UNIVERSITAT OBERTA DE CATALUNYA

Ingeniería Informática

Sistema de visualización SIG del Servei Meteorològic de Catalunya

Alumno/a: Rockywell Otto Inarejos

Dirigido por: Anna Muñoz Bollas

CURSO 2013/14, 2o semestre

Índice de contenido

Índice de tablas

Índice de ilustraciones

1 Descripción del proyecto

El presente proyecto tiene como principal objetivo el desarrollo e implementación de un sistema de visualización de los datos meteorológicos del *Servei Meteorològic de Catalunya*(en adelante SMC), de diferente naturaliza, formato y temporalidad, en un visor web SIG.

Para que un dato meteorológico nos aporte cierta información, ha de estar forzosamente georeferenciado. Sin esta referencia geográfica no se podría ubicar, y su información no tendría utilidad. Por este motivo los sistemas de información geográfica son muy importantes a la hora de analizar datos meteorológicos, en casi cualquiera de sus vertientes, ya sean estudios del cambio climático, realización de predicciones, etc.

Los datos meteorológicos que podemos encontrar en la actualidad, son muy diversos y todos ellos con su propia naturaliza y temporalidad. Tenemos imágenes en forma de fotografía, que hacen referencia a un instante, como pueden ser imágenes de radar o de Meteosat, mediciones válidas para un intervalo de tiempo en forma de variable que se pueden extraer a partir de estaciones meteorológicas automáticas, o modelos numéricos de predicción que pueden hacer referencia a distintos intervalos de tiempo. Todo ésto dificulta la posibilidad de visualización de todos estos datos de manera conjunta.

Se tomará como referencia las necesidades del SMC, en la medida de lo posible se generalizarán estas necesidades funcionales para que el sistema pueda ser utilizado en otras empresas meteorológicas. El SMC es un organismo público de la *Generalitat de Catalunya* que tiene como propósito la divulgación y predicción de la meteorología en Cataluña. Las actividades de esta empresa son muy diversas, entre ellas destacan el pronóstico, vigilancia de episodios de riesgo meteorológico y el estudio del cambio climático en Cataluña. Actualmente el SMC dispone de diversos datos meteorológicos georeferenciados, como son datos de estaciones automáticas, imágenes de diferentes productos de Radar y Meteosat, información de rayos, modelos numéricos de predicción y observaciones manuales. Estos datos se almacenan en diferentes formatos, ya sea en bases de datos(como los rayos o la información de estaciones), o en formato de imagen(raster). En el SMC actualmente, para la visualización de estos datos, se recurre a herramientas específicas para cada dato, careciendo de la posibilidad de una visualización global y estándar.

Para cubrir la necesidad de una visualización conjunta de todos estos datos meteorológicos, el siguiente proyecto propone el desarrollo de un sistema único e integrado de visualización. Donde el usuario podrá consultar datos de una forma uniforme, ágil y estándar. Incluyendo la posibilidad de poder superponer diferentes datos, tanto históricos, como actuales, así como la posibilidad de realizar animaciones temporales de dichos datos, así como valores en puntos geográficos concretos. El sistema estará formado de un visor SIG Web y un servidor de mapas. La comunicación entre estos dos componentes se realizará mediante protocolos estándar.

2 Objetivos

El siguiente proyecto tiene como objetivo el desarrollo e implementación de un sistema visualización de datos meteorológicos del SMC. Mediante esta herramienta los técnicos del SMC, podrán superponer y consultar información meteorológica en un mismo mapa. Facilitando y agilizando sus tareas, como pueden ser validaciones de modelos numéricos, realizar predicciones, validación de datos de sus estaciones meteorológicas, etc. Permitirá al usuario poder consultar tanto datos históricos, como actuales, de una forma uniforme, ágil y estandarizada. La herramienta de visualización, permitirá realizar animaciones de datos en un intervalo de tiempo, dando así la posibilidad de poder visualizar su evolución en el tiempo. El sistema constará de dos grandes

módulos o capas: un cliente a modo de visor Web SIG y un servidor de mapas.

Para el proyecto se acotarán los requisitos funcionales de los técnicos, para que la implementación del sistema, encaje en el periodo de tiempo establecido por la asignatura. No todos los técnicos del SMC realizan las mismas tareas y requieren funcionalidades diferentes. Ésto es común en todas las organizaciones y/o empresas dedicadas a la meteorología. Se plantear la totalidad del sistema en diferentes fases y realizar las entregas en forma de versiones, añadiendo en cada nueva versión nuevas funcionalidades e información. En el presente proyecto se implementará un visor Web SIG que tiene como objetivo cubrir los requisitos que son comunes a todos los técnicos del SMC, que es la visualización de datos meteorológicos en un visor Web SIG, dejando para otras versiones la implementación de funcionalidades meteorológicas más especificas, como por ejemplo meteorogramas, gráficas, cortes verticales, etc.. Por lo tanto éste proyecto es una primera versión del sistema.

El proyecto se realizará en la medida de lo posible utilizando software *OpenSource*. De esta forma se abarata el proyecto y nos aprovechamos de las continuas evoluciones que realizan las comunidades sobre estos software, posibilitando añadir nuevas mejoras al sistema en futuras versiones.

La capa de servidor será consultada mediante protocolos estándar por la capa cliente. Ésto da un valor añadido al sistema, haciendo posible que otros tipos de clientes, (como clientes SIG de escritorio, clientes móviles, etc.) puedan realizar consultas al servidor utilizando estos protocolos estándar. Incluso abrirá la puerta a la posibilidad de ofrecer los servicios del servidor de mapas a clientes externos vía Web.

2.1 Objetivos generales

- Comprender los conceptos de la tecnología SIG y su metodología.
- Conocer las diferentes estructuras de datos y topologías con la que trabaja un SIG.
- Saber plantear un proyecto SIG.
- Estudio y compresión de los diferentes datos de muestra que se utilizarán en el proyecto.
- Visualizar datos georeferenciados de diferente naturaleza y origen.
- Integrar en una misma visualización datos raster y vectoriales.
- Aprender las habilidades para generar y manipular datos geográficos.

2.2 Objetivos específicos

- Ofrecer a los técnicos del SMC una herramienta única, homogénea y ágil, para la visualización y superposición de los diferentes productos del SMC.
- Conocer a fondo las posibilidades de un Servidor de mapas, así como su instalación y explotación.
- Visualizar de forma clara series temporales de datos.
- Diseñar e implementar una interfaz de usuario para la visualización de datos georeferenciados.
- Implementar un servidor de mapas que podrá ser consultado mediante protocolos estándar SIG, como WMS, WMTS, WFS, etc. Ofreciendo la posibilidad de dar servicios a distintos clientes.

- Diseñar e implementar una aplicación cliente Web SIG.
- Conocer las herramientas necesarias para poder implementar y diseñar un sistema de visualización de información georeferenciada.

2.3 Alcance

A continuación se describe el alcance del proyecto:

- Estudio de las tecnologías SIG y de su metodología, así como de los diferentes conceptos que utiliza, como por ejemplo: proyecciones, topologías, raster, vector, etc.
- Análisis de los diferentes tipos de información meteorológica que dispone el SMC, y definir cuáles son potencialmente útiles en un sistema de visualización.
- Establecer un criterio único y coherente para la visualización temporal de datos, ya que éstos hacen referencia a intervalos de tiempo diferentes.
- Análisis e instalación de un servidor de mapas.
- Estudio y análisis de las diferentes posibilidades de consulta estándar a un servidor de mapas.
- Optimizar el rendimiento del servidor de mapas aplicando algún tipo de caché.
- Configurar el servidor para la consulta de datos de diferente origen, ya sean imágenes, ficheros GRIB[1] o bases de datos.
- Análisis de las diferentes librerías disponibles para la visualización de mapas en un cliente web SIG, optando finalmente por alguna de ellas.
- Análisis y uso de los diferentes *frameworks* javascript disponibles para la implementación de la interfaz de usuario.
- Diseño e implementación de un visor Web SIG.
- El visor SIG permitirá realizar consultas de datos históricos y actuales.
- El visor permitirá realizar animaciones temporales de los datos.
- El visor permitirá consultar valores de las capas visualizados mediante un punto geográfico.
- Visualización de datos tanto en formato vectorial como raster.
- Redacción de la memoria del proyecto.
- Realización de la presentación virtual.
- Participación en el debate virtual.

1 Fichero estándar(World Meteorological Organizations Comission) que se utiliza normalmente para almacenar datos meteorológicos, tanto datos históricos como predicciones. http://en.wikipedia.org/wiki/GRIB

3 Planificación

A continuación se describe la planificación que se ha seguido para el desarrollo e implementación del proyecto. Las actividades que se desarrollarán en el proyecto se han planificado de forma que encajen con las entregas de la asignatura.

3.1 Entregas clave

Una vez se han desglosado las fases clave del proyecto y su estimación, a continuación se describen las fechas o hitos clave del proyecto.

Hito	Fecha	Duración
Entrega borrador del plan de trabajo	09/03/2014	
Entrega PAC1 (plan de trabajo)	11/03/2014	37h
Entrega borrador de PAC2(Memoria redactada hasta la fecha)	05/04/2014	
Entrega PAC2	08/04/2014	118h
Entrega borrador PAC3(Memoria redactada hasta la fecha)	08/05/2014	
Entrega PAC3	13/05/2014	148h
Entrega borrador PAC4(Borrador memoria)	05/06/2014	
Entrega de PAC4 (memoria definitiva)	09/06/2014	131h
Debate virtual	25/06/2014	25h
Finalización del proyecto	28/06/2014	

Tabla 1: Hitos claves del proyecto

3.2 Calendario del proyecto

A continuación se muestra el calendario general de la planificación del proyecto.

