos de tiempo serán inicialmente: mes, día, noche (00-06), mañana (06-12), tarde (12-18), tarde-noche (18-24). En cualquier caso, el diseño debe asegurar que la introducción de nuevos períodos de cálculo sea sencilla.

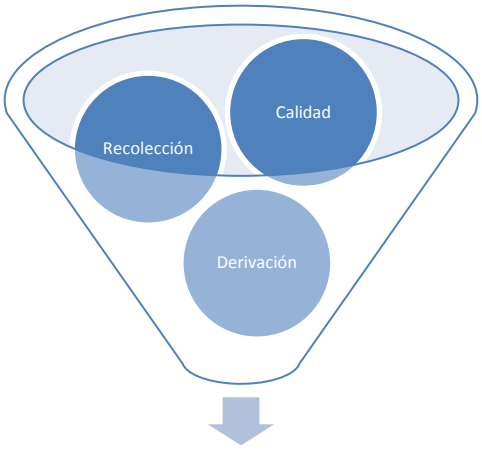

Ilustración 5 - Normalización de información

INTERFAZ DE LECTURA DE OBSERVACIONES

Una vez almacenadas las observaciones, controladas y calculadas las variables derivadas, se construirá un **interfaz de usuario web** que permita la representación de las mismas. Todas las estaciones dadas de alta serán accesibles para cualquier usuario del Sistema, y para todas ellas el interfaz de usuario presentará:

- Unas funciones de **búsqueda y localización de estaciones** basadas en criterios administrativos o espaciales, para su posterior selección.
- Una **representación tabular de las observaciones** obtenidas organizadas por fecha y en las que se resalten los posibles problemas de calidad identificados. Puesto que las observaciones de las diferentes variables medidas están agrupadas en periodos de tiempo (generalmente diezminutales), se considerarán las observaciones siempre como una tupla de información asociada al instante de observación común. Así pues, se podrá hablar de las observaciones de las 12:10 del día 01/04/2015 de una estación X, incluyendo en esta tupla todas las variables que se medían en aquel momento, estén éstas o no presentes en el Sistema.

- Una **representación gráfica de las observaciones** organizadas por fecha. Dicha representación gráfica podrá ser de tipo área, barras o líneas según se haya configurado la variable asociada.
- Una **representación tabular de las observaciones derivadas** (mínimas, medias, máximas) en los períodos de tiempo pertinentes.
- Una **representación gráfica de las observaciones derivadas** (mínimas, medias, máximas) en los períodos de tiempo pertinentes.
- La **geolocalización de estaciones** sobre un mapa GIS donde se presentan además datos globales de cada una de las estaciones.

INTEGRACIÓN SOCIAL

La idea de la integración con una red social de la información meteorológica obtenida proviene de un hecho tan cotidiano y fútil como "hablar del tiempo". Se trata en este caso de que esa charla sobre el tiempo esté basada en información observada sobre el terreno. En realidad se trata de un proceso que estaría encuadrado dentro de las actividades de **generación y difusión de productos**, que si bien generalmente son de tipo estándar, en este caso serían **concretamente comentarios en Twitter**.

Puesto que una estación va a generar, típicamente, 144 observaciones por cada variable y día, lo más adecuado parece tomar las **observaciones derivadas como base para esta actividad** de publicación de comentarios. En este caso, el Sistema deberá buscar para cada una de las estaciones que producen observaciones, aquellas observaciones derivadas que se consideren cerradas, es decir, calculadas sobre el 100% de la base posible. Por ejemplo, el valor de la temperatura media de un mes estará listo para ser publicado cuando se haya concluido dicho mes, y todas las observaciones de temperatura del mes cuentan con resultados de calidad favorables. En cualquier otro caso no se generará comentario alguno. Los comentarios agruparán todas las observaciones derivadas de cada observación, esto es, referirán la mínima, media y máxima de una determinada variable en el período de tiempo en el que se basa el cálculo.

La generación de comentarios legibles supondría idealmente el desarrollo de un interfaz de configuración que permita a cada usuario introducir comentarios propios, en uno o varios idiomas, y con unas reglas de publicación más o menos completas: variables que se publican, períodos utilizados, horarios de publicación, etc. Sin embargo, en el presente proyecto se generarán comentarios simples en castellano comunes a todos los usuarios.

GESTIÓN DE METADATOS Y PERFILES DE ESTACIONES Y USUARIOS

Cada usuario que se dé de alta en el Sistema dispondrá de un nombre de usuario y contraseña y la configuración básica de su perfil. Cada usuario podrá disponer en principio de una estación meteorológica que estará asociada a su perfil.

Se contempla como parte de la flexibilidad del Sistema la posibilidad de que el usuario configure hasta cierto punto el comportamiento del mismo en lo que respecta a su propia estación y a su perfil social en Twitter. Así pues, el usuario será capaz de configurar en el interfaz web:

- **El modelo de su estación**, a elegir de entre los registrados en el Sistema, ya que sólo aquellos para los que se dispone de plugin de integración serán presentados. Asociado a este modelo se encontrará naturalmente el tipo de conexión y de formato de la estación.
- El plugin de la estación declarará una serie de parámetros que son configurables por el usuario, típicamente **parámetros** necesarios para la comunicación con la estación, como direcciones IP, puertos TCP/UDP, usuarios, contraseñas, rutas, nombres de ficheros, etc. En el interfaz web podrá el usuario proporcionar valores a estos parámetros de forma que se establezca la conexión con la estación.
- El plugin de la estación declarará asimismo la lista de **variables** asociadas a la estación (y sus sensores) junto con los valores umbrales (mínimo y máximo) previstos por defecto por el fabricante para cada una de las variables. Además de ello, el usuario podrá sobrescribir estos valores umbrales para su propia estación, adaptándolos así a las condiciones concretas en las que la misma ha

sido instalada. Esto afectará directamente a los resultados de calidad obtenidos por el núcleo de recolección para cada una de las observaciones.

- La estación estará caracterizada por una serie de **metadatos** que serán también configurables vía interfaz web. Estos metadatos incluirán como mínimo la dirección postal y la latitud, longitud y altitud donde se ubica.
- Por último el usuario podrá configurar su propio **perfil de usuario**, incluyendo no sólo los datos básicos como nombre de usuario, contraseña, datos de contacto, sino además los datos de conexión al perfil de social en Twitter. Se entenderá en el momento en que se configura el perfil social, que el Sistema debe publicar comentarios relativos a las observaciones derivadas.

ANÁLISIS DE REQUISITOS

ESPECIFICACIÓN FORMAL DE REQUISITOS

REQUISITOS FUNCIONALES

RF1. Los usuarios podrán **autenticarse** mediante un par usuario y contraseña.

RF2. El Sistema mantendrá un registro **de estaciones meteorológicas** asociadas a los usuarios del mismo. Dichas estaciones producirán determinadas mediciones, como pueden ser la temperatura, humedad, velocidad y dirección del viento, etc.

RF3. El Sistema ejecutará con la periodicidad configurada una **lectura de observaciones** desde la estación meteorológica asociada.

RF4. Para cada estación se especificará el **protocolo de comunicaciones** necesario para llevar a cabo la lectura y el **formato de los datos** una vez realizada dicha comunicación.

RF5. Cada lectura que se produzca desde la estación se caracterizará mediante la siguiente información:

a. Fecha de **adquisición**.
b. Fecha de **observación** (de la propia medida por parte de la estación)
c. Lista de **observaciones** (mediciones)

Dichas lecturas serán almacenadas permanentemente para su posterior presentación, difusión o análisis.

RF6. Para cada medición de la lista, se comprobará si su valor está o no dentro de los **umbrales** configurados para la misma en la estación asociada. Los valores umbrales serán 2:

a. Los provistos por la propia estación que se considerarán **valores físicos**.
b. Los sobrescritos (configurados) por el usuario que se considerarán **valores actuales y locales**.

RF7. Cada medición estará asociada con un **resultado de control de calidad** cuando se hayan aplicado las comprobaciones umbrales. Dicho resultado podrá tomar los valores "**correcto**" si el valor está dentro de los umbrales físicos y locales, o "**sospechoso**" si el valor está fuera de alguno de ellos.

RF8. A partir de las mediciones producidas en períodos de tiempo fijos pero configurables, el Sistema calculará:

a. El valor **medio, mínimo y máximo** en tres períodos diferenciados del día:
    i. **Noche**: entre las 00 y las 06 horas
    ii. **Mañana**: entre las 06 y las 12 horas
    iii. **Tarde**: entre las 12 y las 18 horas
    iv. **Tarde-noche**: entre las 18 y las 24 horas
b. El valor medio, mínimo y máximo en un **día**
c. El valor medio, mínimo y máximo en un **mes**

RF9. Para todas las estaciones dadas de alta, el Sistema mostrará las mediciones en forma de tabla. La **tabla de observaciones** de una estación se podrá obtener para una fecha concreta y representará todas las observaciones del día solicitado, estén o no presentes en el Sistema (en caso de que no estén se mostrará explícitamente el hueco). Cada fila de la tabla representará el período de tiempo observado, típicamente diezminutal, y las columnas representarán los valores de cada una de las variables configuradas. Además del valor, se representará visualmente el resultado del control de calidad (mediante un código de colores) y será posible ver los valores umbrales aplicados como referencia.

RF10. Para todas las estaciones dadas de alta, el Sistema mostrará las mediciones en forma de gráficas. Cada variable tendrá asociado un tipo de **gráfica** de entre las siguientes: **línea, barras, área o ninguna**. Las gráficas podrán obtenerse para una fecha concreta y se representará una gráfica por cada una de las variables observadas ese día. El eje X de la gráfica será el del tiempo mientras el eje Y será el de los valores. Cada punto dentro de la gráfica será una observación N-minutal, cuyo detalle (valor y fecha de observación) podrá ser

obtenido pulsando en dicho punto, acción tras la cual se obtendrá un tooltip con el detalle.

RF11. El Sistema mostrará sobre el **mapa de Google Maps** todas las estaciones dadas de alta en el mismo. Inicialmente estará centrado en la estación propia del usuario conectado, que será además marcada de forma especial. Las estaciones adyacentes que queden dentro del área del mapa serán también mostradas con una marca diferente. El mapa dispondrá de las funcionalidades normales de Google Maps para desplazarse y realizar zoom+ y zoom-.

RF12. Pulsando sobre cada estación en el mapa se mostrará un **tooltip** con sus datos básicos y las últimas observaciones obtenidas en la misma, en caso de que existan.

RF13. El usuario podrá realizar **búsquedas de estaciones** mediante los datos administrativos (**dirección postal**) o geolocalizados (**latitud y longitud más un radio**). Dicha búsqueda podrá realizarse desde el mapa de estaciones o desde la representación de observaciones. Desde el listado resultante de estaciones se podrán consultar los datos más básicos de cada una y además será posible:

a. En caso de haber sido realizada la búsqueda desde la representación de observaciones, cambiar la representación para reflejar específicamente los datos de la estación seleccionada.
b. En caso de haber sido realizada la búsqueda desde el mapa de estaciones, centrar el mapa en la estación y marcarla de forma especial. Las estaciones próximas serán representadas y marcadas de forma diferenciada.

RF14. El Sistema podrá **compartir en Twitter mensajes (comentarios)** conteniendo las condiciones calculadas a partir de datos medidos por la estación del usuario. Los comentarios se compondrán para aquellas observaciones medias, mínimas y máximas que estén calculadas sobre el 100% de su base. Por ejemplo, si la temperatura media, mínima y máxima para el día 21/05/2015 para una estación de lectura diezminutal ha sido calculada sobre 144 valores sin errores de calidad, se generará un comentario que contenga los 3 valores, el nombre de la variable y el período observado. Los comentarios de

carácter social se realizarán para derivadas de carácter diario y de carácter mensual.

RF15. El usuario podrá caracterizar la estación asociada, completando los siguientes datos:

a. El **modelo** de la estación, a elegir de entre los registrados en el Sistema
b. Los **parámetros** declarados por el plugin seleccionado, que típicamente contendrán valores como direcciones IP, puertos TCP/UDP, usuarios, contraseñas, rutas, nombres de ficheros, etc.
c. Para cada variable declarada por el plugin seleccionado, se podrán introducir los valores mínimo y máximo del **umbral** que se ha de considerar en el control de calidad.
d. Los **metadatos** propiamente dichos, al menos la dirección postal y la latitud, longitud y altitud donde se ubica.

RF16. El usuario podrá configurar su propio **perfil de usuario**, incluyendo no sólo los **datos básicos** como nombre de usuario, contraseña, datos de contacto, sino además los datos de **conexión al perfil de social** en Twitter.

REQUISITOS NO FUNCIONALES

RNF1. La inclusión de nuevos formatos de datos y protocolos de comunicaciones será sencilla, con un enfoque tipo "**plugins**".

