

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./Da.

(firma)

Secretario: D./Da.

(firma)

Vocal: D./Da.

(firma)

Fecha de lectura y defensa:

Calificación:

# Resumen

Durante mi carrera profesional me he desempeñado en diferentes aéras 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 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 de 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.

http://www.wordle.net/

# Abstract

Durante mi carrera profesional me he desempeñado en diferentes aéras 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 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 de 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.

Keywords: observation, station, network, meteorology,

# ÍNDICE

Resumen 3

Abstract 4

ÍNDICE 5

ÍNDICE DE FIGURAS 7

Introducción 8

Objetivos 8

Planificación 8

Contexto de la observación meteorológica 11

Agencias e instituciones 11

Uso personal 12

Procesos meteorológicos y climatológicos 13

Definición de alcance 16

Recopilación de observaciones 16

Almacenamiento y estandarización de observaciones 18

Interfaz de lectura de observaciones 20

Integración social 21

Gestión de metadatos y perfiles de estaciones y usuarios 22

Análisis de requisitos 24

Especificación formal de requisitos 24

Casos de uso 27

Subsistemas de análisis 39

Subsistema: Núcleo de recolección y difusión 40

Subsistema UI local 41

Subsistema: Interfaz de usuario central 42

Diseño del Sistema 43

Diseño de la página web 43

Vista global de componentes 45

Componentes de servidor 46

Componentes de cliente 49

Componentes compartidos 54

Entorno tecnológico 54

Planificación y ejecución 57

Análisis del código 58

Métricas de volumen 58

Indicadores de cumplimiento de reglas 58

Indicadores de problemas de diseño 58

Información detallada de incumplimientos 58

Cobertura y éxito de tests automáticos 58

Conclusiones 60

Líneas de evolución 61

Bibliografía 62

Listado de abreviaturas y acrónimos 64

# ÍNDICE DE FIGURAS

[Ilustración 1 - Planificación incremental 9](#_Toc418432315)

[Ilustración 2 - Procesos meteorológicos 14](#_Toc418432316)

[Ilustración 3 - Objetivos engranados 16](#_Toc418432317)

[Ilustración 4 – Normalización de información 20](#_Toc418432318)

[Ilustración 5 - Caso de uso: consulta de observaciones 28](#_Toc418432319)

[Ilustración 6 - Caso de uso: núcleo temporizado 33](#_Toc418432320)

[Ilustración 7 - Caso de uso: configuración del Sistema 37](#_Toc418432321)

[Ilustración 8 - Diagrama de subsistemas 40](#_Toc418432322)

[Ilustración 9 - Estructura de la página web 44](#_Toc418432323)

[Ilustración 10 - Vista global de componentes 46](#_Toc418432324)

# 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 subojetivos:

* 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 dificícilmente 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 más 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 30/04/2014, 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 1 - 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 metorológico 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 Autonómas, 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 especfíciamente aeronáutico con aeropuertos y aerodromos 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**, cada uno de los cuales de **diversos modelos** y **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 estáción meteorológica es básicamente una composición de los siguientes elementos:

* **Una serie de** **sensores** como termómetros, anemómetros, barómetros, pluviómetro, 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 hardware 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.

## Uso personal

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 2 - 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 comunciaciones).

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.

### 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 3 - 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 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 ya 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 proporciona 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))

## 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 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 tengan en consideración esta posibilidad.
* En **tipología**: la mayoría de las estaciones meteorológicas cuentas 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 de forma que dicho no impacte 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 superior a un mínimo o inferior 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 en la base). 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 de 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 4 – 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 mismo, 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, 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, si bien se considerará en tiempo de diseño la posible ampliación futura en este sentido.