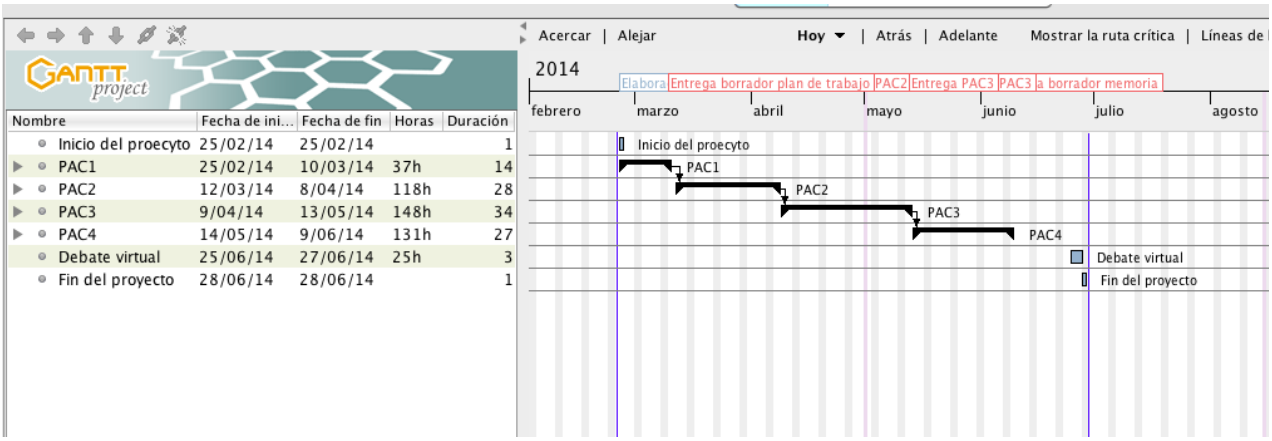


Ilustración 1: Calendario proyecto

La suma total de horas de dedicación es de 459 horas, en un total de 106 días. Ésto nos da una cifra aproximada de dedicación de unas 5h diarias.

El desglose de la planificación en cada una de las cuatro entregas el siguiente:

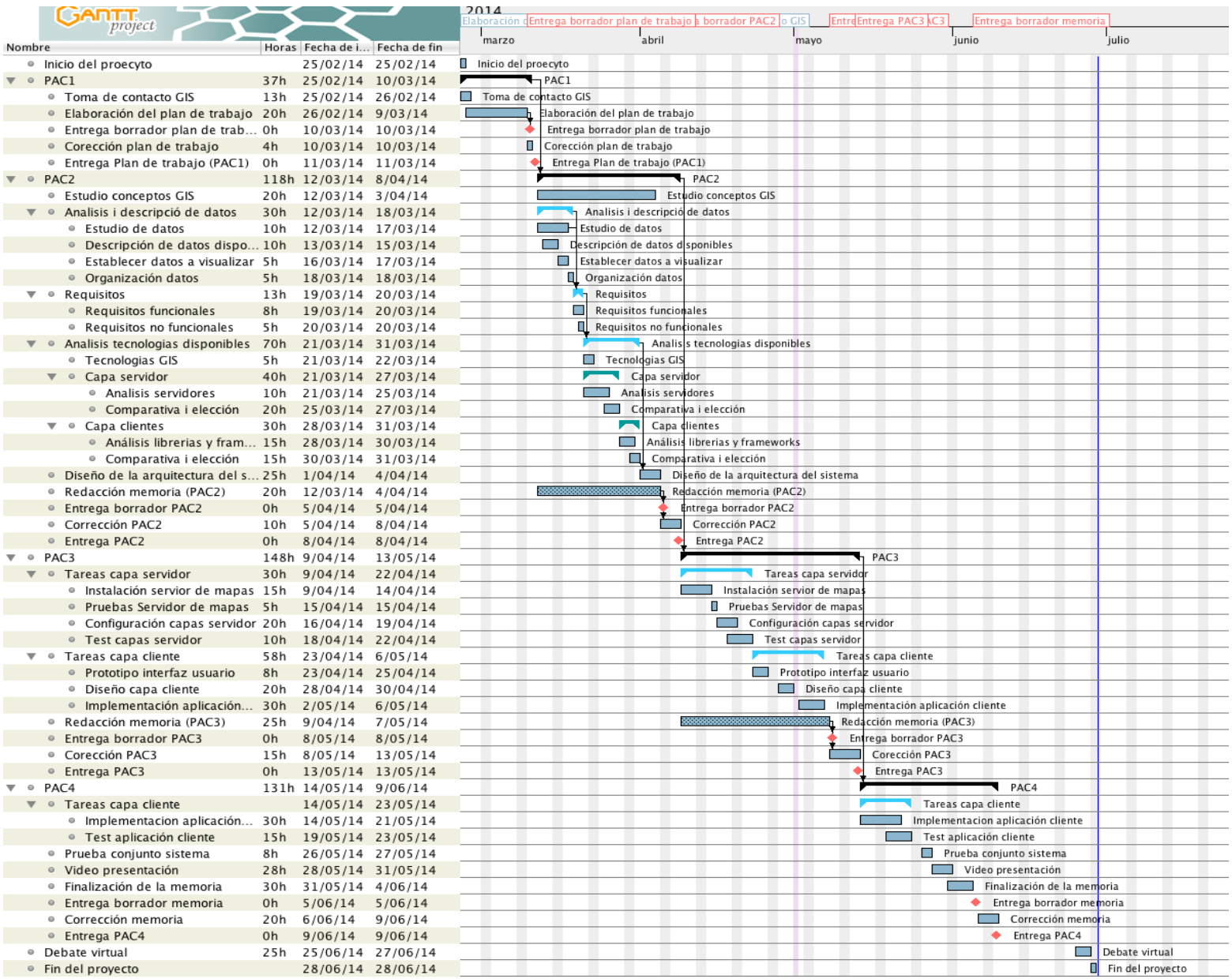


Ilustración 2: Desglose de la planificación

4 Marco teórico

4.1 ¿Que es un SIG?

En la actualidad existen muchas definiciones de lo que es un SIG (SIG o GIS en su acrónimo inglés). Una de las definiciones más extendidas es la realizada por *Burrough*, *Goodchild* entre otros:

"Conjunto integrado de medios y métodos informáticos, capaz de recoger, verificar, almacenar, gestionar, actualizar, manipular, recuperar, transformar, analizar, mostrar y transferir datos espacialmente referidos a la Tierra"

De una forma u otra, la mayoría de definiciones coinciden en que un SIG, es un conjunto de elementos integrados, y la integración de estos elementos conforman un SIG. Estos elementos son: tecnología, (tanto hardware como software) personas e información geográfica. El conjunto de todos estos elementos permite capturar, analizar, representar, transformar, almacenar, editar y representar datos georeferenciados. Por lo tanto podemos decir que un SIG permite:

- Guardar, consultar y editar datos espaciales.
- Realizar un gran número de análisis, ya sea al valor temático del dato, como al valor espacial de la información.
- Realizar operaciones y transformaciones sobre la información almacenada. Como pueden ser superposición de información, transformaciones de escalas, calculo de rutas, etc
- Generar resultados en diferentes formato, como pueden ser visualizaciones en mapas, gráficos informes, etc..

De lo comentado anteriormente podemos resumir diciendo que un SIG es un sistema que permite gestionar de una forma completa datos referenciados geográficamente. Estos datos normalmente suelen contener información geográfica, como pueden ser las coordenadas o la proyección, y también contienen la información temática de los datos, o dicho de otra forma valores alfanuméricos o descriptivos que nos indica algún tipo de información cuantificable. En el apartado "Descripción de los datos" se describen los datos que se utilizarán en el proyecto.

La mayoría de datos que se manipulan en la actualidad son datos que están localizados geográficamente. Por este motivo las tecnologías SIG, se han convertido en una tecnología muy importante en nuestro entorno. Y ésto se ha acentuado mucho más en la actualidad, debido a la aparición en los últimos años de herramientas que usan directa o indirectamente información georrferenciada. Un ejemplo, es el uso de mashups[2] en aplicaciones y páginas web, normalmente utilizan un mapa para ubicar la información.

5 Escenario

En la actualidad el SMC dispone de una gran cantidad de productos e información georeferenciada. Hasta la implantación del presente proyecto cada área muestra los productos que generan mediante diferentes herramientas, ya sea mediante pequeñas páginas web o mediante software de terceros. Lo cual hace que la información quede dispersa y lo más importante, que no se siga ningún tipo de control, ni un patrón de homogeneidad. Tampoco existe

2 Hace referencia a aplicaciones que tiene como objetivo combinar diferente tipo de información (vídeos, fotos, etc), y que posteriormente es visualizado en un entorno gráfico. http://es.wikipedia.org/wiki/Mashup_(aplicaci%C3%B3n_web_h%C3%ADbrida)

la posibilidad de superponer la información de los diferentes productos en una única herramienta. El diagrama de la situación antes de la implantación del proyecto se puede ver en la ilustración 3.

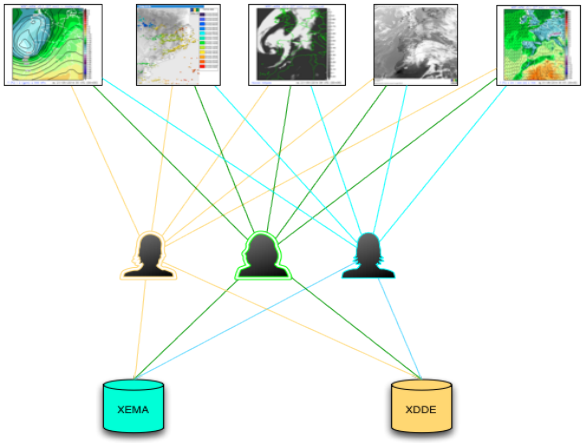


Ilustración 3: Escenario actual SMC

Mediante el sistema de visualización SIG que se propone realizar en este proyecto, los técnicos del SMC dispondrán de una única herramienta capaz de visualizar y superponer los diferentes productos del SMC. De forma que los técnicos puedan consultar en una sola herramienta dichos productos. El escenario una vez implantando el sistema se puede ver en la ilustración 4.