RNF2. El Sistema de almacenamiento de observaciones debe ser **flexible** respecto a las observaciones que puede producir cada estación, y será **adaptable** a nuevas observaciones.

RNF3. El Sistema de almacenamiento garantizará la **escalabilidad** del Sistema respecto al posible crecimiento del tamaño del mismo, asegurando el buen comportamiento en tales circunstancias tanto del almacenamiento como de la lectura de las observaciones y estaciones.

CASOS DE USO

A continuación se presenta una representación de casos de uso de los requisitos funcionales formalizados anteriormente:

AUTENTICACIÓN

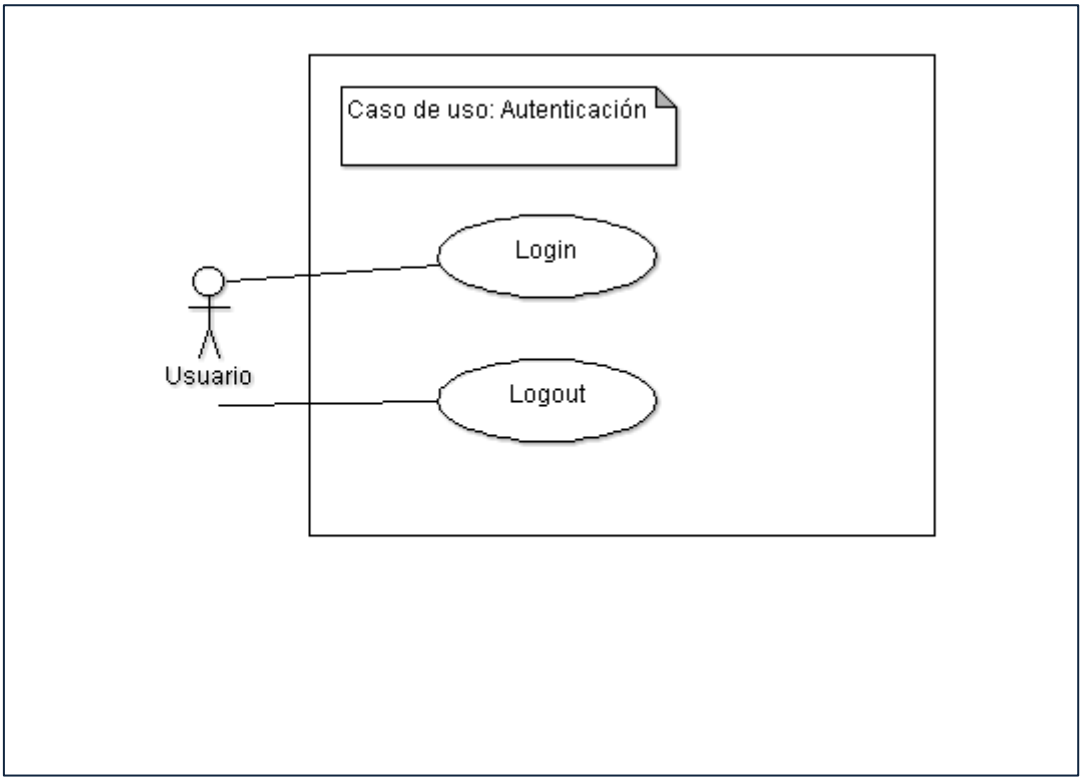

Ilustración 6 - Caso de uso: autenticación

Caso de uso:	**Login**
Código:	**CU01**
Descripción:	El usuario obtiene un formulario donde puede introducir usuario y contraseña
Actor	Todos
Flujo Normal:	

▪ El usuario proporciona sus datos de conexión de manera correcta y accede a la pantalla principal del Sistema: la tabla de observaciones de la estación asociada
Flujo Alternativo:
▪ El usuario introduce un par incorrecto y obtiene un mensaje de error indicando el fallo de autenticación. El usuario puede intentarlo de nuevo

Caso de uso:	**Logout**
Código:	**CU02**
Descripción:	El usuario termina la sesión que tiene abierta con el servidor
Actor	Usuario autenticado
Flujo Normal:	
▪ Desde cualquier punto de la aplicación, el usuario pulsa en la acción de desconectar y se le muestra de nuevo la pantalla de login. Si introduce manualmente cualquier URL del Sistema, se le redirige a dicha pantalla.	

CONSULTA DE OBSERVACIONES

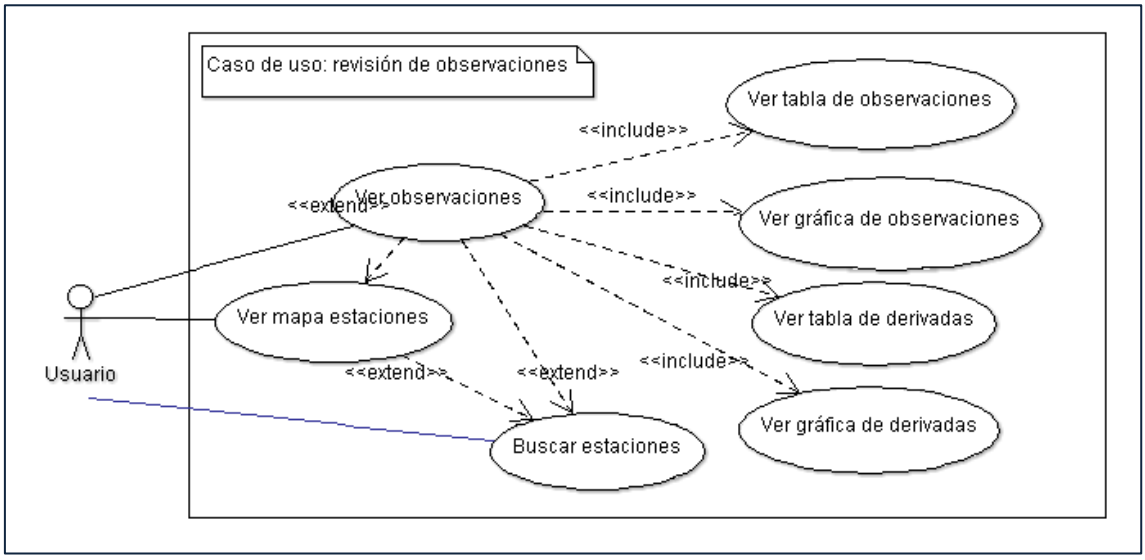

Ilustración 7 - Caso de uso: consulta de observaciones

Caso de uso:	**Ver observaciones**
Código:	**CU03**
Descripción:	El usuario obtiene una representación de los datos de una determinada estación (por defecto, la propia), acorde a los filtros de búsqueda (por fecha) aplicados. Nótese que el caso de uso incluye otros dos casos de uso que se encargan de describir los procesos de representación en sí mismos.
Actor	Usuarios autenticados
Flujo Normal:	
▪ El usuario accede al listado de observaciones de la estación propia en las últimas 24 horas. ▪ El usuario selecciona un filtro de búsqueda de observaciones para una fecha y/o un conjunto de variables a representar, actualizándose la representación respecto a dicho filtro.	

Flujo Alternativo:
▪ Desde la búsqueda de estaciones, el usuario selecciona una estación y obtiene el listado de observaciones de la misma en las últimas 24 horas.

Caso de uso:	**Ver tabla de observaciones**
Código:	**CU04**
Descripción:	Una vez aplicada una determinada selección de estación y filtros, el Sistema muestra una tabla de observaciones donde las columnas representan las diferentes observaciones y sus valores de calidad, y las filas representan los diferentes períodos de ingreso (es decir, en cada fila se mostrarán todas las observaciones y calidades obtenidas en un determinado instante de tiempo). Observaciones de calidad sospechosa serán destacadas visualmente.
Actor	El caso de uso está incluido en el CU “Ver Observaciones” y por tanto el actor corresponde a aquél.

Caso de uso:	**Ver gráfica de observaciones**
Código:	**CU05**
Descripción:	Una vez aplicada una determinada selección de estación y filtros, el Sistema representa una gráfica para cada una de las variables obtenidas, donde el eje X representará el tiempo y el eje Y los valores recibidos. Pasando el ratón sobre los puntos de la gráfica, se mostrará el valor puntual en cada instante.

Actor	El caso de uso está incluido en el CU “Ver Observaciones” y por tanto el actor corresponde a aquél.

Caso de uso:	**Ver tabla de observaciones derivadas**
Código:	**CU06**
Descripción:	Una vez aplicada una determinada selección de estación y filtros, el Sistema muestra una tabla de observaciones derivadas para cada uno de los períodos de derivación considerados. Cada período de derivación será destacado y el tiempo concreto de cálculo descrito claramente. Se indicará el porcentaje de progreso del cálculo sobre el total teórico y se destacarán visualmente aquellos cálculos en los que se evitó el uso de observaciones con calidad sospechosa. Para cada período, una fila representará una variable concreta y una columna cada uno de los valores mínimo, medio y máximo en el período.
Actor	El caso de uso está incluido en el CU “Ver Observaciones” y por tanto el actor corresponde a aquél.

Caso de uso:	**Ver gráfica de observaciones derivadas**
Código:	**CU07**

Descripción:	Una vez aplicada una determinada selección de estación y filtros, el Sistema representa un bloque por cada período de derivación. En dicho bloque se mostrará una gráfica para cada variable que contendrá sus tres valores mínimo, medio y máximo. El tiempo abarcado por el gráfico será la unidad temporal superior al propio período, es decir, para períodos mensuales se mostrarán gráficas que abarquen los 12 meses del año, mientras para períodos del día se mostrarán gráficas que abarquen todos los días del mes al que pertenecen. Pasando el ratón sobre los puntos de la gráfica, se mostrará el valor puntual en cada instante.
Actor	El caso de uso está incluido en el CU "Ver Observaciones" y por tanto el actor corresponde a aquél.

Caso de uso:	**Ver mapa estaciones**
Código:	**CU08**
Descripción:	Se presenta sobre Google Maps la localización de las diferentes estaciones vinculadas con el Sistema. Cada estación representada dispondrá de un tooltip donde se ampliará la información de la misma: metadatos, geoposicionamiento y últimas observaciones recibidas acompañadas de la calidad de las mismas.
Actor	Usuarios autenticados
Flujo Normal:	

▪ El usuario selecciona la vista de mapa y el Sistema muestra Google Maps con la estación del propio usuario centrada en el visor. ▪ El usuario puede realizar las funciones de zoom y desplazamiento clásicas de Google Maps. ▪ El Sistema mostrará en todo momento con un icono las estaciones conocidas dentro de la región de mapa representada.
Flujo Alternativo:
▪ Desde la búsqueda de estaciones, el usuario selecciona una estación y decide geolocalizarla en el mapa. La vista de Google Maps aparece en este caso centrada en la estación seleccionada.

Caso de uso:	**Buscar estaciones**
Código:	**CU09**
Descripción:	Es posible listar todas las estaciones del Sistema, filtrando por ciudades, provincias o países.
Actor	Usuarios autenticados
Flujo Normal:	
▪ El usuario selecciona la búsqueda de estaciones y realiza una búsqueda con o sin país, provincia, ciudad. El Sistema mostrará una tabla de estaciones indicando su ubicación y la fecha de la última observación recibida. Pulsando sobre las estaciones resultantes se podrá acceder al detalle de observaciones o se podrá situar sobre el mapa, según el contexto de la propia búsqueda	

NÚCLEO TEMPORIZADO

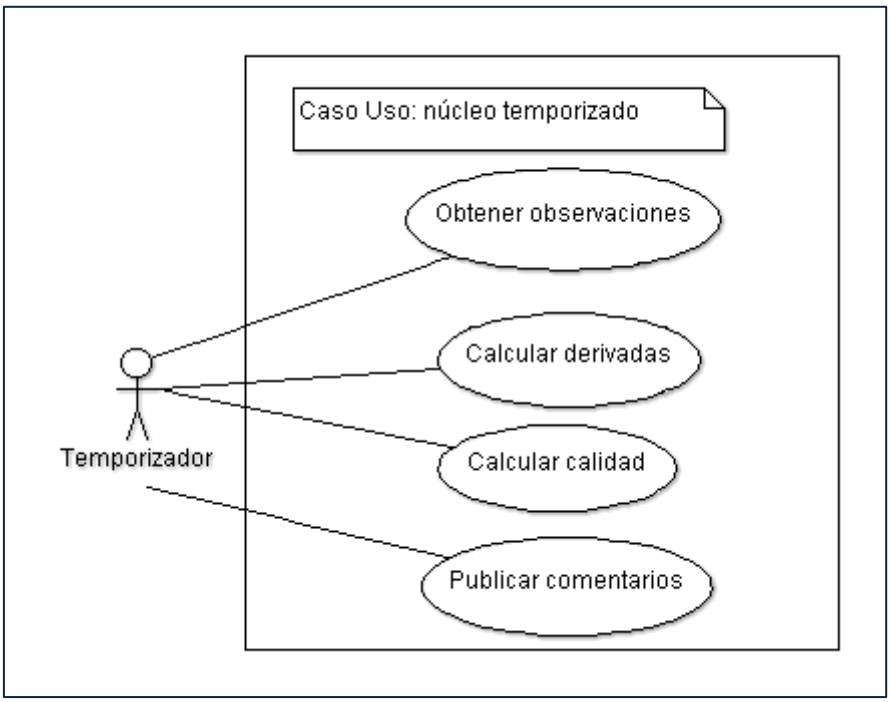

Ilustración 8 - Caso de uso: núcleo temporizado

Los casos de uso aquí descritos son disparados por procesos temporizados (planificador) del Sistema sin interacción del usuario final. Por tanto, el actor representado es el propio temporizador.