## 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 presentado. 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 la 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

1. Los usuarios podrán autenticarse mediante una pareja de usuario y contraseña.
2. El Sistema se asociará con una estación meteorológica que producirá determinadas mediciones, como pueden ser la temperatura, humedad, velocidad y dirección del viento, etc.
3. El Sistema ejecutará con la periodicidad configurada una lectura de observaciones desde la estación meteorológica asociada.
4. 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.
5. Cada lectura que se produzca desde la estación se caracterizará mediante la siguiente información:
   1. Fecha de adquisición.
   2. Fecha de observación (de la propia medida por parte de la estación)
   3. Lista de observaciones (mediciones).
   4. Lista de flags de calidad (uno por cada medición)

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

1. 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:
   1. Los provistos por la propia estación que se considerarán valores físicos.
   2. Los sobreescritos (configurados) por el usuario que se considerarán valores actuales y locales.
2. 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 también de los locales, o “sospechoso” si el valor está fuera de alguna de ellos.
3. A partir de las mediciones producidas en períodos de tiempo fijos pero configurables, el Sistema calculará:
   1. El valor medio, mínimo y máximo en tres períodos diferenciados del día:
      1. Noche: entre las 00 y las 06 horas
      2. Mañana: entre las 06 y las 12 horas
      3. Tarde: entre las 12 y las 18 horas
      4. Tarde-noche: entre las 18 y las 24 horas
   2. El valor medio, mínimo y máximo en un día
   3. El valor medio, mínimo y máximo en un mes
4. 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 marcará 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 configurada. 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.
5. 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. Unidades en gráficas
6. 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-.
7. 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.
8. 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 resultado de estaciones se podrán consultar los datos más básicos de cada una y además será posible:
   1. 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.
   2. 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.
9. 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.
10. El usuario podrá caracterizar la estación asociada, completando los siguientes datos:
    1. El modelo de la estación, a elegir de entre los registrados en el Sistema
    2. Los parámetros declarados por el plugin seleccionado, que típicamente contendrán valores como direcciones IP, puertos TCP/UDP, usuarios, contraseñas, rutas, nombres de ficheros, etc.
    3. 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.
    4. Los metadatos propiamente dichos como mínimo la dirección postal y la latitud, longitud y altitud donde se ubica.
11. 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

1. La inclusión de nuevos formatos de datos y protocolos de comunicaciones será sencilla, con un enfoque tipo “plugins”.
2. El Sistema de almacenamiento de observaciones debe ser flexible respecto a las observaciones que puede producir cada estación, y será adaptable a nuevas observaciones.
3. 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 mismas.

## Casos de uso

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

FALTA EL LOGIN

### Consulta de observaciones

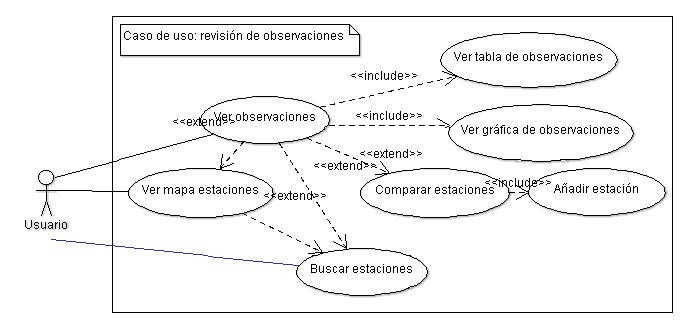


Ilustración 5 - Caso de uso: consulta de observaciones

|  |  |
| --- | --- |
| **Caso de uso:** | **Ver observaciones** |
| **Código:** | **CU01** |
| **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** | Propietario de la estación (único usuario) |
| **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 mediante un rango de fechas y/o un conjunto de observaciones a representar, actualizándose la representación respecto a dicho filtro. | | |
| **Flujo Alternativo:** | |
| |  | | --- | | * Desde el mapa de estaciones, el usuario selecciona una estación y obtiene el listado de observaciones de la misma en las últimas 24 horas. * 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:** | **CU02** |
| **Descripción:** | Una vez aplicada una determinada selección de estación y filtros, el Sistema representa 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 incluído en el CU “Ver Observaciones” y por tanto el actor corresponde a aquél. |

|  |  |
| --- | --- |
| **Caso de uso:** | **Ver gráfica de observaciones** |
| **Código:** | **CU03** |
| **Descripción:** | Una vez aplicada una determinada selección de estación y filtros, el Sistema representa una una gráfica para cada una de las observaciones 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 incluído en el CU “Ver Observaciones” y por tanto el actor corresponde a aquél. |