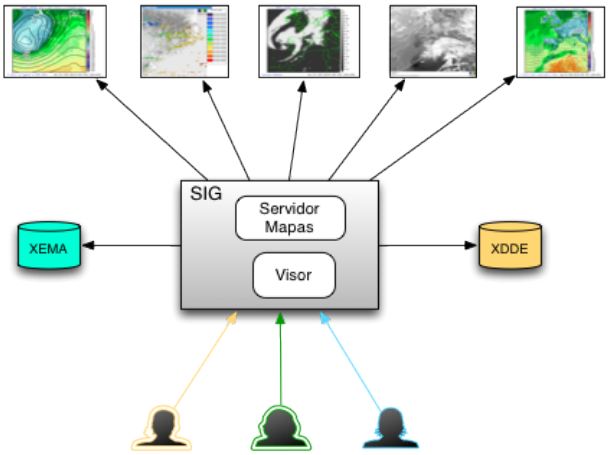


Ilustración 4: Futuro escenario SMC

6 Análisis de los datos

Para saber qué datos podemos visualizar a través del sistema SIG, primero realizaremos un análisis de la información que dispone el SMC. Durante el análisis, se describirán los diferentes datos, la organización y características más importantes. Finalmente se determinarán qué datos utilizará el sistema SIG en su primera versión.

El SMC dispone de un gran volumen de información georeferenciada que utiliza para realizar diferentes tareas, tales como realizar la predicción, realizar análisis, estudios, aplicar validaciones, correcciones, y análisis del cambio climático, entre otras actividades.

La información con la que trabaja el SMC se puede dividir en dos grupos:

- **Información procesada:** es la información que recibe el SMC de diferentes medios (como por ejemplo aplicaciones externas, radares, detectores de rayos, etc). Que una vez recibida, es procesada o tratada por el área o equipo responsable de dicha información. Una vez procesada la información se generan diferentes productos que son los que el resto de áreas consulta. Éstos productos pueden ser de precipitación, de Meteosat, modelos numéricos de predicción, etc. Éstos se describirán en apartados posteriores.
- **Información directa**: es la información que recibe el SMC, y que se insertan directamente en una base de datos relacional. Estos datos son principalmente los recibidos de las estaciones meteorológicas automáticas y observaciones por parte de la red de observadores del SMC

Se establecen una serie de condiciones para que la información pueda ser tratada de forma correcta en el sistema SIG:

- Los datos han de estar georeferenciados correctamente. Ésto incluye los metadatos, los cuales ofrecerán tanto información geográfica, como una definición de la estructura de información de los datos a los que hace referencia.
- El servidor de mapas tendrá que ser capaz de leer el formato en el que se encuentran los datos. Ésto quiere decir que los datos deben de estar en algún formato que el servidor de mapas pueda leer. En el caso que la información éste almacenada en una base de datos, el servidor ha de tener los *drivers* necesarios para acceder. Y en el caso de ficheros, estos han de estar en algún formato que el servidor sea capaz de interpretar.
- Los datos han de estar correctamente etiquetados a nivel temporal. Es decir, el servidor ha de poder identificar un dato concreto para una fecha determinada.
- Los datos han de estar accesibles para el servidor de mapas.

Para determinar qué datos se visualizarán en la primera entrega del sistema de visualización SIG, se realizan una serie de entrevistas con los técnicos de las diferentes áreas. En estas entrevistas se determinan tanto la información prioritaria para los técnicos, como el esfuerzo en recursos y tiempo que pueden invertir las áreas en adaptar los datos para que el SIG pueda explotarlos. Cruzando estos dos parámetros se determina qué datos se explotarán mediante el sistema en la primera versión son los siguientes:

- Productos de radar, básicamente de acumulación de precipitación.
- Rayos, acumulación de rayos descargados.
- Datos de estaciones meteorológicas automáticas
- Productos de Meteosat
- Modelos numéricos de predicción.

La descripción de cada uno de estos productos se hará en apartados posteriores.

6.1 Temporalidad de los datos

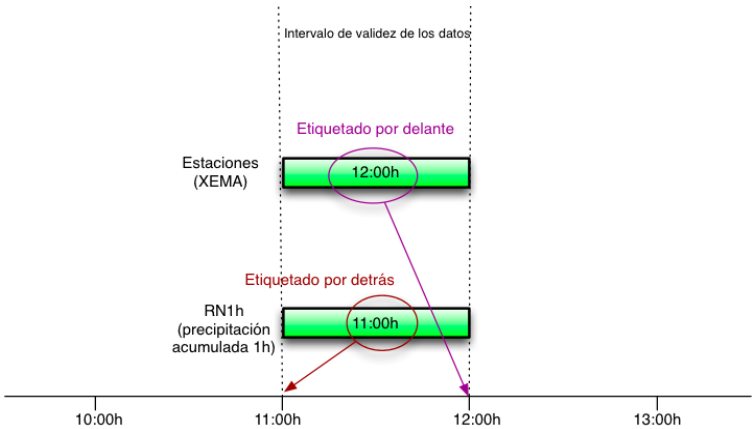
Antes de realizar el análisis de los datos, hay que aclarar una serie de conceptos en referencia a la temporalidad de los datos. Con temporalidad nos referimos al intervalo o instante de tiempo en el cual es válido un dato. Uno de los requisitos de los técnicos del SMC, es poder visualizar diferentes tipos de datos que hagan referencia a un mismo intervalo o instante de tiempo. Por ejemplo, si se visualiza una imagen de radar de un día determinado a las 12:00h, y se quiere superponer la imagen del Meteosat. Estos dos productos tienen temporalidades diferentes. Las imágenes de radar se generan cada seis minutos y las imágenes del Meteosat cada quince minutos. Por lo tanto es muy importante saber qué temporalidad tiene cada dato, para saber cómo visualizarlo. Éste es uno de los problemas que tendrá que resolver el sistema.

La mayoría de datos del SMC contienen una única fecha. Ésta fecha suele hacer referencia al momento en el que se tomó la medida, foto, predicción, etc. Con esta única fecha, no se puede determinar si el dato hace referencia a un intervalo de tiempo o un único instante, ni tampoco se puede determinar si esta fecha hace referencia al inicio o al final de un intervalo. En la actualidad el usuario tiene que saber de antemano como establecer el intervalo correcto para cada producto. Ésto no es muy práctico debido al gran volumen de información de la que dispone el SMC. El sistema de visualización SIG tendrá que realizar este cálculo para cada dato, con el objetivo de abstraer de esta tarea al usuario.

La manera de establecer el intervalo correcto para cada producto no se realiza igual para todos los productos. Ésto es debido a una mala gestión de este concepto por parte del SMC. Debido a esta falta de homogeneidad a la hora de especificar la información temporal, es muy importante saber cómo se etiquetan los datos en el tiempo. En el SMC existe un concepto para determinar la temporalidad de un dato, el etiquetado. El etiquetado hace referencia a la fecha en la que es válido el dato. El etiquetado indica si la fecha del dato determina el inicio o el final del intervalo para el cual es válido un dato. El intervalo por norma general viene dado por el tiempo que transcurre entre dato y dato. El etiquetado puede ser:

- **Por delante:** indica que la fecha etiqueta el dato por delante. Significa que la fecha del dato hace referencia al final del intervalo de validez del dato.
- **Por detrás:** indica que la fecha etiqueta el dato por detrás. Es decir, que la fecha del dato hace referencia al inicio del intervalo de validez del dato.

Ilustración 5: Representación de la temporalidad de los datos.



La ilustración 5, muestra el funcionamiento del etiquetado de los datos. Los dos rectángulos de color verde representan dos datos diferentes. La longitud del rectángulo representa el período de validez de los datos (es decir a los que hace referencia). La línea inferior horizontal representa una

linea temporal. Las horas que se especifican en cada rectángulo, representan las fecha con la que esta etiquetado el dato, y que estarán presentes o en el nombre del fichero o en un campo de la base de datos. Como se puede observar los dos datos son válidos para el mismo intervalo de tiempo, pero con la diferencia que el dato del producto RN1 tiene como etiqueta las 11:00h (etiquetado por detrás, ya que la fecha hace referencia al inicio del intervalo de validez), y el dato de estaciones tiene como etiqueta las 12:00h (etiquetado por delante, ya que la fecha con la que se etiqueta el dato hace referencia al inicio del intervalo). Éste ejemplo se puede ver en el visor en la ilustración 47.

El SMC trabaja únicamente con horaria en UTC[3], que es el estándar de tiempo que se suele utilizar en el ámbito meteorológico.

Éstos conceptos referentes a la validez temporal de los datos son muy importantes a la hora de consultar los datos mediante el visor SIG. En adelante se seguirán utilizando los conceptos descritos en este apartado.

6.2 Descripción de los datos

En el siguiente apartado se describe la información que potencialmente interesa visualizar mediante el sistema. El SMC posee otro tipo de información, como pueden ser las predicciones comarcales y municipales o la red de observadores meteorológicos. Pero esta información, a día de hoy no es posible añadirla en el sistema, ya que no poseen una geolocalización correcta, o el formato de almacenamiento no es el idóneo para que el servidor de mapas los pueda visualizar.

6.2.1 Radar meteorológico

Un radar meteorológico se utiliza para localizar precipitaciones y calcular su trayectoria. Básicamente funciona de la misma forma que cualquier otro radar. La antena del radar emite una serie de señales que al encontrarse con el blanco(partículas de precipitación, que pueden ser agua, nieve, granizo o la una combinación de todos ellos) hacen rebotar la señal (eco) que es recibida de nuevo por el radar. La señal devuelta, es la que el radar analiza y procesa, y puede determinar la cantidad de precipitación y su movimiento.