Caso de uso:	**Obtener observaciones**
Código:	**CU10**
Descripción:	Periódicamente (según la configuración temporal almacenada para cada estación dada el alta) el Sistema se conectará con la estación para obtener las observaciones generadas por la misma y almacenará la información en el Sistema local para su posterior presentación/tratamiento.
Actor	Temporizador (automático)

Flujo Normal:
▪ El temporizador se dispara en el momento configurado (periódicamente) ▪ El Sistema se conecta con la estación utilizando el protocolo de comunicaciones establecido para el modelo de estación ▪ El Sistema obtiene las observaciones utilizando el protocolo de parseo establecido para el modelo de estación ▪ Las observaciones son almacenadas en el Sistema.

Caso de uso:	**Calcular calidad**
Código:	**CU11**
Descripción:	Periódicamente (según configuración temporal) el Sistema inspeccionará las observaciones obtenidas que aún no han sido controladas para ejecutar los controles de calidad sobre las mismas y almacenar el resultado
Actor	Temporizador (automático)
Flujo Normal:	
▪ El temporizador se dispara en el momento configurado (periódicamente) ▪ El Sistema comprueba si existen observaciones aún no controladas en términos de calidad. ▪ El Sistema comprueba la calidad de las observaciones mediante los umbrales configurados y marca las observaciones como correctas o sospechosas. ▪ Las observaciones, sus etiquetas de calidad, y los umbrales utilizados para calcular dicha calidad, son almacenados en el Sistema.	
Flujo Alternativo:	

▪ El temporizador se dispara en el momento configurado (periódicamente) pero no encuentra observaciones con la calidad pendiente. Por tanto, no realiza ninguna acción y duerme hasta el siguiente momento configurado.

Caso de uso:	**Calcular derivadas**
Código:	**CU12**
Descripción:	Periódicamente (según configuración temporal) el Sistema inspeccionará las observaciones obtenidas para cada estación dada de alta y revisará los cálculos correspondientes a los valores máximos, medios y mínimos de cada observación para los períodos de tiempo contemplados, es decir, horario, período del día (mañana, tarde, noche), día y mes.
Actor	Temporizador (automático)
Flujo Normal:	
▪ El temporizador se dispara en el momento configurado (periódicamente) ▪ El Sistema comprueba si existen observaciones aún no contempladas en los cálculos derivados. ▪ Si es así, se recalculan los valores máximos, medios y mínimos para los períodos horarios, período de día, día y mes afectados por las nuevas observaciones.	
Flujo Alternativo:	
▪ El temporizador se dispara en el momento configurado (periódicamente) pero no encuentra nuevas observaciones (no utilizadas previamente). Por tanto, no realiza ninguna acción y duerme hasta el siguiente momento configurado.	

Caso de uso:	**Publicar comentarios**
Código:	**CU13**
Descripción:	Periódicamente (según configuración) el Sistema genera comentarios de texto a partir de las observaciones medias, máximas y mínimas según configuración y los difunde a través de la cuenta de Twitter vinculada por el usuario.
Actor	Temporizador (automático)
Flujo Normal:	
▪ El temporizador se dispara en el momento configurado (periódicamente) ▪ El Sistema comprueba si existen comentarios de texto pendientes de generar desde la última generación, así como si existen observaciones más actuales que permitan la composición de nuevos comentarios ▪ El Sistema genera los nuevos comentarios y los publica a través de la cuenta de Twitter del perfil del usuario.	
Flujo Alternativo:	
▪ El temporizador se dispara y encuentra que la generación de comentarios está deshabilitada, o no existen observaciones más modernas que los últimos comentarios, y por tanto no emprende ninguna acción hasta el siguiente momento configurado.	

CONFIGURACIÓN DEL SISTEMA

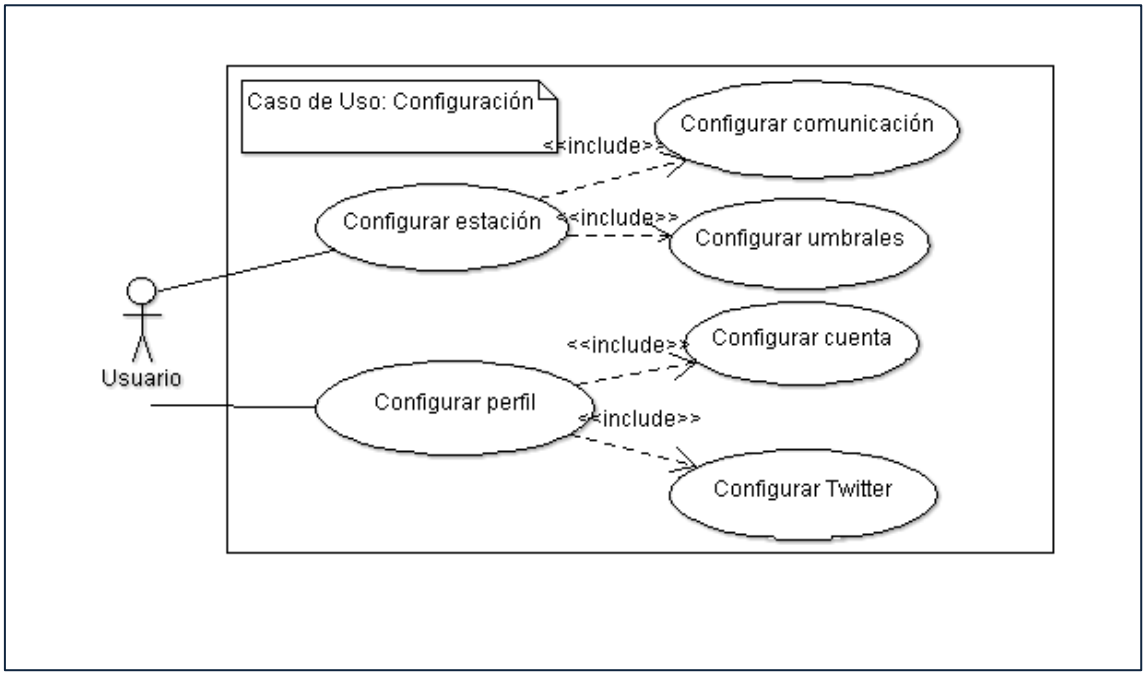

Ilustración 9 - Caso de uso: configuración del Sistema

Caso de uso:	**Configurar estación (incluye Configurar comunicación y Configurar umbrales)**
Código:	**CU14**
Descripción:	Permite al usuario configurar la estación vinculada, en concreto: • Modelo de la estación (determina también el tipo de comunicación, el formato de datos y la lista de observaciones a recuperar) • Período de observación de la estación • Valores umbrales (máximo y/o mínimo o ninguno) para cada una de las observaciones a recuperar.
Actor	Usuarios autenticados
Flujo Normal:	

▪ El usuario selecciona de entre la lista de modelos de estación conocidos, el que corresponde con su estación. ▪ Según el modelo, el Sistema presenta parámetros de comunicación/formato que son específicos de dicho modelo (obtenidos dinámicamente del *plugin* que implementa cada estación) ▪ Según el modelo, el Sistema lista todas las variables conocidas por la estación y permite al usuario introducir los valores máximo y mínimo para cada una de ellas.

Caso de uso:	**Configurar perfil social (incluye Configurar cuenta y Configurar Twitter)**
Código:	**CU15**
Descripción:	Permite al usuario configurar su perfil y la difusión de mensajes según los siguientes parámetros: • Datos personales • Usuario de Twitter para la difusión (si no se proporciona, se deshabilita la publicación de comentarios)
Actor	Usuarios autenticados
Flujo Normal:	
▪ El usuario accede a la configuración de perfil y puede modificar sus datos personales así como su usuario de Twitter	
Flujo Alternativo:	
▪ El usuario deshabilita la difusión eliminando los datos de conexión a Twitter	

SUBSISTEMAS DE ANÁLISIS

Del catálogo de requisitos descrito anteriormente, se puede desprender la existencia de tres elementos diferenciados, que en el diseño posiblemente se convertirán en paquetes de trabajo diferentes (con elementos comunes) y que en esta fase de análisis se describen mediante subsistemas de análisis que a su vez se descompondrán en funciones concretas. Dentro de dichas funciones estarán organizados todos los requisitos del análisis de manera que su transformación en paquetes de trabajo sea más sencilla (si bien no directa).

Los subsistemas identificados son:

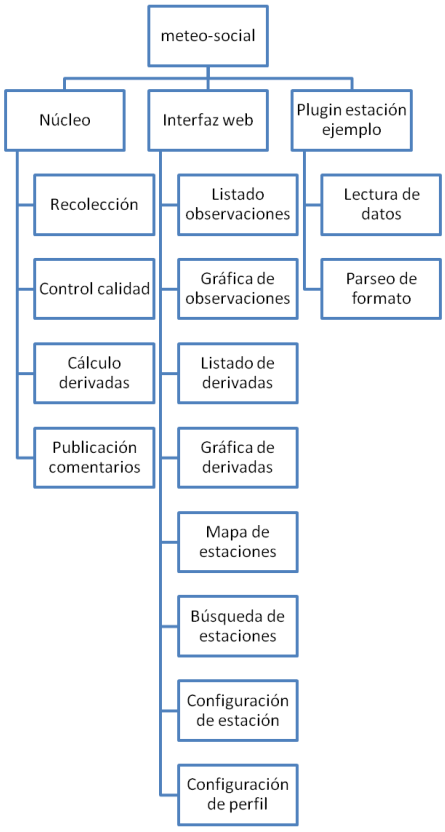

Ilustración 10 - Diagrama de subsistemas

SUBSISTEMA: NÚCLEO DE RECOLECCIÓN Y DIFUSIÓN

FUNCIÓN: RECOLECCIÓN

- **Temporización** de recolección
- **Interfaz de plugin** de recolección y parseo
- **Recuperación** de observaciones no ingresadas
- **Almacenamiento** de observaciones en base de datos

FUNCIÓN: CONTROL DE CALIDAD

- **Temporización** de la tarea de control
- **Recuperación** de observaciones con la calidad no controlada
- **Aplicación de límites** por variable medida
- Fijar **marcas de calidad** para cada observación

FUNCIÓN: CÁLCULO DE DERIVADAS

- **Temporización** del cálculo de derivadas
- **Identificación** de observaciones que no han sido utilizadas para base de cálculo de derivadas.
- **Búsqueda de observaciones** que, junto a las identificadas, deben **usarse para el cálculo** de cada uno de los períodos de derivación
- **Ejecución de los cálculos** afectados, actualización de los mismos en la base de datos y marca de observaciones iniciales como ya utilizadas para la derivación.

FUNCIÓN: DIFUSIÓN

- **Temporización** de la generación y difusión de comentarios
- **Obtención** de observaciones derivadas que aún no han sido utilizadas para la generación de comentarios y que están completas, es decir, se han calculado sobre el 100% de la base posible.
- **Generación de los comentarios** en lenguaje natural que agrupan las medias, mínimas y máximas por variable y período de derivación
- Conexión con cuenta de **Twitter** y envío de comentarios generados

- **Actualización** de las observaciones utilizadas para la generación de comentarios, de forma que no sean utilizadas de nuevo.

SUBSISTEMA: INTERFAZ WEB

FUNCIÓN: LISTADOS DE OBSERVACIONES

- Interfaz de **filtrado** de observaciones
- **Recuperación** de lotes de observaciones aplicando filtros con cobertura completa del período utilizado para la búsqueda, incluyendo valores vacíos cuando éstos no existan.
- **Presentación tabular** de lotes de observaciones, marcando los eventuales problemas de calidad y mostrando los valores umbrales utilizados para dicho control.