|  |  |
| --- | --- |
| **Caso de uso:** | **Ver mapa estaciones** |
| **Código:** | **CU04** |
| **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: usuario propietario, geoposicionamiento, última observación recibida y se permitirá el acceso al detalle de la misma. |
| **Actor** | Propietario de la estación (único usuario) |
| **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:** | **CU05** |
| **Descripción:** | Es posible listar todas las estaciones del Sistema, filtrado por ciudades, provincias o países. |
| **Actor** | Propietario de la estación (único usuario) |
| **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, su propietario y la última observación recibida. Para cada una de ellas se podrá acceder al detalle de observaciones de cada estación, o se podrá situar sobre el mapa. | | |

|  |  |
| --- | --- |
| **Caso de uso:** | **Comparar estaciones** |
| **Código:** | **CU06** |
| **Descripción:** | El usuario podrá añadir a un determinado listado de observaciones (con o sin filtrado previo) los resultados de cualquier otra estación del Sistema. |
| **Actor** | Propietario de la estación (único usuario) |
| **Flujo Normal:** | |
| |  | | --- | | * Desde la vista de observaciones de una estación, el usuario añade, mediante una búsqueda, una nueva estación cuyos datos se recuperan aplicando el mismo filtrado que ya existía para la anterior, y se representa de forma que sea posible la comparación visual entre ambas estaciones. | | |

### Núcleo temporizado

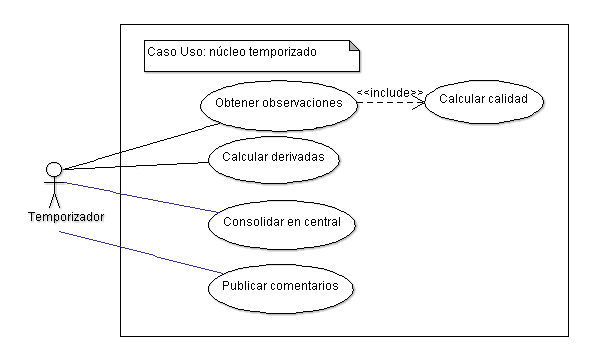


Ilustración 6 - Caso de uso: núcleo temporizado

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 (incluye calcular calidad)** |
| **Código:** | **CU07** |
| **Descripción:** | Periódicamente (según la configuración temporal almacenada para la estación vinculada) el Sistema se conectará con la estación para obtener las observaciones generadas por la misma, les aplicará el control de calidad en función de los valores umbrales también configurados, 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 * 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. | | |

|  |  |
| --- | --- |
| **Caso de uso:** | **Calcular derivadas** |
| **Código:** | **CU08** |
| **Descripción:** | Periódicamente (según configuración temporal) el Sistema inspeccionará las observaciones obtenidas para la estación vinculada 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:** | **Consolidar en central** |
| **Código:** | **CU09** |
| **Descripción:** | Periódicamente (según configuración) el Sistema transmite al servidor central las observaciones de la estación vinculada, si es que la configuración de difusión lo permite. |
| **Actor** | Temporizador (automático) |
| **Flujo Normal:** | |
| |  | | --- | | * El temporizador se dispara en el momento configurado (periódicamente) * El Sistema comprueba si existen observaciones no consolidadas en el servidor central, y si la consolidación está permitida para la estación vinculada. En tal caso, transmite las observaciones pendientes al servidor central. | | |
| **Flujo Alternativo:** | |
| |  | | --- | | * El temporizador se dispara en el momento configurado pero encuentra que no existen nuevas observaciones, o bien que la consolidación (difusión) no está permitida. Por lo tanto, no realiza acción alguna y duerme hasta el siguiente momento configurado | | |

|  |  |
| --- | --- |
| **Caso de uso:** | **Publicar comentarios** |
| **Código:** | **CU10** |
| **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 emprede ninguna acción hasta el siguiente momento configurado. | | |