El SMC dispone de cuatro radares meteorológicos distribuidos en diferentes zonas de Cataluña. La red de radares meteorológicos del SMC se denomina XRAD[4]. Los radares envían la información a un servidor en el SMC. Un software especializado genera una serie de productos relacionados con la precipitación. Estos productos son procesados en el SMC, y se genera una serie de productos que finalmente serán los utilizados por los técnicos del SMC.

3 UTC, Universal Time Coordinated. Principal estándar de tiempo, que se sincroniza con el tiempo medio de Greenwich. http://aa.usno.navy.mil/faq/docs/UT.php

4 Siglas de “Xarxa de Radars”, red de detectores de rayos del Servei Meteorològic de Catalunya. Http://www.meteo.cat

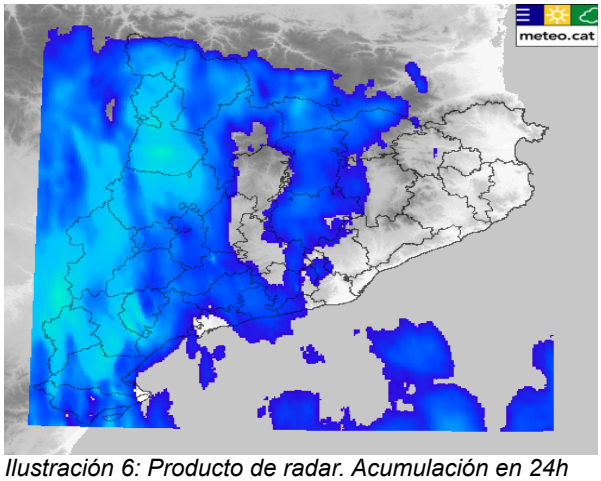


Ilustración 6: Producto de radar. Acumulación en 24h

Imagen obtenida de productos generados por el SMC

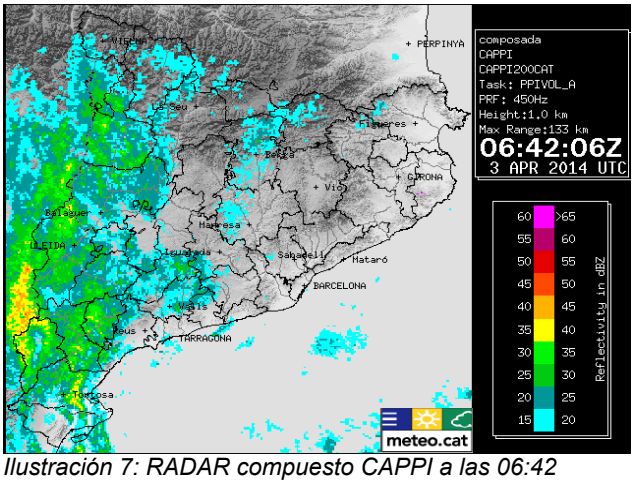


Ilustración 7: RADAR compuesto CAPPI a las 06:42

Imagen obtenida de productos generados por el SMC

Productos radar

A continuación se describen los productos de radar que serán gestionados por el sistema SIG.

Propiedades comunes a todos los productos de radar:

- **Formato**: Fichero, en formato GeoTIFF.
- **Etiquetado**: por detrás
- **Tipo de dato**: Raster
- **Repositorio de datos**: Sistema de ficheros

Producto	Descripción	Proyección	Temporalidad	Formato
CAPPI 250m	Imagen de radar a 250 metros compuesta(de los tres radares) . Mide la reflectividad de las nubes.	epsg:25831	6 minutos	geoTIFF
RAIN1H_COMP_CE	Acumulación horaria compuesta y corregida EHIMI	epsg:25831	1 hora	geoTIFF
RAIN1H_COMP_CEXEMA	Acumulación de lluvia horaria, corregida con datos de estaciones	epsg:25831	1hora	geoTIFF
RAIN24H_COMP_CEXEMA	Acumulación de lluvia diaria y corregida con datos de estaciones	epsg:25831	24h	geoTIFF

Tabla 2: Descripción productos de radar

6.2.2 Rayos

El SMC dispone de una base de datos con la información de las descargas eléctricas que se generan en territorio catalán. Los rayos se detectan a partir de un sistema de detección de rayos que detectan las emisiones electromagnéticas generadas por las tormentas.

El SMC dispone de una red de tres detectores de tipo VHF[5], denominada XDDE[6]. Mediante estos

5 (VHF)Very High Frequancy, frecuencia utilizadas para detectar descargas nube-tierra y nube-nube. www.meteo.cat

6 Xarxa de Detecció de Descàrregues Elèctriques. www.meteo.cat

detectores se localizan las descargas mediante triangulación. Una vez recopilada la información, ésta es enviada mediante radio enlace a unos servidores ubicados en el SMC. Ha esta información se le aplica una serie de procesos que finalmente insertan en una base de datos relacional PostgreSQL, que tiene instalado el módulo PostGIS. El módulo PostGIS hace que la base de datos relacional, se convierta en una base de datos espacial, capaz de gestionar y almacenar datos espaciales. Esta base de datos es la que los técnicos del SMC explotan.

Producto rayos

Los rayos representan las descargas producidas en un instante concreto, esto significa que no se le puede aplicar una durabilidad temporal. Las descargas eléctricas se consultan realizando una acumulación temporal, es decir, cantidad de rayos descargados en un intervalo de tiempo determinado, por ejemplo la cantidad de rayos descargados en seis minutos. Por lo tanto, se establece que las consultas se realizarán en un intervalo determinado. Los intervalos que se establecen para realizar las consultas son: 6, 15, 30, 60, 120 y 240 minutos. De esta forma, se podrá establecer un intervalo temporal de validez a los datos consultados. Por ejemplo, si se realiza una consulta con una acumulación de 6 minutos para la hora 12:00h, la consulta nos devolvería todas las descargas que se hayan producido entre las 11:54 i las 12:00, ambos incluido.

Otro tipo de consulta sobre las descargas eléctricas, es realizar consultas de descargas acumuladas en un día, agrupando las descargas en 12 intervalos. En la ilustración 49 se puede ver este ejemplo aplicado en el visor.

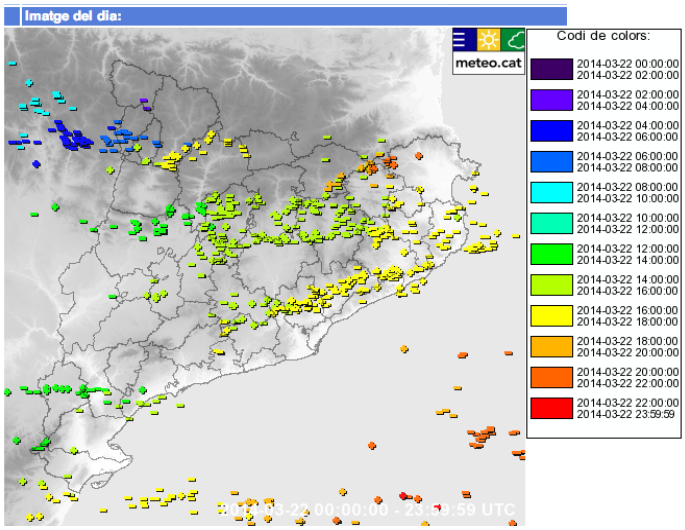


Ilustración 8: Imagen producto de descarga eléctrica en 12 intervalos

Imagen obtenida del SMC

En la ilustración 8, podemos ver un ejemplo de un producto de rayos que actualmente genera el SMC. En el se puede ver la localización de las diferentes descargas, donde el color de cada descarga representa el periodo en el cual se ha producido la descarga. Los intervalos de tiempo se representan mediante 12 periodos, donde cada periodo tiene una duración de dos horas.

Propiedades comunes a todos los productos de rayos:

- **Tipo de dato:** Vectorial
- **Repositorio de datos:** Base de datos PostgreSQL
- **Etiquetado**: por delante.

Producto	Temporalidad	Tipo	Polaridad
Acumulación	• 6 minutos • 15 minutos • 30 minutos • 60 minutos • 120 minutos • 240 minutos	nube-tierra	Positivos Negativos
Diario	24 horas	nube-tierra	Positivos Negativos

Tabla 3: Productos de rayos

6.2.3 XEMA

El SMC dispone de una red de estaciones meteorológicas automáticas(en adelante XEMA[7]) distribuidas por Cataluña. Las estaciones disponen de una serie de sensores meteorológicos que miden una serie de variables(radiación solar, varios tipos de temperaturas, velocidad y dirección del viento a diferentes alturas, humedad relativa, presión atmosférica, precipitación, etc.).

Las estaciones disponen de unos sensores que miden las diferentes variables. Cada cierto tiempo envía los datos mediante GPRS a los servidores del SMC. Una vez recibida la información, en los servidores del SMC se ejecutan una serie de procesos que insertan la información en una base de datos relacional(en adelante base de datos XEMA). Ésta base de datos es la que explotan los técnicos del SMC.

A partir de la información de la base de datos XEMA, se realizan múltiples estudios y análisis, sobre todo a nivel climático. Como por ejemplo atlas climáticos, cálculos temporales sobre los valores(como por ejemplo estadísticos diarios, mensuales o anuales), estudios sobre el cambio climático, etc. En este proyecto los datos que interesan son las lecturas que realizan las estaciones de las diferentes variables. En apartados posteriores se especificará la información que será visualizada.

Producto XEMA

Cada lectura almacenada en la base de datos XEMA, está informada con una fecha, indicando la fecha en la que se tomó la lectura. Por otro lado tenemos la base horaria de la estación. La base horaria indica cada cuanto tiempo genera una lectura una estación. Teniendo en cuenta esta base horaria, se clasifican las estaciones en dos grupos.

- **Semi-horarias**: estaciones que generan una lectura cada treinta minutos.
- **Horarias**: estaciones que genera una lectura cada hora.

En ésta primera versión del sistema se mostrarán las estaciones de forma horaria. Ésto implica realizar una agregación (de los dos valores generados en una hora) en las estaciones semi-horarias. La manera de agregar los datos dependerá de la variable, para la variable precipitación se realiza una suma de las dos lecturas realizadas en una hora. Para el resto de variables la agregación se realiza mediante una media.