FUNCIÓN: GRÁFICAS DE OBSERVACIONES

- Interfaz de **filtrado** de observaciones
- **Recuperación** de lotes de observaciones aplicando filtros
- Identificación del **tipo de gráfica** a utilizar para cada una de las variables recuperadas.
- **Presentación en gráficas** interactivas del tipo adecuado de los lotes de observaciones recuperados.

FUNCIÓN: LISTADOS DE DERIVADAS

- Interfaz de **filtrado** de observaciones
- **Recuperación** de observaciones derivadas aplicando los filtros con cobertura completa del período utilizado para la búsqueda, incluyendo valores vacíos cuando éstos no existan.
- Cálculo de porcentajes de **avance del cálculo**

- **Presentación** en bloques según el período de derivación de las observaciones recuperadas, marcando visualmente aquellas que, debido a errores de calidad, han ignorado alguna observación que era susceptible de ser base para dicho cálculo.

FUNCIÓN: GRÁFICAS DE OBSERVACIONES DERIVADAS

- Interfaz de **filtrado** de observaciones
- **Recuperación de observaciones** derivadas aplicando los filtros
- Cálculo de porcentajes de **avance del cálculo**
- Identificación del **tipo de gráfica** a utilizar para cada una de las variables recuperadas.
- **Presentación en gráficas** interactivas del tipo adecuado de los lotes de observaciones recuperados. Cada gráfica agrupará los valores mínimos, medios y máximos. Las gráficas serán a su vez agrupadas según los períodos de derivación.

FUNCIÓN: MAPA DE ESTACIONES

- Geolocalización de estaciones en **Google Maps** con funciones de navegación y zoom
- Diálogo de **información básica** de cada estación incluyendo el último bloque de observaciones obtenidas.
- Marca especial de la **estación asociada** del usuario
- Marca especial de la **estación seleccionada** en la búsqueda

FUNCIÓN: BÚSQUEDA DE ESTACIONES

- Interfaz de **filtrado** de estaciones por criterios administrativos o por criterios geográficos
- **Recuperación y presentación** dinámica y paginada de estaciones acordes con los filtros proporcionados.

- **Selección** de estaciones para su utilización en los dos ámbitos donde se integra la búsqueda: la representación de observaciones y el mapa de estaciones.

FUNCIÓN: CONFIGURACIÓN DE ESTACIÓN

- Selección de **modelo** de estación
- Configuración dinámica de los **parámetros** acordes al modelo de estación
- Configuración de **valores umbrales** para las variables declaradas por el modelo de estación.

FUNCIÓN: CONFIGURACIÓN DE PERFIL DE USUARIO

- Configuración de los **datos personales**
- Configuración de los **datos de conexión** a Twitter

SUBSISTEMA: PLUGIN DE ESTACIÓN DE EJEMPLO

LECTURA DE DATOS

- Implementación de un sencillo **protocolo de comunicación con la estación**, en este caso simplemente localizar y encontrar un fichero CSV local. En un escenario real, este tipo de integración podría representar la recepción de ficheros FTP enviados activamente por la propia estación, lo cual es factible en el mercado.

PARSEO DE FORMATO

- Implementación de un sencillo **protocolo de parseo de observaciones leídas** de la estación, en este caso el parseo de un CSV donde cada línea se identifica con un período de tiempo y cada columna contiene el valor de una de las variables observadas en el período, que aquí se define como diezminutal. También es un tipo de formato que es factible encontrar en el mercado.

DISEÑO DEL SISTEMA

Se procura construir un **diseño altamente modular** que facilite el mantenimiento posterior del código y mantenga lo más bajo posible el esfuerzo de ampliación de nuevas funcionalidades. Es preciso notar que el entorno tecnológico escogido tiene un impacto importante en el diseño de componentes asociados a la navegación web y a la comunicación entre el interfaz de usuario y el propio backend. Aquí se expone el resultado de dicho diseño de forma tecnológicamente neutral, pero **puede ser interesante revisar** primero la descripción del entorno tecnológico para ganar una mejor comprensión sobre el propio diseño.

DISEÑO DE LA PÁGINA WEB

Otro elemento que impacta en el diseño (específicamente de las partes dedicadas a gestionar el interfaz de usuario) es la especificación de construcción de las páginas web del Sistema. Éstas estarán compuestas por diferentes elementos cuyo comportamiento será independiente y que se adaptará a la funcionalidad específica que se muestre en cada momento.

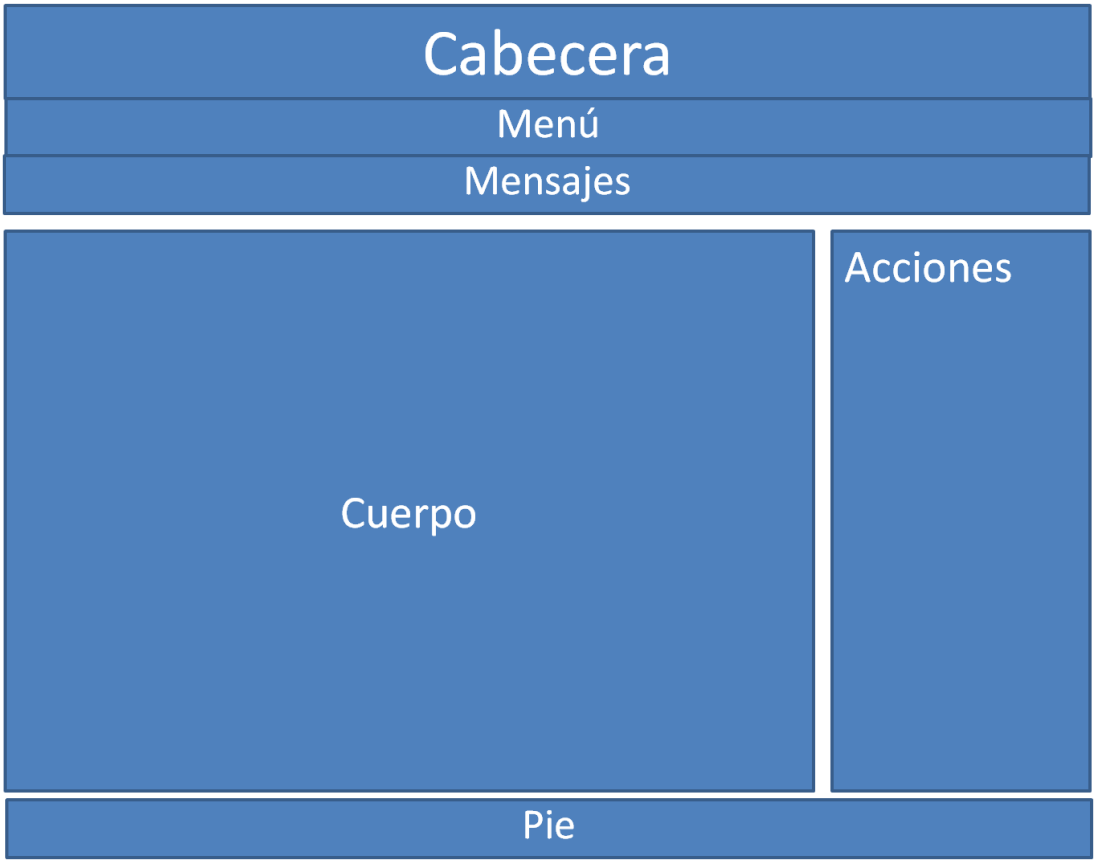

Ilustración 11 - Estructura de la página web

Todas las páginas contarán con los elementos mostrados en la figura, a saber:

CABECERA:

Un elemento relativamente estático cuyo contenido no se adaptará a la navegación puesto que su información no es relativa a la misma. Básicamente mostrará información de la aplicación (**logo y nombre**) **y del usuario conectado** (nombre completo). Sería el elemento adecuado también para mostrar diferentes banderas para cambiar el idioma del interfaz de usuario, en caso de completarse la internacionalización del mismo.

MENÚ

Las opciones de presentación de información se encuentran mayoritariamente accesibles a través de acciones, pero se mantiene este espacio para colocar un menú

especialmente para las **opciones de configuración**: de estación y de perfil del usuario. Sería el elemento adecuado para ampliar dicho menú con nuevas opciones.

MENSAJES

Se trata de un elemento que proporciona un espacio destinado a presentar **mensajes de error, validación**, etc., al usuario. Los mensajes han de estar adecuadamente clasificados mediante códigos de colores, iconos, etc.

CUERPO

El elemento **principal** y de mayor espacio se destina a la presentación de la información asociada a la funcionalidad actual, es decir, será el recipiente de las tablas de observaciones, listados gráficos, mapas de estaciones, formularios de configuración, etc.

ACCIONES

Asociado a cada elemento funcional cuya información principal se muestra en el cuerpo, habrá siempre una serie de **acciones disponibles** sobre el mismo o destinadas a ampliar la información. Por ejemplo, cuando en el espacio principal se muestra la tabla de valores observados, una acción podrá ser cambiar la vista para ver las gráficas de dichos valores, o dirigirse al mapa de estaciones.

PIE

Un elemento, como la cabecera, relativamente estático, en este caso dedicado a mostrar un **mensaje simple**. Sería el elemento adecuado para incluir enlaces externos a organismos institucionales (WMO, Agencias de Meteorología), fabricantes de estaciones y sensores, estudios científicos sobre la materia, etc.

VISTA GLOBAL DE COMPONENTES

Si bien la aplicación será **completamente web**, ésta será diseñada como un Sistema **cliente-servidor**, donde el cliente será en efecto ejecutado dentro del navegador del

usuario y el servidor será soportado por un servidor de aplicaciones. La tecnología seleccionada permite esta visión y este tipo de despliegue sin que sea necesaria la construcción de un auténtico cliente, ni por supuesto su despliegue e instalación en las máquinas locales del usuario, para el que la aplicación se muestra como una web 2.0 normal.

Además de la división principal entre elementos cliente y elementos servidor, se identifican una serie de subcomponentes que colaboran entre sí y que se describen en la siguiente figura:

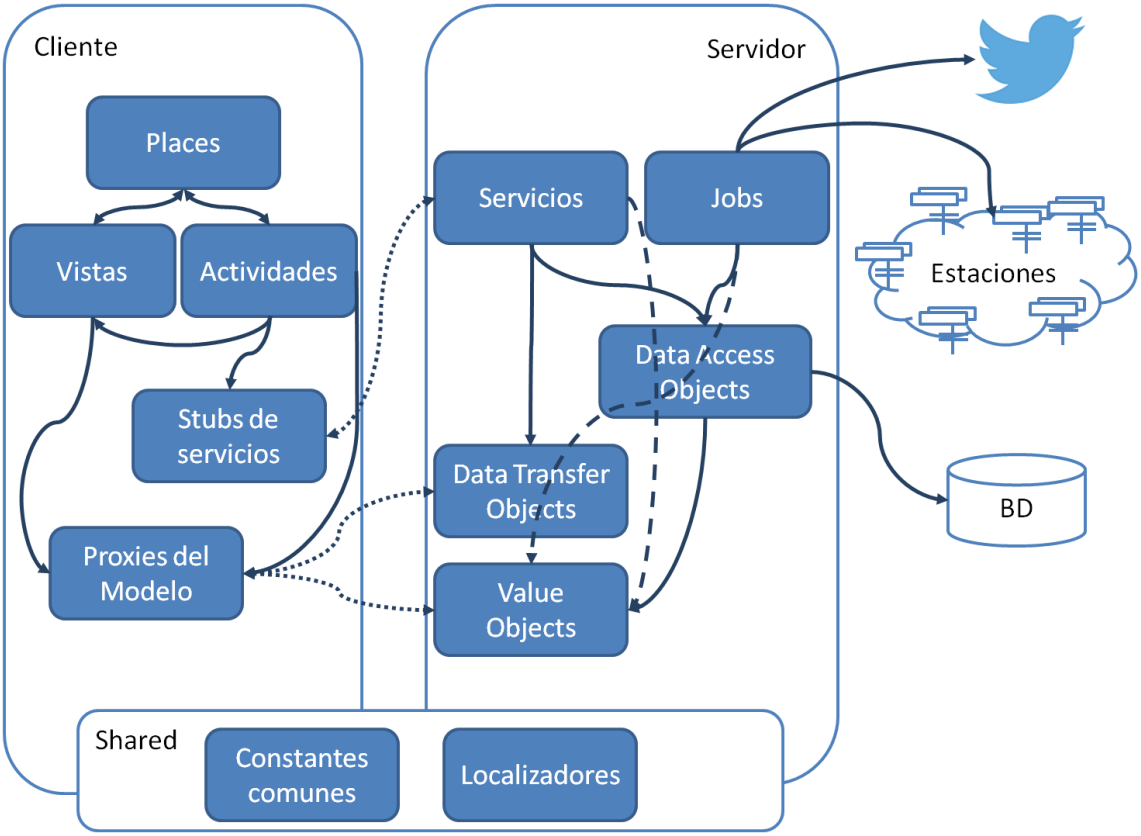

Ilustración 12 - Vista global de componentes

COMPONENTES DE SERVIDOR

VALUE OBJECTS

Las **entidades que componen el modelo de datos** del Sistema serán unos objetos planos con métodos getters y setters que únicamente proporcionan acceso al

contenido de la base de datos relacional del Sistema. Carecen de cualquier tipo de lógica adicional, incluida la propia de localizarse, actualizarse y eliminarse en la base de datos, lógica ésta que corresponde a la propia capa de persistencia representada por los Data Access Objects.