### Configuración del Sistema

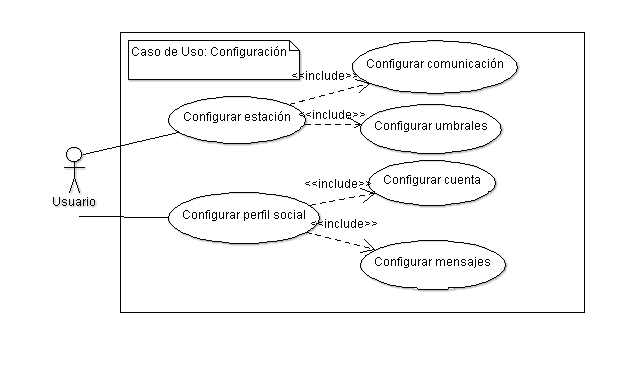


Ilustración 7 - Caso de uso: configuración del Sistema

|  |  |
| --- | --- |
| **Caso de uso:** | **Configurar estación (incluye Configurar comunicación y Configurar umbrales)** |
| **Código:** | **CU11** |
| **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 * Período de recuperación de observaciones de la estación. * Período de difusión al servicio central (o deshabilitar consolidación) * Valores umbrales (máximo y/o mínimo o ninguno) para cada una de las observaciones a recuperar. |
| **Actor** | Propietario de la estación (único usuario) |
| **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 (o no) 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. * Independientemente del modelo, el Sistema presenta los parámetros de períodos (observación, recuperación, difusión) | | |

|  |  |
| --- | --- |
| **Caso de uso:** | **Configurar perfil social (incluye Configurar cuenta y Configurar mensajes)** |
| **Código:** | **CU12** |
| **Descripción:** | Permite al usuario configurar la el perfil y la difusión de mensajes según los siguientes parámetros:   * Habilitación/deshabilitación de la difusión * Usuario de Twitter para la difusión * Lista de mensajes a enviar para cada cálculo de diferentes períodos |
| **Actor** | Propietario de la estación (único usuario) |
| **Flujo Normal:** | |
| |  | | --- | | * El usuario introduce su usuario de Twitter y configura una lista de mensajes de texto a difundir para:   1. Cuando un valor máximo de una variable en el período de un día supera un determinado umbral   2. Cuando un valor mínimo de una variable en el período de un día está por debajo de un determinado umbral   3. Cuando un valor medio de una variable en el período de un día está por debajo de un determinado umbral   4. Cuando se conocen los valores máximos, medios y mínimos de una variable en un mes. * Dentro de los textos, en formato libre, se introducirán expresiones regulares con el nombre de las variables a incluir, que serán posteriormente substituidas por el Sistema por los valores específicos de cada mensaje generado. | |  | | |
| **Flujo Alternativo:** | |
| |  | | --- | | * El usuario deshabilita la difusión mediante el perfil social. Toda la configuración permanece y se vuelve a utilizar en casi de volver a habilitar. | | |