7 Xarxa d'Estacions Meteorològuiques Automátiques http://meteo.cat/servmet/info/xemec.html

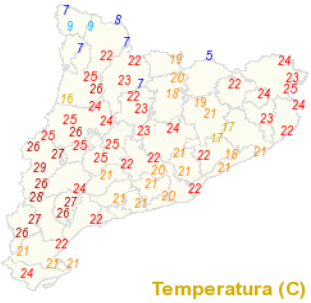


Ilustración 9: Temperatura a partir de XEMA

Imagen obtenida del SMC

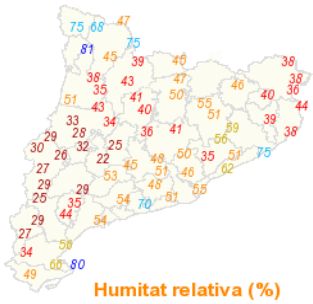


Ilustración 10: Humedad relativa a partir de XEMA

Imagen obtenida del SMC

Propiedades comunes:

- **Tipo de dato:** Vectorial
- **Repositorio de datos:** Base de datos Oracle
- **Proyección**:
- **Temporalidad**: dependiendo de la estación, puede ser cada 30 minutos o cada 60 minutos.
- **Etiquetado**: por delante

Producto	Variable	Descripción
Agregado horario	Temperatura	Temperatura media en una hora.
	Temperatura máxima	Media temperatura máxima en una hora.
	Temperatura mínima	Media temperatura mínima en una hora.
	Humedad relativa	Media humedad relativa en una hora.
	Presión atmosférica	Media presión atmosférica en una hora
	Precipitación acumulada	Precipitación acumulada en una hora.
	Irradiancia solar global	Irradiancia solar global en una hora.
Estadístico diario	Temperatura media diaria	Temperatura media diaria.
	Temperatura máxima diaria	Media temperatura máxima diaria.
	Temperatura mínima diaria	Media temperatura mínima diaria.
	Humedad relativa media diaria	Media humedad relativa diaria.
	Presión atmosférica media diaria	Media presión atmosférica diaria.
	Precipitación acumulada diaria	Precipitación media acumulada en un día.
	Irradiación solar global diaria	Irradiación solar global en un día.
Estadístico mensual	Temperatura media mensual	Temperatura media mensual.
	Temperatura máxima absoluta mensual	Media temperatura máxima mensual.
	Temperatura mínima absoluta mensual	Media temperatura mínima mensual.
	Humedad relativa media mensual	Media humedad relativa mensual.

Producto	Variable	Descripción
	Presión atmosférica media mensual	Media presión atmosférica mensual.
	Precipitación acumulada mensual	Precipitación acumulada en un mes
	Media mensual irradiación solar	Media irradiación solar mensual.

Tabla 4: Lista de productos de estaciones meteorológicas automáticas

6.2.4 Meteosat

Los satélites de Meteosat son un grupo de satélites meteorológicos construidos y lanzados por la ESA[8]. El principal propósito de un satélite Meteosat es el de medir las radiaciones térmicas y visibles en distintas bandas espectrales. Cubre principalmente el territorio europeo y parte de África. El Meteosat genera una serie de imágenes denominadas visible, infrarroja térmica e infrarroja de vapor de agua.

Los datos que recibe el SMC del Meteosat son datos en bruto. A estos datos se le han de aplicar una serie de correcciones y calibraciones antes de que puedan ser interpretadas y generar productos útiles.

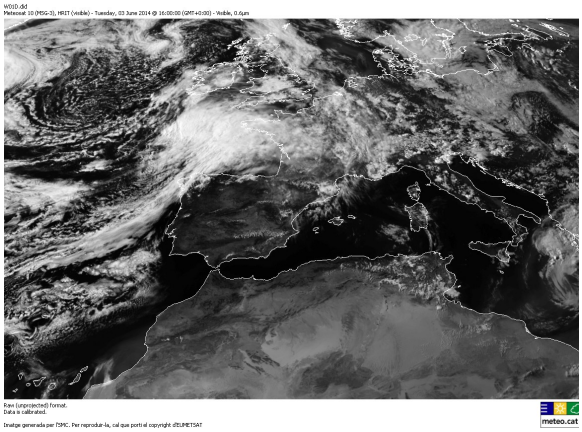


Ilustración 11: Meteosat, canal visible

www.meteo.cat

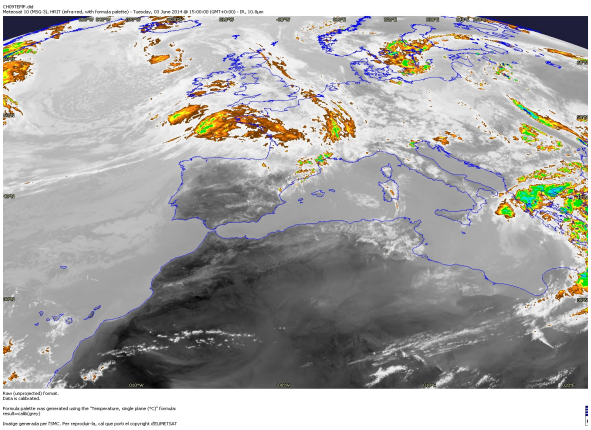


Ilustración 12: Meteosat, canal IR

www.meteo.cat

Productos Meteosat

A partir de los datos del satélite de Meteosat, el SMC genera una serie de productos. Los productos de Meteosat que se añadirán al sistema de visualización se genera cada 15 minutos. El producto es un único fichero GeoTIFF, que contienen diferentes bandas. Cada una de estas bandas es un producto distinto.

Producto	Descripción	Proyección	Temporalidad	Formato
HRIT_MSG3	Infrarrojo térmico: Muestra las emisiones terrestres en el rango térmico del espectro electromagnético	proj=merc ; lon_0=-0; lat_ts=44; x_0=0; y_0=0; ellps=WGS84	15 minutos	geoTIFF

8 Agencia Espacial Europea http://www.esa.int/ESA

Producto	Descripción	Proyección	Temporalidad	Formato
MSG3_VIS_006	Visible: Reproduce el grado de reflexión de la luz solar incidente de los diversos tipos de cobertura terrestre o de las nubes que tienen por encima	proj=merc ; lon_0=-0; lat_ts=44; x_0=0; y_0=0; ellps=WGS84	15 minutos	geoTIFF
MSG3_WV_062	Vapor de agua: Reproduce el registro de banda del espectro de 6,2 µm, banda donde hay una fuerte absorción del vapor de agua atmosférico.	proj=merc ; lon_0=-0; lat_ts=44; x_0=0; y_0=0; ellps=WGS84	15 minutos	geoTIFF

Tabla 5: Productos derivados del satélite de Meteosat

6.2.5 Modelos numéricos

Los modelos numéricos de predicción, consisten básicamente en simular la evolución futura de variables meteorológicas a partir de modelos matemáticos. Estos modelos matemáticos compuestos por ecuaciones complejas, se ejecutan únicamente en ciertos puntos de una determinada extensión. Así se reduce el tiempo de computación. Para poder realizar las futuras predicciones, los modelos suelen utilizar unos datos de entrada, que utilizan para determinar el estado actual a nivel meteorológico. Existen diferentes tipos de modelos numéricos, que difieren tanto en el conjunto de formulas matemáticas que los generan, como en el área geográfica a la que se aplican o hacen referencia.

Los modelos del SMC se ejecutan una o varias veces al día(dependiendo del modelo, pero suelen ser una, dos, cuatro u ocho veces al día). Cada modelo hace referencia a predicciones en diferentes ventanas de tiempo. Es decir, para cada ejecución de un modelo, se generan varias predicciones válidas para un intervalo de tiempo concreto.

Los modelos numéricos de predicciones meteorológicas poseen una particularidad que no poseen el resto de datos. Cuando un modelo se ejecuta, realiza una serie de predicciones con un intervalo de varias horas entre predicción y predicción, hasta llegar a su alcance máximo. Por ejemplo, si el intervalo de predicción de un modelo es de 3 horas y su alcance máximo es de 72 horas, genera 25 predicciones, 72/3 = 24 predicciones, más la predicción a las 0h. Éste ejemplo esta representado en la ilustración 13.

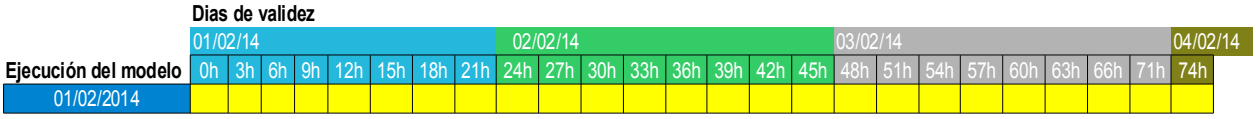


Ilustración 13: Ejemplo ejecución de un modelo

Los recuadros de color amarillo representan las diferentes predicciones generadas por un modelo en su ejecución del día 01/02/2014.

El hecho que para cada ejecución del modelo se generen varias predicciones, hace que para una fecha determinada, existan varias predicciones. Éste caso queda reflejado en la ilustración 14.