Los VOs del Sistema serán los siguientes:

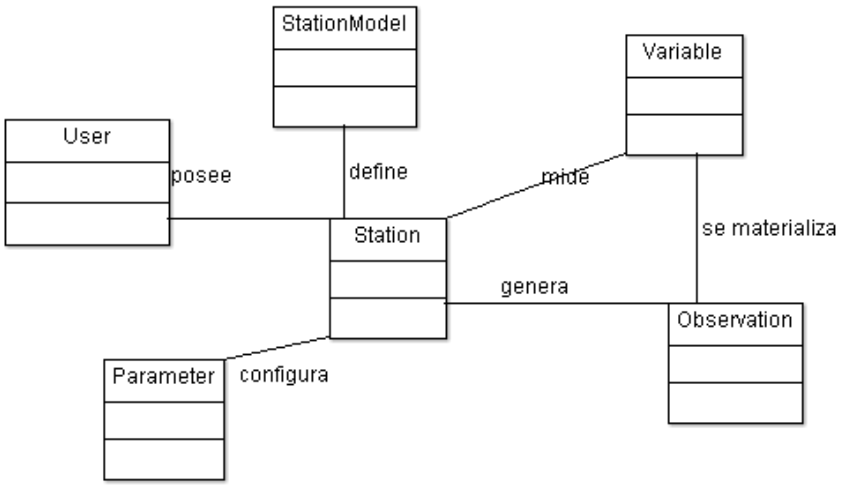

Ilustración 13 - VOs del Sistema

- **StationModel**: contiene los modelos de estación conocidos y dados de alta en el Sistema, de entre los que el usuario podrá seleccionar el correspondiente a su estación
- **User**: contiene todos los usuarios registrados en el Sistema
- **Station**: contiene todas las estaciones registradas en el Sistema, asociadas con un usuario y pertenecientes a un modelo de estación
- **Variable**: las variables físicas que son medidas por las estaciones existentes, relacionadas siempre con las mismas.
- **Parameter**: los parámetros configurados para cada estación
- **Observation**: cada instancia de observación ingresada en el Sistema, correspondiente a una variable y proveniento de una estación.

El mapeo entre las entidades y propiedades del Sistema, y las tablas, columnas y relaciones de la base de datos será declarativo.

DATA ACCESS OBJECTS

La capa DAO de persistencia se encarga de las **operaciones clásicas de lectura, inserción, modificación y borrado de los VO** en la base de datos relacional.

Para cada una de las entidades (VO), se construirá una interfaz que expone las operaciones disponibles sobre la misma y una clase que implementa dicha interfaz (y por lo tanto las operaciones expuestas). Puesto que todas las entidades tendrán al menos un conjunto básico de operaciones a realizar sobre las mismas, se construirá una interfaz y una clase abstracta que exponga/implemente dichas operaciones comunes, y será responsabilidad de cada DAO final proveer operaciones específicas del VO sobre el que trabaja.

El control transaccional y la comunicación con la propia base de datos serán también declarativos.

DATA TRANSFER OBJECTS

Las entidades del modelo (VO) deben poder ser intercambiadas entre el cliente y el servidor de manera que el cliente represente/complemente la información de las mismas. Dichos VOs, como se refiere más adelante, serán codificados/decodificados automáticamente para su intercambio a través de la conexión HTTP estándar. Sin embargo, no siempre en el cliente se necesitarán entidades simples o colecciones de entidades simples, sino que en ocasiones será necesario la creación de objetos que pueden considerarse también del modelo, y de hecho siguen el patrón VO en el sentido de ser colecciones de propiedades y métodos getter/setter, pero que **agrupan información o cálculos** que no se encuentran necesariamente en las entidades persistentes del Sistema. Por ejemplo, cuando se representa la tabla de observaciones es más adecuado para el cliente disponer de un objeto que agrupe las observaciones por período y complete siempre todas las observaciones aunque no existan en la base de datos (con valores nulos), de manera que la lógica del cliente sea lo más sencilla

posible y cualquier transformación más o menos compleja quede localizada en el servidor. A este efecto se crean los DTOs, que están destinados a su envío/recepción a través de HTTP entre cliente y servidor. Para el cliente, dichos DTOs son indistinguibles de los VOs del propio modelo, y son en definitiva parte del modelo igualmente.

TAREAS

El Sistema requiere de la intervención de una serie de tareas periódicas (jobs), para llevar a cabo las diferentes actividades del núcleo de recolección y publicación del mismo:

- **Recolección**: debe inspeccionar periódicamente cada una de las estaciones dadas de alta en busca de nuevos datos disponibles para leerlos, interpretarlos y escribirlos en la base de datos.
- **Control de calidad**: esta tarea busca observaciones en la base de datos que aún no hayan sido controladas, las analiza respecto a los valores umbrales configurados para la estación asociada y almacena el dato de calidad junto con los umbrales utilizados para su cálculo (puesto que éstos pueden variar según configuración, es importante mantener siempre los utilizados en cada momento)
- **Cálculo de valores derivados**: encuentra valores que no se han utilizado aún para el cálculo de valores mínimos, medios y máximos, para volver a computar los mismos utilizando los valores previos más los recién recibidos.
- **Publicación de comentarios**: busca observaciones mínimas, medias y máximas que hayan sido calculadas de forma completa (100% de las observaciones de base para el cálculo se encontraban presentes y con una calidad adecuada), compone un mensaje de texto para las mismas y lo publica a través del perfil de Twitter del usuario al que se asocia la estación.

Las tareas y su periodicidad se especifican de forma declarativa.

SERVICIOS

Si bien la lógica de negocio del Sistema se puede localizar en las tareas de ingreso, control, derivación y publicación, se puede decir que la lógica de negocio orientada al intercambio de información con el usuario se encuentra en la capa de servicios. Se trata de elementos que permiten su **invocación remota desde el cliente**, con el cual intercambiarán tipos primitivos así como VOs y DTOs. Los servicios por lo tanto exponen una serie de operaciones que son invocables a través de HTTP. Inicialmente dichas operaciones se agrupan en dos servicios:

- De **observaciones**: provee/recoge información asociada con las observaciones de cara a su representación en el cliente, utilizando los filtros proporcionados por el mismo para producir la salida que el cliente debe representar.
- De **estación**: provee/recoge información asociada a las estaciones dadas de alta en el Sistema y a sus metadatos asociados.

Por convenio se decide que los Servicios en sí queden lo más descargados de lógica posible, que se limitará al intercambio de datos con el cliente, mientras se apoya principalmente en las clases de persistencia (DAOs). En caso de que sea necesaria una lógica adicional a la simple propuesta por los DAOs, como por ejemplo para codificar los VOs en DTOs, los Servicios se apoyarán en clases de soporte (Helpers), que estarán relacionadas 1:1 con cada Servicio.

COMPONENTES DE CLIENTE

PROXIES DEL MODELO

El cliente necesita disponer de la información representada por el modelo del Sistema (VOs y DTOs) para poderla presentar al usuario final, o bien para obtenerla del usuario y transmitirla al servidor de modo que éste realice operaciones de escritura sobre la misma (por ejemplo, para actualizar el perfil del usuario o la configuración de la estación). La codificación/decodificación de estos objetos a través de HTTP es realizada automáticamente por el API de RequestFactory de GWT, mediante JSON. Sin embargo, el cliente no usa directamente los objetos del servidor sino una **interfaz** que declara únicamente aquellas **propiedades** de cada entidad del modelo que son **accesibles**

desde el cliente. Estas interfaces son denominadas proxies y para el cliente son la única representación del modelo disponible.

STUBS DE SERVICIOS

Así como las entidades del modelo están mapeadas en el cliente por proxies, los servicios y sus **operaciones públicas son accesibles en el cliente** a través de interfaces que actúan como Stubs para la ejecución remota de dichas operaciones a través de HTTP. Los stubs en conjunto con los proxies del modelo centran toda la comunicación que existe entre el cliente y el servidor. La invocación remota es ejecutada de forma transparente por el API RequestFactory de GWT, que utiliza peticiones asíncronas de tipo AJAX.

PLACES

El diseño del cliente se ha basado en el patrón **MVP** (Model-View-Presenter), tal como se recomienda en GWT (http://www.gwtproject.org/doc/latest/DevGuideMvpActivitiesAndPlaces.html). Este patrón permite una separación de responsabilidades que mejora la claridad del diseño, de la implementación y facilita el soporte y evolución del mismo.

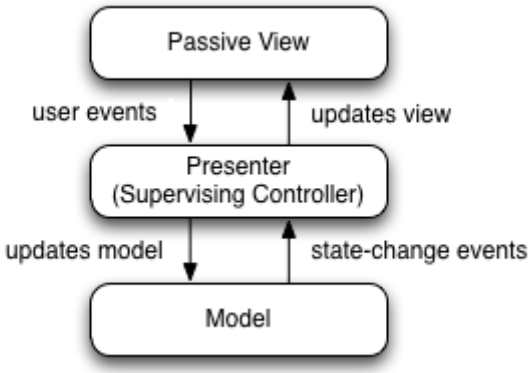

Ilustración 14 - Diagrama MVP

En el modelo MVP implantado, la M serían los proxies del modelo, la V serían las vistas y la P serían las actividades. No estrictamente parte de este patrón, pero sí

relacionados con el mismo y con la composición de las páginas web descritas al inicio del diseño, se introduce el concepto de Place.

Un Place inicialmente no es más que la representación de **un estado del interfaz web** que es navegable (pulsando en algún enlace o acción, las diferentes regiones del interfaz presentan la información específica de dicho estado, que puede ser marcado como favorito en el navegador o accesible a través del botón "Volver" del mismo). Normalmente se identificará con una **funcionalidad concreta**, y servirá para definir cómo ha de comportarse cada una de las regiones de la web en un estado concreto. **Cada región del interfaz constará de**:

- **Una vista**, que muestra/obtiene los datos para/del usuario
- **Una actividad,** que coordina la lógica de manipulación de dichos datos, las acciones del usuario y sus relaciones con el servidor.

Así pues, un determinado estado (por ejemplo, el "listado de observaciones tabulares") será un Place cuya definición incluirá las actividades y vistas de cada una de las 6 regiones del interfaz. Las regiones de cabecera, menú, mensajes y pie son comunes a todos los Places, es decir, se comportan exactamente igual independientemente de la funcionalidad proporcionada. Así pues, las regiones de cuerpo y acciones son las que definen una funcionalidad concreta o Place.

Los Places disponibles en el Sistema son:

- **ObservationListPlace**: presentará las tablas y gráficas de observaciones ingestadas así como de observaciones derivadas, según los parámetros que reciba (tipo del place). Cuando la interfaz se ubique en este Place, ocurrirá lo siguiente:
    - Se construirá una actividad de listado de observaciones (ObservacionListActivity) para la región del cuerpo y otra para la región de acciones (ObservationListActionsActivity).

  - La actividad ObservacionListActivity interpretará el tipo de Place (observaciones normales o derivadas, presentación tabular o gráfica) y construirá la vista adecuada (vista de tabla, de gráficas, etc.).
  - La actividad ObservacionListActionsActivity interpretará el tipo de Place (observaciones normales o derivadas, presentación tabular o gráfica) y construirá una vista que presente las acciones disponibles en cada caso (ir a la vista gráfica de la tabular y viceversa, etc.). Al pulsar en dichas acciones, la actividad construirá un nuevo Place e indicará al Sistema que ha de cargarlo, permitiendo así la navegación cuando las regiones se adapten al nuevo Place.
- **StationMapPlace**: presentará el mapa de estaciones en la región de cuerpo (para lo cual se construirá una vista y una actividad) y las acciones disponibles en la región de acciones (para lo cual se construirá una vista y una actividad)
- **StationSetupPlace**: presentará los datos de configuración de la estación en la región de cuerpo y permitirá guardarlos, previa validación, con acciones de la región de acciones. Se construyen vistas y acciones para cada una de las dos regiones.
- **UserSetupPlace**: presentará los datos de configuración del perfil de usuario en la región de cuerpo y permitirá guardarlos, previa validación, con acciones de la región de acciones. Se construyen vistas y acciones para cada una de las dos regiones.