## 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:

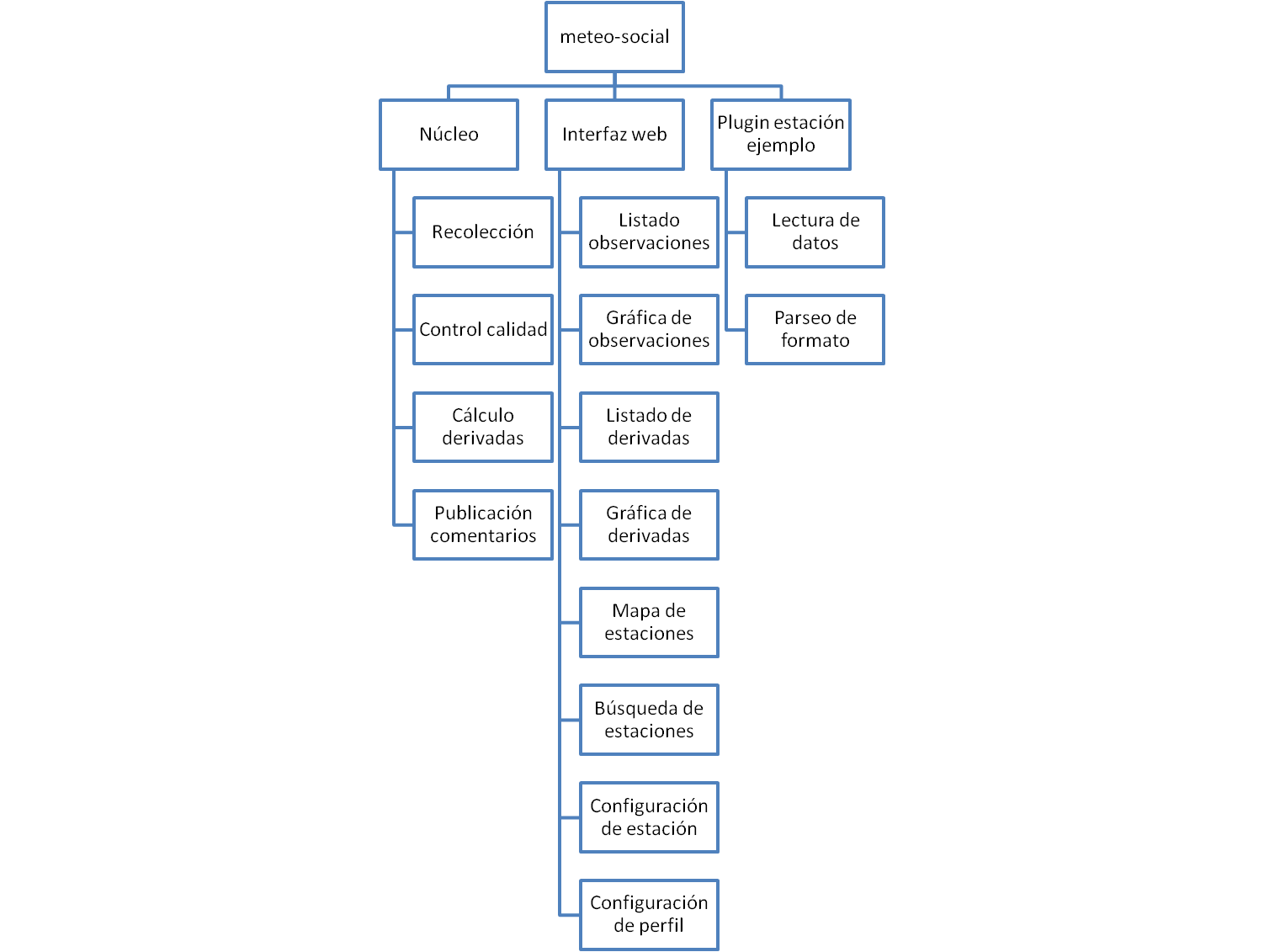


Ilustración 8 - Diagrama de subsistemas

## Subsistema: Núcleo de recolección y difusión

### Función: Recolección

* Temporización de recolección
* Interfaz de plugin de recolección y parseo
* 1 protocolo de comunicación con la estación
* 1 formato de parseo de observaciones
* Recuperación de observaciones no ingresadas
* Almacenamiento de observaciones en base de datos

### Función: Control de calidad

* Aplicación de límites por variable medida
* Fijar marcas de calidad para cada observación

### Función: Generación de comentarios

* Temporización de generación de comentarios
* Evaluación de comentarios y substitución por valores observados
* Conexión con cuenta de Twitter y envío de comentarios generados

### Función: cálculo de derivadas

* Temporización del cálculo de derivadas
* Identificación de derivadas afectadas y ejecución de los cálculos afectados

### Función: Difusión

* Temporización de la difusión de observaciones
* Envío de observaciones (observadas y calculadas) no consolidadas al servidor central
* Recepción y almacenamiento en servidor central de observaciones procedentes de los nodos distribuidosSubsistema: Interfaz de usuario local

## Subsistema UI local

### Función: listados de observaciones

* Interfaz de filtrado de observaciones
* Recuperación de lotes de observaciones, paginadas, aplicando filtros
* Presentación paginada de lotes de observaciones