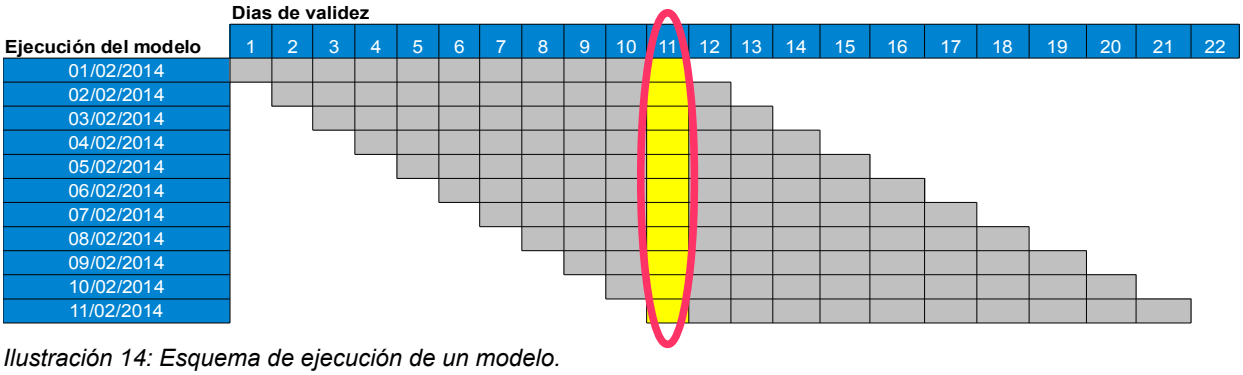


Ilustración 14: Esquema de ejecución de un modelo.

Productos de modelos

Los modelos que genera el SMC, son ficheros en formato GRIB[9]. La información almacenada en estos ficheros está agrupada en diferentes bandas. Cada una de estas bandas contiene la información de una predicción para una variable a cierta altura.

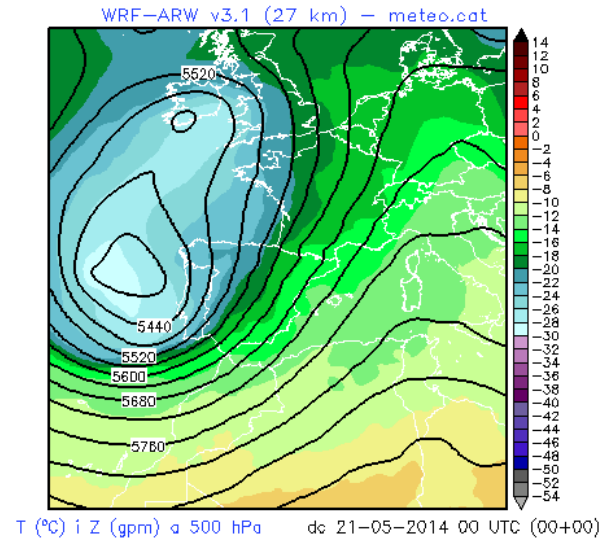


Ilustración 15: Modelo WRF27 altura geopotencial

http://www.meteo.cat

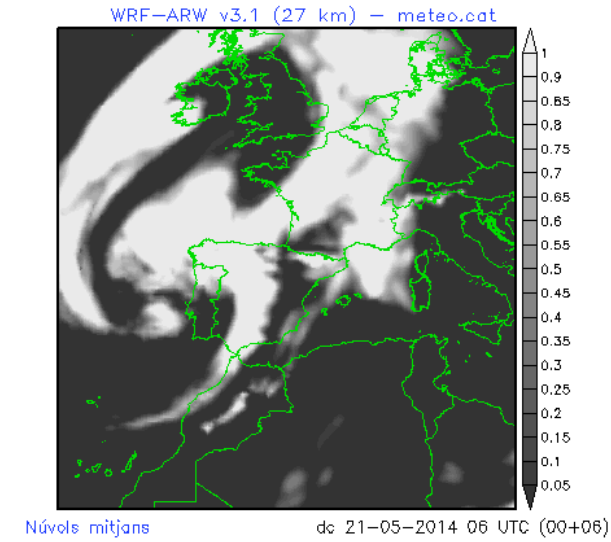


Ilustración 16: Modelo WRF27 nubosidad media

http://www.meteo.cat

Los modelos que se utilizarán en la primera versión del sistema son los siguientes:

Prescat

Modelo de predicción desarrollado por el SMC, que realiza predicciones probabilistas en el territorio catalán.Basado en analogías entre situaciones pasadas y futuras. Sus características son las siguientes:

- **Número de salidas**: 1
- **Alcance**: +11 días
- **Intervalo entre predicciones**: 24h

9 Fichero estándar(World Meteorological Organizations Comission) que se utiliza normalmente para almacenar datos meteorológicos, tanto datos históricos como predicciones. http://en.wikipedia.org/wiki/GRIB

Variable	Nivel(altitud)	Descripción
Probabilidad de precipitación > 0.5mm	Superficie	Predicción de la variable probabilidad de precipitación en el territorio catalán.
Temperatura máxima	Superficie	Predicción de temperaturas máximas en territorio catalán.
Temperatura mínima	Superficie	Predicción de temperaturas máximas en territorio catalán.
Estado del cielo	Superficie	Predicción del estado del cielo en territorio catalán.

Tabla 6: Productos del modelo Prescat

WRF27

Modelo de área limitada diseñado para aplicaciones operativas y de investigación. Se puede utilizar en un rango amplio de escalas (altitudes).

- **Número de salidas**: a las 00h / a las 12h cada día
- **Alcance**: +72horas
- **Intervalo**: 3 horas
- **Dominio 27:** Km

Variable	Nivel(altitud)	Descripción
Temperatura	2 metros	Predicción de la temperatura a una altura de 2 metros.
	500 hPa	Predicción de la temperatura en alturas donde la presión es de 500hPa.
	850 hPa	Predicción de la temperatura en alturas donde la presión es de 850hPa.
Humedad relativa	2 metros	Predicción de la humedad relativa a una altura de 2 metros.
	500 hPa	Predicción de la humedad relativa en alturas donde la presión es de 500hPa.
	850 hPa	Predicción de la humead relativa en alturas donde la presión es de 850hPa.
Velocidad i dirección del viento	2 metros	Predicción de la velocidad y dirección del viento a una altura de 2m sobre la superficie terrestre.
	500 hPa	Predicción de la velocidad y dirección del viento a una altura donde la presión es de 500hPa.
	850 hPa	Predicción de la velocidad y dirección del viento a una altura donde la presión es de 500hPa.
Nubosidad		Predicción de nubes medias.
Altura geopotencial	500 hPa	Predicción de la altura geopotencial en alturas donde la presión es de 500hPa
	850 hPa	Predicción de la altura geopotencial en alturas donde la presión es de 850hPa
Presión reducida a nivel del mar	Nivel del mar	Predicción de la variable presión reducida al nivel del mar.

Tabla 7: Productos del modelo WRF27

6.2.6 Interpolaciones

Productos que contienen información interpolada de ciertas variables. La interpolación requiere de un calculo costoso, dado que tienen en cuenta ciertos factores como altitud o la cercanía al mar. En la actualidad el SMC dispone de dos productos interpolados, para las variables temperatura y humedad. Se generan cada 30 minutos, utilizando los últimos datos de la XEMA.

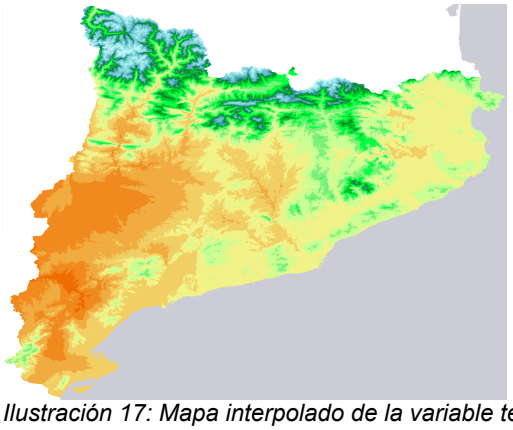


Ilustración 17: Mapa interpolado de la variable temperatura

Imagen obtenida del SMC

Productos interpolados

Variable	Descripción	Temporalidad
Temperatura	Interpolación de la variable temperatura..	30 minutos
Humedad relativa	Interpolación de la variable humedad relativa	30 minutos

Tabla 8: Productos interpolados

6.3 Mapas base

A parte de los productos generados por el SMC, se necesitaran mapas base para el visor. Se configuran diferentes mapas base para para ofrecer diferentes tipos de fondo.

6.4 Organización de la información

Para que los datos puedan ser explotados mediante el servidor de mapas, los datos han de estar organizados, ya sea en una base de datos o en una estructura de ficheros. Utilizando una nomenclatura estándar para los nombres de los ficheros(productos).

Bases de datos

Las descargas eléctricas se consultarán directamente a la base de datos PostgreSQL, y se utilizarán las funciones proporcionadas por el módulo PostGIS para realizar las consultas. Mediante estas consultas, se recogerá la información en forma vectorial(aunque el servidor de mapas la sirva en forma de imagen).

API REST

Para acceder a la información de la XEMA, se utilizara una API REST. Esta API tiene como principal objetivo ser un punto de consulta único para ciertos datos del SMC. La API consulta

directamente la base de datos de estaciones. La consulta a la API se realizará directamente desde la aplicación cliente, que mostrará los datos de las estaciones de forma vectorial.

Sistema de ficheros

El sistema de ficheros es donde se almacenarán los productos generados por el SMC. El servidor de mapas utilizará los ficheros para mostrar la información de los productos. La organización de los ficheros ha de seguir un estándar definido por el SMC. La organización se realiza mediante una estructura de carpetas, donde los primeros niveles indicarán el producto(nombre del producto, altitud, etc.), y a partir de aquí los siguientes niveles hacen referencia a la fecha. Indicando en diferentes directorios el año, mes, y día. El nombre del fichero tendrá un parte fija que identifica el fichero, y otra parte variable que hace referencia a la fecha. Un ejemplo de esta organización se puede ver en la ilustración 18.