VISTAS

La Vista es la encargada de **presentar la información al usuario final**, es decir, recibe una serie de datos que transforma en formularios, tablas, gráficas, mapas, etc., y es capaz de obtener datos del usuario que son legibles por la actividad. Son carentes de toda lógica y se limitan única y exclusivamente al intercambio de información con el usuario. Cualquier lógica de cliente estará incluida en las actividades, que gobiernan el comportamiento de las vistas.

Existirá una vista por cada una de las cuatro regiones comunes, que no variarán según el Place. Finalmente existirá una vista por cada Place (funcionalidad) y región dinámica (de cuerpo y de acciones). A su vez las vistas harán uso de otras clases igualmente sin lógica de ningún tipo que no sea de pura presentación, que se utilizarán como subcomponentes o widgets (formularios, diálogos, widgets de algún tipo, etc.).

ACTIVIDADES

Las actividades son las que **gobiernan la lógica de presentación del interfaz web**. Según el Place que se ubique en un momento dado, se cargará una u otra actividad, que será responsable de:

- Obtener los datos del servidor
- Pasar los datos a las vistas para su representación
- Obtener los datos de las vistas, validarlos y enviarlos al servidor para su almacenamiento.
- Reaccionar a las acciones del usuario (que pulsa en acciones, enlaces, botones) construyendo diálogos, manipulando componentes o cambiando el Place para que un nuevo conjunto de actividades se encargue de una funcionalidad diferente que el usuario ha seleccionado.

El comportamiento de las actividades es asíncrono, tanto en su relación con el servidor como con el usuario, de manera que la experiencia de usuario es rica y se asemeja a la de una aplicación de escritorio.

COMPONENTES COMPARTIDOS

Existe un último vínculo entre cliente y servidor que no se refiere al existente en tiempo de ejecución (invocaciones remotas, intercambio serializado de objetos), sino al existente en **tiempo de compilación**.

CONSTANTES COMUNES

El cliente y el servidor compartirán una serie de **constantes** que se declaran en una interfaz. Dicha interfaz es accesible tanto desde el cliente como desde el servidor de

manera que sea posible intercambiarse información como por ejemplo el tipo de filtro provisto por un usuario, formatos de fechas comunes, etc.

LOCALIZADORES

La "magia" de la serialización de entidades e invocación remota de RequestFactory necesita de unos elementos que "localicen" en el servidor las **entidades mapeadas en los proxies** del modelo y los **servicios mapeados en los stubs**. Para ello es preciso construir un localizador de entidades y un localizador de servicios, cumpliendo las interfaces definidas en RequestFactory.

ENTORNO TECNOLÓGICO

FRAMEWORKS Y LIBRERÍAS

El Sistema será construido fundamentalmente con tecnologías **J2EE** y el framework **GWT**. Al tratarse de una aplicación J2EE, podrá desplegarse dentro de un servidor de aplicaciones estándar de mercado para su ejecución web. En el caso de la versión inicial, objeto de este proyecto, se desplegará la aplicación en un servidor **Wildfly 8.1.0**. La aplicación se compilará con el **JDK 1.8.0_25**.

GWT 2.6.1 será utilizado como base para la construcción de un interfaz de usuario rico que simule un funcionamiento cliente-servidor, si bien en definitiva se trate de una única aplicación cuya parte de cliente (un conjunto de Javascripts autogenerados por el framework) se comunica con el servidor mediante peticiones AJAX y serialización de datos JSON. GWT permitirá además desarrollar el Sistema enteramente en Java y, sin necesidad de crear un solo Javascript o página web, el propio framework se encargará de "compilar" el código Java a código HTML+Javascript, que además tiene la propiedad de ser compatible con la mayoría de los navegadores del mercado, en sus diversas versiones y diferentes Sistemas Operativos.

Se aplicará asimismo el patrón de diseño de Inversión de Control, para lo cual se contará con dos librerías diferenciadas:

- **Spring 4.1.3** para la inyección de dependencias exclusivamente de servidor.
- **Guice+Gin 3.0** para la inyección de dependencias en las clases de cliente, ya que es la tecnología soportada por GWT.

La persistencia estará basada en **JPA 2.0 + Hibernate 3**, de manera que tanto el mapeo de entidades con tablas como en la implementación de las operaciones de lectura, modificación, inserción y borrado de las mismas se realice de manera declarativa y

siempre orientada a objetos. Para soportar asimismo transacciones declarativas, se utiliza Spring 4.1.3.

El mecanimos de autenticación es declarativo y se basa en **Spring Security 4.1.3**.

El propio motor de base de datos relacional será **MySQL 5.5**, y por lo tanto se utilizará su driver para Java MySQL/JConnector.

Para la representación de gráficas en GWT se utiliza **Google Visualization API**. Dicha librería se enlaza mediante AJAX de manera que siempre se descarga la última versión disponible en el cliente.

Para la representación geolocalizada se utiliza **Google Maps API**. Dicha librería se enlaza mediante AJAX de manera que siempre se descarga la última versión disponible en el cliente.

Para la integración con Twitter se utiliza **JTwitter 3.0.7** junto a **Signpost 1.2.1.1** para la comunicación vía Oauth.

HERRAMIENTAS DE SOPORTE

Además de las librerías y componentes de desarrollo y despliegue, se utilizarán para la ejecución del proyecto un número de herramientas de trabajo y soporte:

- **MS Word y MS PowerPoint** para la generación de documentación y presentaciones.
- **MS Excel** para la hoja de seguimiento de avance del proyecto.
- **ArgoUML** para el modelado UML.
- Como herramienta de desarrollo J2EE se utilizará **Eclipse Juno** Service Release 2.
- Como servidor de aplicaciones de depuración integrado en el IDE se utilizará **Jetty 9.2.3**.
- Como herramienta de compilación y empaquetado se utilizará **Apache Ant 1.6.1.**
- El repositorio de código fuente utilizado será **GitHub**.
- Se integrará el código del Sistema con una herramienta de análisis de calidad y errores llamada **SonarQube 5.0.1**.

PLANIFICACIÓN Y EJECUCIÓN

Como ya se ha comentado, la construcción del Sistema se plantea mediante una serie de sprints. Se definieron **3 iteraciones**, con las dos primeras enfocadas a producir el Sistema plenamente funcional mientras la última sumaba las funcionalidades de configuración. Para el seguimiento del avance el proyecto, se utilizó una hoja Excel de Scrum, cuya información se volcará aquí para ilustrar la evolución del desarrollo. El esfuerzo global se calculó en días naturales, estimando una disponibilidad media de 1,5 horas de trabajo real en cada día natural.

El plan de iteraciones concreto fue el siguiente:

Sprint	Start	Days	End	Size	Status	Goal
1	12/12/2014	42	22/01/2015	42	Released	Recolección y presentación básica
2	23/01/2015	48	11/03/2015	50	Released	Recolección y presentación avanzada
3	12/03/2015	22	02/04/2015	22	Released	Configuración y difusión

Ilustración 15 - Planificación de iteraciones

Y el backlog completo fue el siguiente:

Story ID	Story name	Status	Size	Sprint
1	Estructura básica el proyecto	Done	3	1
2	Modelo datos (común, local y central)	Done	3	1
3	Interfaz de plugins de recolección y parseo	Done	3	1
4	Temporización de recolección	Done	6	1
5	Protocolo de estación dummy	Done	6	1
6	Recuperación y almacenamiento de observaciones	Done	6	1
7	Búsqueda y representación tabular de observaciones	Done	9	1
8	Control de calidad por variable	Done	6	1
9	Cálculo y representación de variables derivadas	Done	9	2
10	Representación gráfica de observaciones	Done	12	2
11	Geolocalización en Google Maps	Done	15	2
12	Tooltip de estación en mapa	Done	6	2
13	Búsqueda de estaciones	Done	6	2
14	Revisión de Sprint 1 y 2	Done	2	2
15	Configuración de estación	Done	3	3
16	Integración con Twitter	Done	6	3
17	Generación de comentarios	Done	9	3
18	Revisión final	Done	4	3

Ilustración 16 – Backlog

Habría sido interesante poder presentar la información del *project burndown* y los gráficos de velocidad de desarrollo esperada y obtenida. Sin embargo, puesto que no se realizó un trabajo continuado (diario), no se actualizó en consecuencia la información de seguimiento. Se puede considerar en este caso que tanto el backlog general como el backlog de iteraciones ha servido para mantener un registro de avance y de tareas pendientes, pero en ningún caso para la estimación de velocidad de progreso ni, en consecuencia, de finalización de cada una de las iteraciones.

ITERACIÓN 1: RECOLECCIÓN Y PRESENTACIÓN BÁSICA (TABULAR)

El objetivo de la iteración 1 es construir:

- Los **elementos comunes** del proyecto como: modelo de datos, elementos de persistencia, clases básicas de interfaz de usuario, las interfaces de los plugins de recolección, etc.
- El **núcleo de recolección** casi completo, incluyendo:
  - Núcleo de lectura y almacenamiento de observaciones
  - Plugin de estación de ejemplo
  - Control de calidad
- Función básica de **consulta de observaciones en formato tabular**

El esfuerzo para completar la iteración se estimó en **42** días naturales, con el siguiente desglose de tareas:

Task name	Story ID	Responsible	Status	Est.
Montaje básico multiproyecto	1	GRC	Done	2
Integración en servidor app	1	GRC	Done	1
Modelo de datos común	2	GRC	Done	1
Modelo de datos local	2	GRC	Done	0,5
Modelo de datos central	2	GRC	Done	0,5
Entidades y persistencia	2	GRC	Done	1
Interfaz de plugin de recolección y parseo	3	GRC	Done	3
Planificador de tareas	4	GRC	Done	6
Protocolo de lectura de ficheros locales	5	GRC	Done	3
Parseo de CSV normalizado	5	GRC	Done	3
Núcleo de recolección	6	GRC	Done	6
Búsqueda de observaciones	7	GRC	Done	2
UI filtrado de observaciones	7	GRC	Done	3
Tabla de observaciones	7	GRC	Done	4
Control de umbrales	8	GRC	Done	6

Ilustración 17 - Planificación iteración 1

ITERACIÓN 2: REPRESENTACIÓN GRÁFICA Y MAPA

El objetivo de la iteración 2 es construir:

- El núcleo de recolección en su completitud, añadiendo la función de **cálculo de observaciones derivadas**.
- Las funciones avanzadas de representación de observaciones, incluyendo:
  - Representación **gráfica de observaciones**
  - Representación **tabular de observaciones derivadas**
  - Representación **gráfica de observaciones derivadas**
  - **Mapa de estaciones**
  - **Búsqueda de estaciones**

El esfuerzo para completar la iteración se estimó en **50** días naturales, con el siguiente desglose de tareas:

Task name	Story ID	Responsible	Status	Est.
Cálculo y representación de variables derivadas	9	GRC	Done	9
Representación gráfica de observaciones	10	GRC	Done	12
Integración de GoogleMaps	11	GRC	Done	9
Geolocalización de estaciones	11	GRC	Done	6
Tooltip de estaciones	12	GRC	Done	6
Búsqueda de estaciones	13	GRC	Done	6
Revisión de sprint	14	GRC	Done	2

Ilustración 18 - Planificación iteración 2

ITERACIÓN 3: INTEGRACIÓN EN TWITTER Y CONFIGURACIÓN

El objetivo de la iteración 3 es completar el Sistema implementando las últimas funcionalidades, a saber:

- Generación de **comentarios** e integración con **Twitter**
- **Configuración de la estación**
- **Configuración del perfil de usuario**

El esfuerzo para completar la iteración se estimó en **22** días naturales, con el siguiente desglose de tareas:

Task name	Story ID	Responsible	Status	Est.

Desarrollo de un Sistema de recogida y normalización de información meteorológica de carácter distribuido y social

Configuración de estación	15	GRC	Done	3
Integración con Twitter	16	GRC	Done	6
Generación de comentarios	17	GRC	Done	9
Revisión final	19	GRC	Done	4

Ilustración 19 - Planificación iteración 3

ANÁLISIS DEL CÓDIGO

Como ejercicio adicional, además de como práctica estrictamente útil durante el desarrollo del proyecto, se ha integrado el código en una herramienta de **análisis de calidad** del mismo, en este caso SonarQube.

La herramienta proporciona información para relativa a diferentes dimensiones del proyecto:

MÉTRICAS DE VOLUMEN

Se proporcionan una serie de métricas para dar idea del **tamaño** del Sistema en términos de: líneas totales y líneas de código; ficheros y clases; funciones, comandos y métodos.