### Función: gráficas de observaciones

* Presentación en gráficas de líneas interactivas de lotes de observaciones

### Función: configuración de estación

* Selección de modelo de estación y configuración dinámica de la misma
* Configuración de valores umbrales para variables observadas

### Función: configuración de difusión

* Interfaz de habilitación/deshabilitación difusión de observaciones y períodos de difusión
* I nterfaz de configuración de cuenta de Twitter y habilitación/deshabilitación difusión de comentarios y períodos de difusión

## Subsistema: Interfaz de usuario central

### Función: mapa de estaciones

* Geolocalización de estaciones en Google Maps con funciones de navegación y zoom
* Diálogo de información básica de cada estación

### Función: comparación de estaciones

* Representación en cuadrícula de tablas de observaciones de múltiples estaciones
* Re presentación en cuadrícula de gráficas de observaciones de múltiples estaciones

### Función: búsqueda de estaciones

* Interfaz de filtrado de observaciones por criterios de nombre de usuario, o por criterios geográficos
* Recuperación y presentación dinámica y paginada de estaciones acordes con los filtros proporcionados.

# 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 las páginas y el propio backend. Aquí se expone el resultado de dicho diseño de forma tecnológicamente neutral, pero **puede ser interesante revisar** la [descripción del entorno tecnológico](#_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.