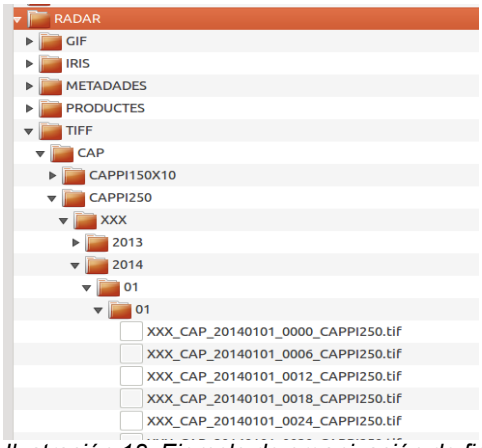


Ilustración 18: Ejemplo de organización de ficheros

A continuación se muestra el esquema del acceso a los diferentes repositorios de datos que realiza el conjunto del sistema SIG. Como se puede ver en la ilustración 19, el servidor de mapas accederá a la base de datos PostgreSQL que contiene la información de las descargas eléctricas, y al sistema de ficheros donde estarán los ficheros de los diferentes productos. La aplicación cliente consultará directamente a la API REST del SMC para acceder a los datos de la XEMA.

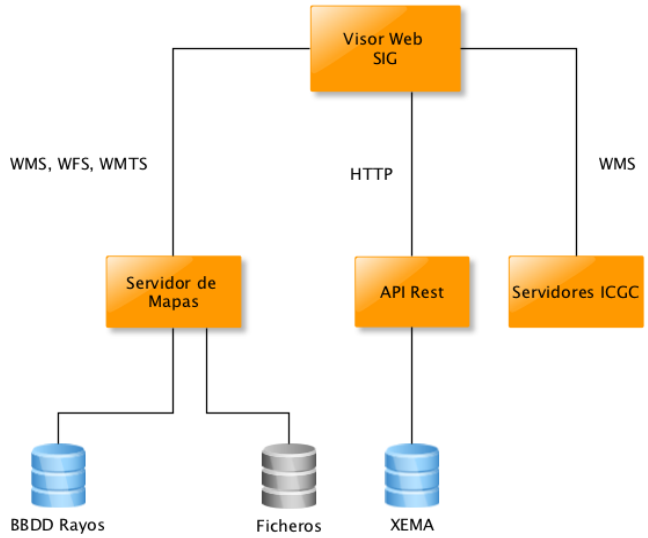


Ilustración 19: Arquitectura del sistema para el acceso a la información

7 Requisitos del sistema

En éste apartado se realiza una descripción de los requisitos del sistema. Se realiza una descripción tanto de los requisitos funcionales, como de los no funcionales. Los requisitos se describen en una lista, donde la primera columna hace referencia al identificador único del requiso y la segunda columna corresponde a la descripción del mismo. El identificador de cada requisito se compone de tres elementos, el primero indica el tipo de requisito, el segundo indica a qué capa pertenece y el tercero un número secuencial. De esta forma se podrá hacer referencia fácilmente al requisito en todo momento.

7.1 Requisitos no funcionales

A continuación se describen los requisitos no funcionales del sistema. Los requisitos no funcionales describen las características y necesidades del sistema, que no hagan referencia a funcionalidades propias del mismo.

A continuación se describen los requisitos no funcionales en dos apartados, servidor y cliente.Servidor

Identificador	Descripción
RNF-S-01	El servidor de mapas tendrá en todo momento acceso a todos los orígenes de datos que estén configurados como capas en dicho servidor. Por lo tanto tendrá permisos de lectura sobre los distintos repositorios en los cuales residan los datos a consultar, ésto incluye tanto bases de datos, como repositorio de ficheros.
RNF-S-02	El servidor de mapas contará con un sistema de caché que permitirá almacenar durante un tiempo determinado la información más consultada por los usuarios. De esta forma se reducirá el tiempo de respuesta.
RNF-S-03	Se crearán un conjunto de procesos en el servidor de mapas, que se ejecutarán periódicamente, con el objetivo de pregenerar los datos más relevantes y almacenarlos en el sistema de caché. De esta forma no hará falta que un usuario haya consultado un dato para que éste sea cacheado.
RNF-S-04	El servidor de mapas será accesible vía HTTP dentro de la red del SMC. Para que todos los usuarios del SMC puedan consultar el servidor desde sus máquinas de trabajo.
RNF-S-05	En la máquina que alberga el servidor, se creará un proceso que elimine periódicamente información almacenada en caché, eliminando todo fichero cuya información haga referencia a más de dos días de antigüedad.
RNF-S-06	El servidor deberá responder a las peticiones en un tiempo máximo de 8 segundos.
RNF-S-07	El servidor estará ubicado en una máquina virtual con sistema operativo linux CentOS

Tabla 9: Requisitos no funcionales de la capa servidor

7.1.1 Cliente

Identificador	Descripción
RNF-C-01	El visor SIG tendrá que ser accesible desde un navegador Web. Sin necesidad de instalar ningún tipo de componente o plugin externo. Ésto significa que para la ejecución del visor no será necesario instalar ningún tipo de software en la máquina del usuario, a excepción del navegador Web,que ya tienen instalados todos los usuarios de la empresa.
RNF-C-02	El visor debe poder visualizarse correctamente en Firefox, Google Chrome.
RNF-C-03	El visor es una herramienta únicamente de consulta, por lo tanto no requerirá de un sistema de registro de usuarios.
RNF-C-04	Resolución de pantalla mínima de 1024 X 768 pixels
RNF-C-05	La aplicación cliente ha de tener acceso al servidor de mapas vía HTTP.
RNF-C-06	Cualquier técnico del SMC debe poder utilizar de forma ágil y sencilla la aplicación tras haber recibido una breve formación.
RNF-C-07	Los conceptos, variables y campos utilizados en la aplicación deben de ser familiares a los técnicos del SMC. Ésto facilitará el uso de la interfaz por parte de los técnicos.
RNF-C-08	El visor deberá poder trabajar y visualizar datos raster y vectoriales

Tabla 10: Requisitos no funcionales de la capa cliente

7.2 Requisitos funcionales

En el siguiente apartado se describen los requisitos funcionales del sistema. Los requisitos funcionales se han recopilado a partir de una serie de entrevistas con los técnicos del SMC. En esta primera versión del sistema se han seleccionado únicamente los requisitos y funcionalidades comunes y genéricas a todas las áreas.

Como en el apartado anterior se han diferenciado los requisitos para las dos capas del sistema.

7.2.1 Servidor

Identificador	Descripción
RF-S-01	Las consultas al servidor de mapas se realizarán a través de protocolos estándar establecidos por el OGC[10]: WMTS y WMS. Los protocolos de consulta se ampliarán en posteriores versiones del sistema según lo requieran los requisitos.
RF-S-02	El servidor de mapas dará servicio no solo al visor Web SIG, sino que también tendrá que permitir el acceso a cualquier otro software que pueda realizar consultas mediante los protocolos estándar establecidos por el sistema. Es decir que cualquier técnico del SMC podrá consultar el servidor de mapas a través de otros clientes SIG como puede ser QGIS.

Tabla 11: Requisitos funcionales de la capa servidor

10 Open Geospatial Consortium, es un consorcio internacional de alrededor de 477 empresas, que participan en el proceso de desarrollar interfaces y servicios estándar en el campo SIG. www.opengeospatial.org/

7.2.2 Cliente

Identificador	Descripción
RF-C-01	El usuario podrá añadir capas a la visualización mediante un formulario. Éste formulario contendrá diferentes campos que permitirá al usuario especificar qué tipo de información quiere visualizar y cómo la quiere visualizar. Dentro de estos campos se incluye el campo que permitirá definir la fecha a visualizar.
RF-C-02	El usuario podrá eliminar cualquier capa cargada en la aplicación.
RF-C-03	La aplicación dispondrá de un panel flotante para la gestión de capas. Éste panel flotante se podrá mostrar u ocultar a voluntad del usuario.
RF-C-04	La aplicación permitirá superponer diferentes capas de información en el visor.
RF-C-05	La aplicación permitirá al usuario variar el orden de visualización de las capas, utilizando una lista de capas que actuara como una pila. Donde la capa que se encuentre al inicio de la lista es la que se visualizará en primer lugar, la segunda en segundo lugar, y así sucesivamente.
RF-C-06	La aplicación permitirá mostrar y ocultar la información de las diferentes capas cargadas.
RF-C-07	La aplicación permitirá realizar animaciones temporales de datos. Para ello se utilizará una ventana flotante, que contendrá una serie de controles para realizar la animación. Permitiendo especificar el intervalo de tiempo a visualizar, realizar una reproducción automática, especificar la velocidad de animación y establecer la medida de tiempo entre cada intervalo.
RF-C-08	La aplicación ofrecerá una funcionalidad para poder obtener información de un punto del mapa de las capas cargadas. Es decir, el usuario podrá hacer click en cualquier punto del mapa, y obtener el valor en ese punto para las capas que se estén visualizando en ese momento.
RF-C-09	La aplicación permitirá al usuario realizar *zooms*. Pudiendo acercar o alejar el campo de visión mediante botones o especificando una zona del mapa.
RF-C-10	La aplicación permitirá seleccionar varios mapas de fondo.
RF-C-11	La aplicación permitirá realizar modificaciones de las propiedades de visualización. • Cambio del estilo de la capa • Cambio de la opacidad de la capa.
RF-C-12	La aplicación permitirá navegar por una extensión fija del mapa que englobará toda la superficie terrestre.
RF-C-13	La aplicación permitirá mostrar y ocultar el panel flotante que permite realizar la animación.
RF-C-14	La aplicación permitirá realizar un zoom a la extensión máxima de la capa. Esta acción permitirá ver al usuario la totalidad de la extensión que cubre la capa.
RF-C-15	El visor trabajará con horario UTC, dado que es el sistema con el se trabaja en el SMC

Tabla 12: Requisitos funcionales de la capa cliente

7.3 Diagrama de casos de uso

A continuación se representan las funcionalidades del visor Web SIG, mediante un diagrama de casos de uso.

Ilustración 20: Diagrama de casos de uso

8 Tecnologías disponibles

A continuación se describen y analizan las diferentes tecnologías disponibles para el desarrollo e implantación del proyecto.