Los valores medidos para el proyecto son los siguientes:

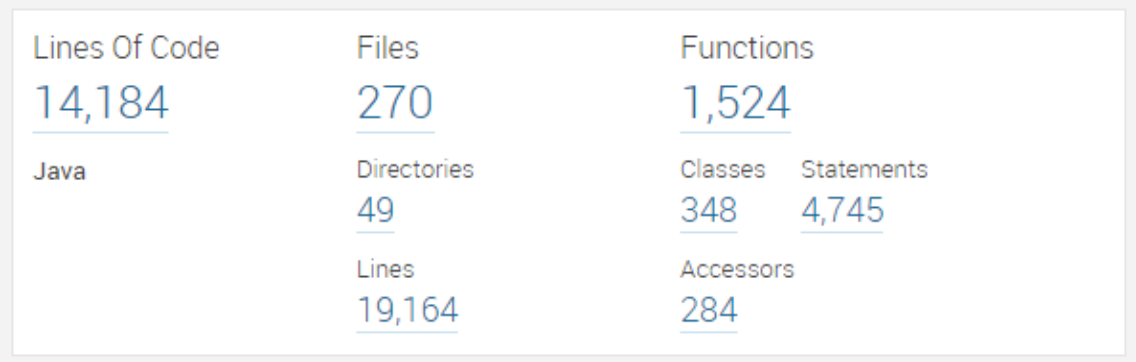

Ilustración 20 - Métricas de volumen

Existe otro indicador de volumen orientado a las líneas de **documentación** introducidas, especialmente en APIs públicas. El valor obtenido dentro del proyecto es un tanto pobre, aunque también hay que indicar que muchos de los elementos del Sistema se han diseñado extendiendo interfaces. En este escenario se ha documentado los métodos de las interfaces, pero no así los métodos de las clases que las implementan por considerarse redundante e inconveniente para el mantenimiento del propio comentario.

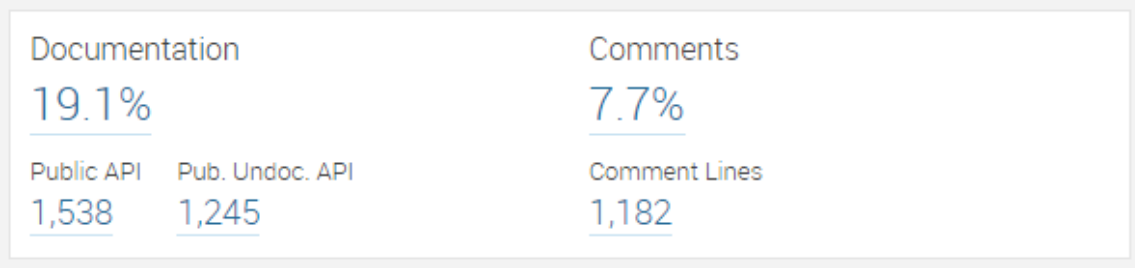

Ilustración 21 - Indicadores de documentación del código

INDICADORES DE CUMPLIMIENTO DE REGLAS

SonarQube aplica al código proporcionado una serie de algoritmos tanto de estilo de código, como de validez (posibles errores), complejidad algorítmica, etc. El comportamiento es configurable, sin embargo en el presente proyecto se ha trabajado con la configuración por defecto. A partir de este análisis, va a registrar una serie de **incumplimientos** que, en mayor o menor medida, deberían ser revisados. El porcentaje de incumplimientos respecto al volumen del proyecto nos dará unos indicadores de calidad que después serán catalogados con valores entre la A y la E (como el sistema norteamericano de notas). Los incumplimientos o incidencias estarán agrupados según **gravedad**.

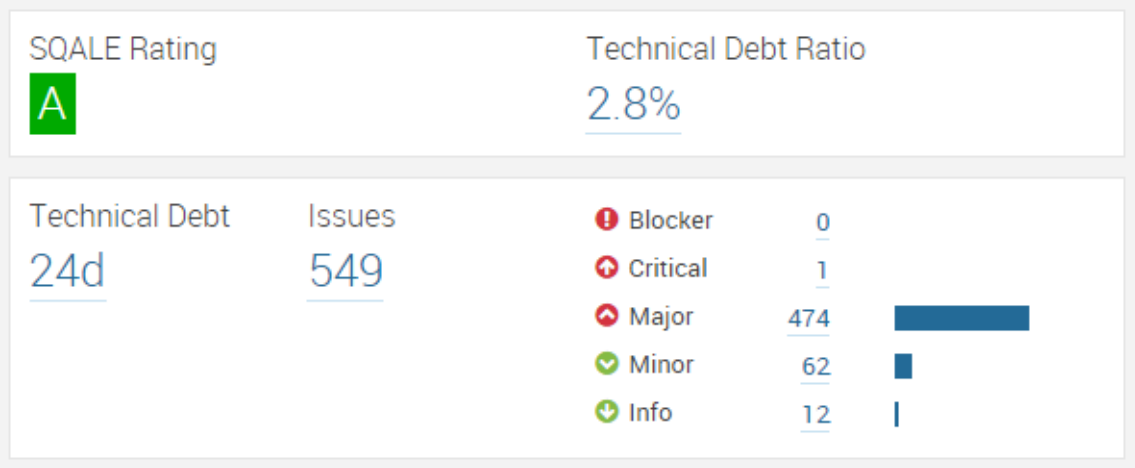

Ilustración 22 - Índice de incumplimiento de reglas

En la figura podemos ver que un 2,8% del código del Sistema presenta algún incumplimiento. El objetivo de desarrollo era mantenerse por debajo del 5%. Otro objetivo era eliminar completamente las incidencias bloqueantes y críticas, y la

mayoría fueron eliminadas salvo un falso positivo marcado como crítico que se detallará más adelante.

Otro de los análisis que se realizan es el de **duplicación** de bloques de código, que da una idea de una de las dimensiones de la modularidad del Sistema.

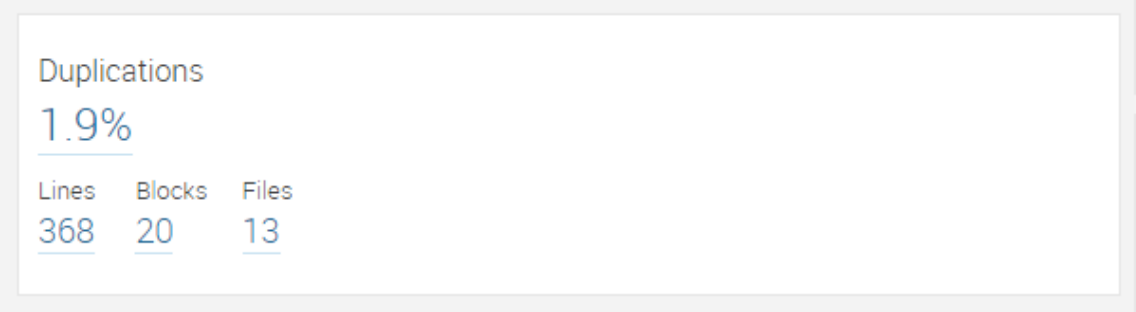

Ilustración 23 - Índices de duplicación

El valor detectado en el presente desarrollo es bajo, si bien tampoco es completamente fiable ya que ha marcado como duplicaciones lo que son simples combinaciones de getters y setters de diferentes entidades que son frecuentes.

INDICADORES DE PROBLEMAS DE DISEÑO

Además de posibles errores o de cuestiones más o menos estilísticas, es posible evaluar en cierta medida la calidad del diseño del código basándose en algunos indicadores.

El índice de **dependencias cruzadas** indica cómo las clases y paquetes desarrollados dependen unos de otros, y si estas dependencias se dirigen en un único sentido lineal (la situación ideal) o si existe algún tipo de cruce o incluso ciclo (situación que debe mantenerse bajo control).

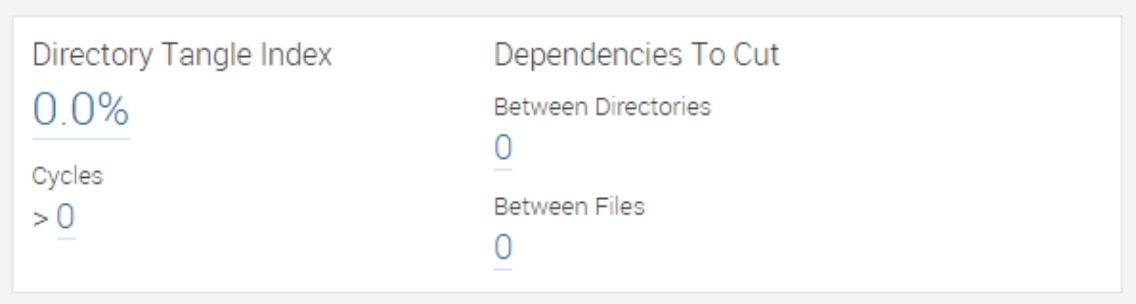

Ilustración 24 - Indicadores de dependencias cruzadas

Indicador de **complejidad ciclomática** de funciones y clases, que ayuda a determinar qué clases o métodos podrían necesitar ser divididos en otras clases o métodos. La distribución del proyecto es mayoritariamente entre 1 y 4, que son buenos valores. De plantearse la refactorización, sería necesario comenzar con aquellas clases que son mostradas en el ranking de complejidad.

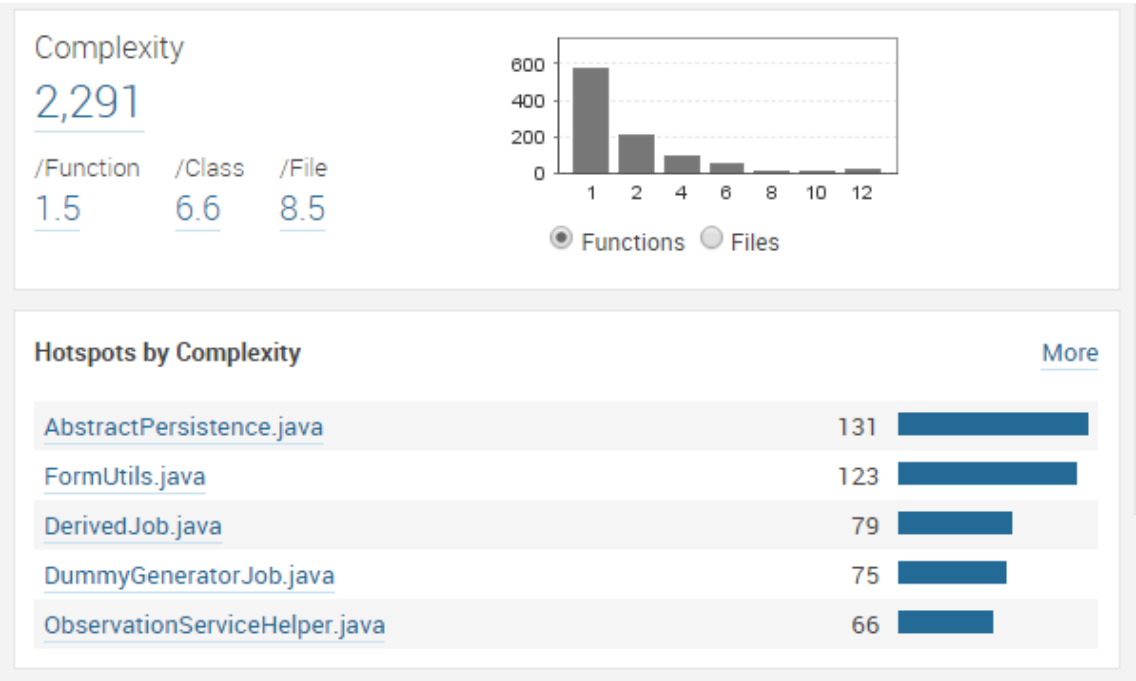

Ilustración 25 - Complejidad ciclomática

INFORMACIÓN DETALLADA DE INCUMPLIMIENTOS

Existe un **desglose** según diferentes parámetros de los incumplimientos detectados. El primero de ellos es relativo a componentes, y en el mismo se mostrará visualmente (mediante colores) los paquetes o módulos del Sistema que tienen mejor o peor

índice de incumplimiento. Dichos **paquetes** son además representados dentro del área general con un tamaño que se corresponde con el tamaño del propio paquete dentro del total del código analizado. Este gráfico es interactivo y se puede continuar pulsando en cada uno de los paquetes para obtener el mismo gráfico a niveles inferiores, hasta llegar a las propias clases.