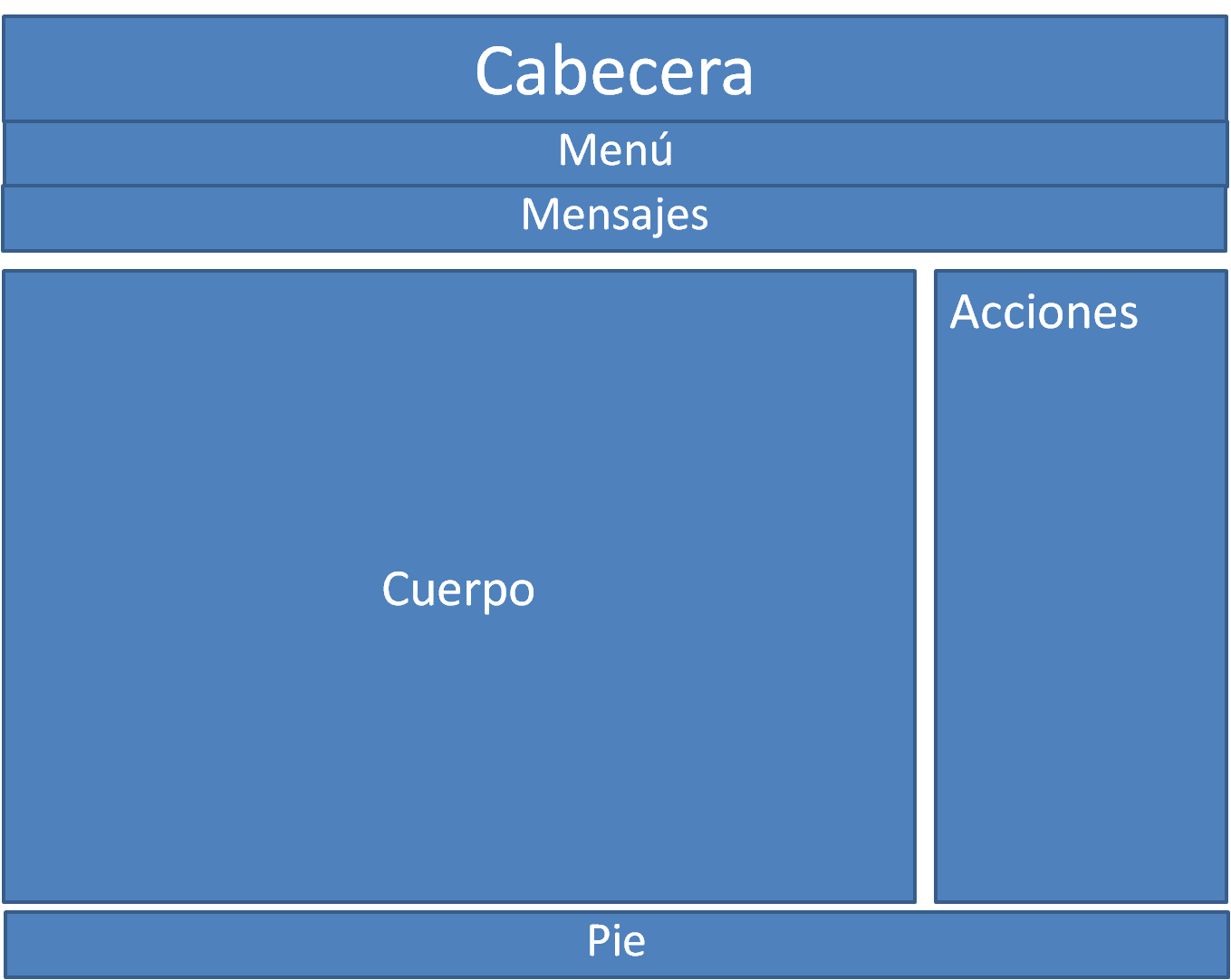


Ilustración 9 - Estructura de la página web

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](#_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](#_Entorno_tecnológico) 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:

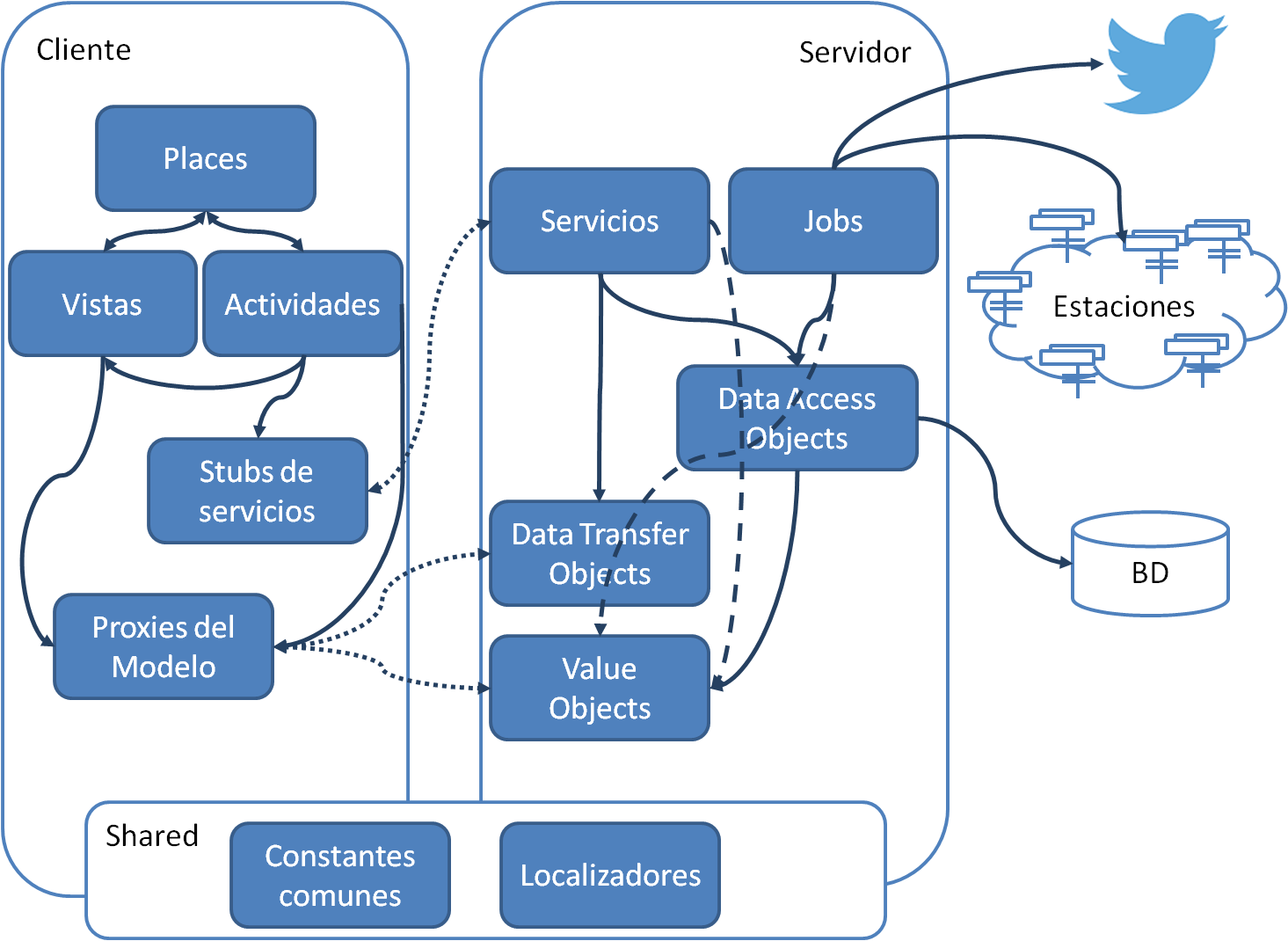


Ilustración 10 - Vista global de componentes

## 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, incluída 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.

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

INSERTAR FIGURA Y DESCRIBIR CADA UNA

### 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 intefaz y una clase abstracta que exponga/implemente dichas operaciones comunes, y será responsabilidad de cada DAO final proveer de 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](#_Proxies_del_modelo), 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 persistences 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 estén presentes 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 leelerlos, 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 recien 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 se especifican de forma declarativa mediante Spring Scheduling Framework.

### 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 están relacionadas 1:1 con cada Servicio.

Los servicios se modelan como prototipos Service de Spring y sus operaciones son expuestas de forma automática por el API RequestFactory de GWT.

## 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 comunciación 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.

INSERTAR FIGURA MVP

En el modelo MVP implantado, la M serían los [proxies del modelo](#_Proxies_del_modelo), la V serían las [vistas](#_Vistas) y la P serían las [actividades](#_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 proporciona 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, menu, 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 vicecersa, 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.

INSERTAR FIGURA DE PLACES, VISTAS Y ACTIVIDADES

### 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, manipulandos 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 muy 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 definitva 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 Inversion 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 el 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 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 el **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 via 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](#_Análisis_del_código) y errores llamada **SonarQube 5.0.1**.

# Planificación y ejecución

# 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 tratar diferentes dimensiones del proyecto:

## Métricas de volumen

A

INSERTAR SNAPSHOT

## Indicadores de cumplimiento de reglas

B

INSERTAR SNAPSHOT

## Indicadores de problemas de diseño

C

INSERTAR SNAPSHOT

## Información detallada de incumplimientos

D

INSERTAR SNAPSHOT

## 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

# Líneas de evolución

* Implementación protocolos de mercado
* Test automático de conexión al poner los parámetros y lectura de logs de la propia estación
* Test de conexión con Twitter
* Configuración de comentarios (qué, cuándo)
* Otras redes sociales
* Imágenes, audios y vídeos generadas por estaciones (no alfanuméricos)
* Agrupar variables en gráficas (p.e. temperatura + humedad relativa)
* Tests unitarios con JUnit
* Integración continua mediante Jenkins
* Despliegue distribuido

# Bibliografía

|  |  |  |
| --- | --- | --- |
| # | Nombre | URL |
| MS01 | Twitter API Documentation | <https://dev.twitter.com/rest/public> |
| MS02 | Jtwitter Documentation | <http://www.winterwell.com/software/jtwitter.php> |
|  | 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> |
|  | Scrum Alliance | <https://www.scrumalliance.org/> |
|  | World Meteorological Organization | <https://www.wmo.int/> |
|  | SonarQube | <http://www.sonarqube.org/> |
|  | GWT Activities and Places | <http://www.gwtproject.org/doc/latest/DevGuideMvpActivitiesAndPlaces.html> |
|  | GUI Architectures | <http://martinfowler.com/eaaDev/uiArchs.html> |
|  | 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> |
|  |  |  |
|  |  |  |

# Listado de abreviaturas y acrónimos

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