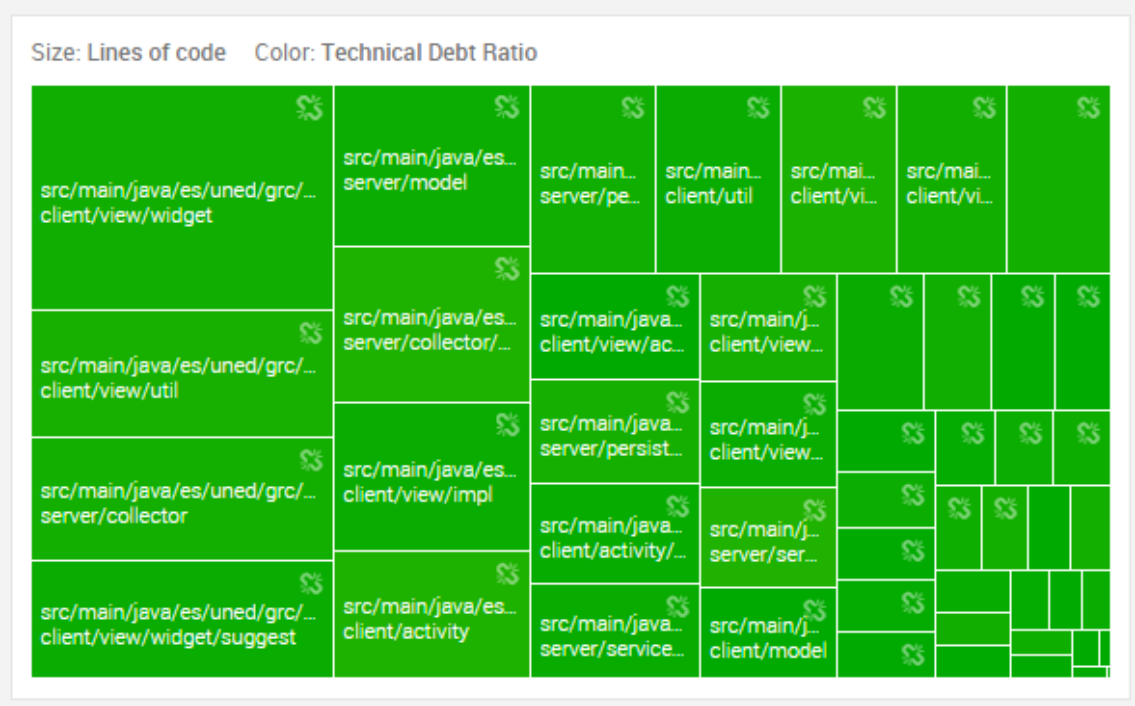

Ilustración 26 - Incumplimientos por módulo

El propio listado de incumplimientos puede ser obtenido con un mayor detalle como se ve en la siguiente figura, donde aparecen agrupados de nuevo según gravedad y es posible obtener el listado completo dentro de cada uno de los niveles. En cada nivel se presentará la lista de incumplimientos agrupados a su vez en el tipo del mismo. El **tipo del incumplimiento** (evitar código comentado, no utilizar excepciones genéricas, no construir bloques de código con complejidad ciclomática excesiva, etc.) se acompaña de la **descripción detallada** del mismo, sus implicaciones y, lo que es más importante, los puntos del código donde se localizan exactamente.

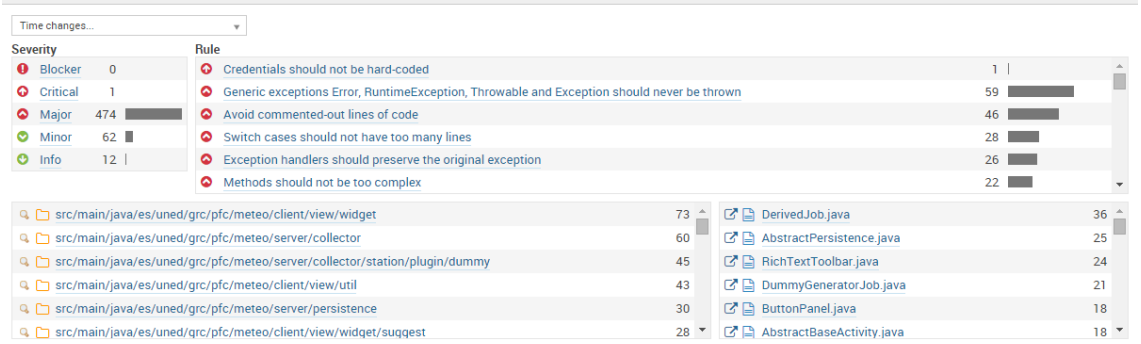

Ilustración 27 - Desglose de incumplimientos

Por último, para cada uno de los incumplimientos se puede identificar la **línea concreta del código** donde se detecta. Como se ve en la siguiente figura, se indica claramente la clase y línea. En este caso se muestra el falso positivo comentado anteriormente donde el analizador ha detectado que se utilizan credenciales en el código fuente, cuando en realidad se trata de un parámetro de configuración referenciado por una constante que contiene el literal “password”.

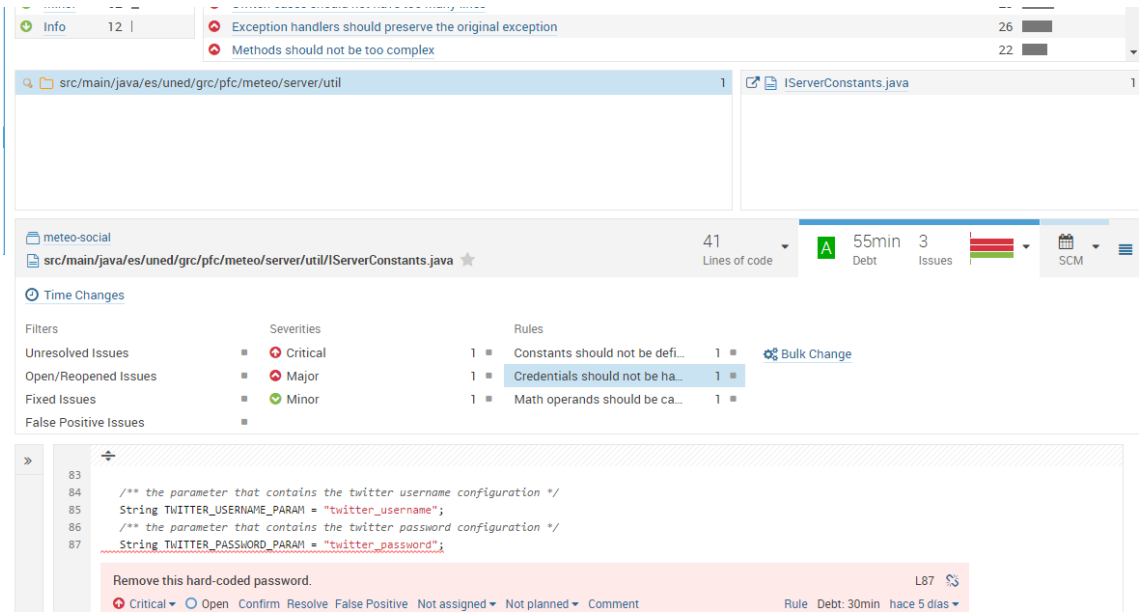

Ilustración 28 - Detalle de incumplimiento

COBERTURA Y ÉXITO DE TESTS AUTOMÁTICOS

La herramienta proporciona información sobre el éxito de los **tests automáticos** desarrollados en el proyecto así como del porcentaje de cobertura de código que

proporcionan los test. Sin embargo, como parte del PFC no se han desarrollado tests unitarios y por lo tanto este indicador es siempre cero.

CONCLUSIONES

Se ha completado el desarrollo de un Sistema que cumple los cuatro objetivos identificados inicialmente:

- Elaborar mecanismos de recopilación de observaciones meteorológicas desde estaciones o sensores estándares del mercado.
- Construir un sistema de normalización y control de información observada, que permita la composición de reglas de calidad de la información generada y que a su vez sea capaz de calcular medidas agregadas o derivadas de las anteriores.
- Consolidación de la información recopilada de manera anónima en un sistema web geolocalizado
- Proporcionar mecanismos de publicación de comentarios en redes sociales existentes (Twitter).

Se ha enmarcado el proyecto dentro de un entorno global, y se ha definido su alcance dentro de dicho entorno. Se ha elaborado un catálogo formal de requisitos que ha servido como base para la especificación de un análisis de subsistemas. A su vez este análisis de subsistemas supuso un buen punto de entrada para diseño y planificación. El diseño se ha basado en patrones estándares de mercado que aseguran la claridad y mantenibilidad del Sistema. Se ha conducido el desarrollo realizando un seguimiento de las tareas planificadas, si bien no se ha conseguido realizar un seguimiento diario que permitiera estimaciones de velocidad o finalización.

Se han utilizado tecnologías interesantes como GWT y se ha comprobado que permite la producción de interfaces de usuario asíncronos muy ricos y fomenta un diseño arquitectónico limpio totalmente compatible con el resto de patrones y directrices clásicos de J2EE, que de hecho es la tecnología básica del Sistema.

Se han realizado mediciones de la calidad del código, obteniendo valores razonables en casi todos los índices y muy buenos en algunos.

LÍNEAS DE EVOLUCIÓN

Las posibilidades de evolución son potencialmente interminables, especialmente cuando el sistema podría estar dirigido a una red amplia de usuarios con conocimientos funcionales y técnicos muy detallados. Se indican aquí algunas de las líneas agrupadas según categorías:

TRATAMIENTO DE LA OBSERVACIÓN

- Ampliar el conjunto de controles de calidad disponibles (de consistencia temporal, consistencia espacial, análisis numérico, etc.)
- Ingresar información de carácter no alfanumérica, bien como observaciones o como parte de los metadatos de la estación, como imágenes, audios y vídeos generados por estaciones o por el usuario

PRESENTACIÓN DE INFORMACIÓN

- Agrupar determinadas variables, según configuración, dentro de las mismas gráficas, como es en ocasiones común dentro del tratamiento meteorológico (p.e. temperatura + humedad relativa o velocidad + dirección del viento)
- Comparar los valores recibidos por varias estaciones mediante la presentación de tablas sincronizadas y gráficas combinadas.

INTEGRACIÓN

- Implementación de protocolos de estaciones en el mercado. Inicialmente al menos uno o dos correspondientes con las que se identifique como más comunes entre la comunidad
- Difusión de información a través de otras redes sociales como Facebook o incluso *photologs* tipo Instagram

CONFIGURACIÓN

- Añadir un proceso de registro de usuarios clásico que permita a cualquiera darse de alta con una dirección de correo electrónico.

- Configuración exacta de los comentarios a publicar, es decir, permitir al usuario detallar qué se publica, cuándo se publica y cómo (en qué formato o con qué palabras exactas) se publica.
- Realización de tests automáticos de conexión al configurar los parámetros de conexión a la estación. Comunicación con la estación para obtener logs o ejecutar comandos de mantenimiento.
- Realización de tests automáticos de conexión con Twitter al configurar los datos de autenticación.

MEJORAS DE ARQUITECTURA O DE SOPORTE AL DESARROLLO

- Empaquetado y despliegue distribuido, donde el usuario pueda instalar una pequeña parte del Sistema que le permita la visualización y tratamiento de la información generada por su propia estación, y que colabore con un elemento central del Sistema tanto para la consolidación de informaciones locales como para la recuperación y representación de informaciones globales (de otros usuarios).
- Tests unitarios con JUnit
- Integración continua mediante Jenkins

BIBLIOGRAFÍA

Nombre	URL
Core J2EE Patterns	http://corej2eepatterns.com/
Design patterns make for better J2EE apps	http://www.javaworld.com/article/2074321/design-patterns/design-patterns-make-for-better-j2ee-apps.html
GWT Activities and Places	http://www.gwtproject.org/doc/latest/DevGuideMvpActivitiesAndPlaces.html
GUI Architectures	http://martinfowler.com/eaaDev/uiArchs.html
GWT Charts API Documentation	http://gwt-charts.appspot.com/
GWT Maps API Documentation	https://developers.google.com/maps/?csw=1
GWT Showcase	http://www.gwtproject.org/examples.html
Jtwitter Documentation	http://www.winterwell.com/software/jtwitter.php
Model-view-presenter @Wikipedia	http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter
Scrum Alliance	https://www.scrumalliance.org/
SonarQube	http://www.sonarqube.org/

Twitter API Documentation	**https://dev.twitter.com/rest/public**
World Meteorological Organization	**https://www.wmo.int/**

LISTADO DE ABREVIATURAS Y ACRÓNIMOS

Acrónimo	**Significado**
DAO	**Data Access Object**
GWT	**Google Web Toolkit**
ICAO	**International Civil Aviation Organization**
J2EE	**Java 2 Enterprise Edition**
MVP	**Model-View-Presenter**
VO	**Value Object**
WMO	**World Meteorological Organization**