8.1 Arquitecturas SIG

Para el desarrollo de un sistema SIG, disponemos básicamente de dos tipos de arquitecturas: arquitectura de escritorio y arquitectura cliente-servidor web. La arquitectura de escritorio, es una arquitectura donde todo el software se ejecuta en la máquina del usuario/cliente. La arquitectura cliente-servidor web está compuesta por una aplicación cliente web(adonde acceden los usuarios) y un servidor que recibe y resuelve las peticiones del cliente ejecutando los cálculos y procesos necesarios. A continuación se describen las ventajas e inconvenientes de las dos arquitecturas.

Arquitectura aplicaciones de escritorio

Ventajas	Inconvenientes
Mejor aprovechamiento del hardware y del software de la máquina del usuario.	Requiere de la instalación de todo el paquete en la máquina del cliente.
Mejor tiempo de respuesta.	El mantenimiento es más costoso y tedioso, dado que cada actualización se debe instalar en cada una de las máquinas de los usuarios.
No depende de internet o de conexiones con servidores externos.	Normalmente suele requerir algún tipo de software de terceros para su funcionamiento.
No depende de otro software para funcionar, como por ejemplo un navegador.	Al actualizar el sistema operativo del usuario, se corre el riesgo de un mal funcionamiento de la aplicación, o que directamente no funcione.
	La información queda descentralizada, dado que cada usuario tiene instalada la aplicación en su máquina y almacenará la información en su propia máquina.
	Como todo el cálculo de la aplicación se realiza en la máquina del cliente, ésta ha de tener una serie de especificaciones concretas y hace que el coste del sistema aumente.

Tabla 13: Ventajas e inconvenientes aplicaciones escritorio

Arquitectura aplicaciones cliente-servidor Web

Ventajas	Inconvenientes
Todo el cálculo del sistema lo realiza el servidor, y por lo tanto será la única máquina que requerirá actualizaciones o mejoras de componentes.	Peores tiempos de respuesta que las aplicaciones de escritorio.
Escalabilidad del sistema, ya que el servidor se puede ir mejorando en lo que a componentes se refiere, o incluso se pueden crear clusters de servidores para aumentar el rendimiento del sistema. Sistema compuesto a base de componentes.	Dependen de los navegadores web, y ésto puede ser un inconveniente dado que cada navegador interpreta las páginas web de forma distinta en ciertos casos.
A parte del navegador web, que hoy en día todos los usuarios tienen, no requiere de ningún otro software.	Si el servidor cae, todo el sistema deja de funcionar.
Toda la información está centralizada.	Congestión de peticiones en el servidor.
La distribución de las actualizaciones es fácil y poco costosa.	Los lenguajes interpretados que se utilizan para el desarrollo de las aplicaciones web tienen un peor rendimiento.
Las máquinas de los clientes no han de tener grandes especificaciones, lo cual reduce el coste del sistema.	
La corrección de posibles errores se centraliza en el servidor o en el cliente web.	
El sistema es más "*portable"*.	
Todos los usuarios con conexión a Internet/Intranet pueden acceder al sistema.	

Tabla 14: Ventajas e inconvenientes de la arquitectura cliente-servidor

Una vez analizadas las dos arquitecturas más populares utilizadas en sistema SIG, se ha optado por realizar el sistema utilizando la **arquitectura cliente-servidor web**. Los motivos para decantarse por esta arquitectura, son básicamente a nivel de costes, mantenimiento y establecer un servidor de mapas que podrá ser consultado por cualquier cliente. En referencia al mantenimiento, se ha tenido muy en cuenta la mejor distribución de las versiones. Otro aspecto a tener en cuenta a nivel de ahorro de costes, es el hecho de no tener que ampliar las especificaciones de las máquinas de los usuarios. También se valora el tener bien controlado todo el sistema. El problema del rendimiento y del tiempo de respuesta del servidor de mapas, se puede contrarrestar aumentando las especificaciones del servidor de mapas, o añadiendo más servidores, ésto es posible dada la escalabilidad del sistema. Para disminuir el tiempo de respuesta del servidor se utilizará un sistema de caché en el servidor.

La decisión implica tener dos capas claramente diferenciadas:

- **Servidor**: En los sistemas SIG, el servidor sería un servidor de mapas. Que recibirá las peticiones de la aplicación cliente y las resuelve en forma de información espacial. Dentro de la capa de servidor se utilizará un sistema de caché, que almacenará la información más consultada, para disminuir el tiempo de respuesta.
- **Cliente**: El cliente será una aplicación web implementada con HTML, CSS y javascript. Los usuarios del SMC se conectarán a la aplicación mediante un navegador web.

8.2 Tecnologías SIG

Existen en el mercado diferentes tipos de tecnologías que se pueden utilizar para implementar un SIG, tanto software como hardware. En este apartado se analizan las tecnologías disponibles y se seleccionarán las más adecuadas para el sistema a desarrollar.

Como en la mayoría de sistemas de información del mundo TIC, en los SIG existen dos grupos de software en referencia a su comercialización, OpenSource y software propietario. Las aplicaciones OpenSource, son aplicaciones de código abierto mantenidas y desarrolladas por una comunidad o grupo de desarrolladores. No requieren ningún tipo de licencia de uso, ni tampoco requieren la comprar de software o de algún tipo hardware específico. Un ejemplo de este tipo de software puede ser MapServer o GeoServer. Los software propietarios están desarrollados y mantenidos por una compañía. Éste tipo de software requiere la compra del producto o de licencias de uso. El software propietario más utilizado en la actualidad son las soluciones que ofrece ArcGIS, de la compañía ESRI.

Para la implementación del presente proyecto se opta por el uso de herramientas OpenSource. Los motivos se describen a continuación:

- El uso de software OpenSource supone un gran ahorro a nivel económico en el proyecto. Dado que la herramienta será utilizada por la gran mayoría de técnicos del SMC, usar herramientas OpenSource evita el uso de compras de licencias. Ésto también es extensible a la adquisición de nuevas versiones.
- Utilizando aplicaciones OpenSource, el sistema se beneficia de la comunidad activa que existe detrás de cada software. Ésto resulta una ventaja frente a las posibles actualizaciones que se realizan en las aplicaciones propietarias. Que suelen tardar más tiempo en lanzar actualizaciones, ya que normalmente suelen esperar a tener un gran pack de mejoras o nuevas funcionalidades antes de lanzar una nueva versión al mercado.
- Posibilidad de participar en los proyectos OpenSource.

8.3 Servicio de comunicación SIG

A continuación se describirán los servicios que se utilizarán en el sistema. Con la intención de que el sistema, y en especial la capa servidor, pueda ser consultado desde cualquier tipo de cliente SIG, se utilizarán protocolos y servicios estándares del OGC. Todos los servicios ofrecerán la posibilidad de realizar consultas mediante HTPP(ésto incluye aplicaciones web o clientes de escritorio capaces de realizar peticiones vía HTTP). En estas URLs, se especificarán ciertos parámetros para que el servidor pueda resolver la petición. Los parámetros indicarán conceptos como, la capa a consultar, extensión de terreno, nivel de zoom, estilo a aplicar a la información, y los parámetros de consulta de la propia información a consultar como puede ser la fecha.

WMS

Son las siglas de *Web Map Service*. Servicio que genera mapas de datos espaciales dinámicos en formato de imagen. A partir de una serie de parámetros en la petición URL, el servicio devuelve una representación en forma de imagen. El servicio permite una serie de acciones que se utilizan para recuperar diferentes tipos de información. Una de estas acciones es *GetCapabilities*, que muestra los metadatos del servicio.

WMTS

Servicio estándar que sirve teselas(o tiles[11])de imágenes pregeneradas(normalmente en un sistema de caché), de una imagen más grande. Las consultas se realizan mediante HTTP. De la misma forma que WMS, el servicio WMTS utiliza una serie de parámetros para determinar la información espacial a consultar. El cliente realiza una serie de peticiones para una imagen. El servidor devolverá las teselas al cliente y éste compondrá la imagen completa. El servicio WMTS también, permite una serie de operaciones que recuperan diferentes tipos de información.

8.4 Capa Servidor

La capa de servidor como se ha comentado en anteriores apartados, será la encargada de recibir las peticiones de los clientes y resolver dichas peticiones. Para resolver estas peticiones el servidor de mapas accederá a los diferentes repositorios, dependiendo del caso, y procesará la información, y devolverá la información en forma de imágenes o en formato de texto. El servidor estará compuesto por dos componentes: MapServer como servidor de mapas y MapCache como sistema de caché. Éstos dos componentes se comunicarán y resolverán las peticiones realizadas por los clientes.

8.4.1 MapCache

MapCache nació en 2010 como proyecto independiente, y que se incluyó dentro del proyecto de MapServer en su versión 6.2 *released* en el año 2012. Al igual que MapServer esta implementado en C/C++. Soporta diferentes tipos de protocolos de tiles, entre ellos Google XYZ, WMTS, TMS, etc. Dispone de un fichero de configuración, *mapcache.xml,* donde se configuran las capas que se han de cachear, y se especifican que parámetros necesitan, como por ejemplo la estructura de directorios, etc. Se puede entender MapCache como un proxy, que intercepta las peticiones que se realizan al servidor de mapas. MapCache intercepta estas peticiones, y en el caso de disponer de las tiles demandadas, las recupera de su sistema de caché y las devuelve. El resto de peticiones, ya sean peticiones de tiles que no tiene almacenadas o peticiones que no es capaz de resolver, las redirecciona al servidor de mapas(hacia MapServer).

11 La traducción al castellano seria *teselas*

MapCache es un servidor que implementa tile caché. Ésta tecnología consiste en dividir una imagen en varias imágenes más pequeñas. Para realizar la división, se define una malla regular (en forma de cuadricula) que divide una imagen en varias imágenes más pequeña, las tiles. Mediante un algoritmo se determinan los nombre de las imágenes según el cuadrante al que pertenezcan, de esta forma se puede saber a que parte de la imagen pertenece una tile en concreto. En la ilustración 21 se puede ver como se crean las tiles. Los *levels*, hacen referencia al nivel de zoom.

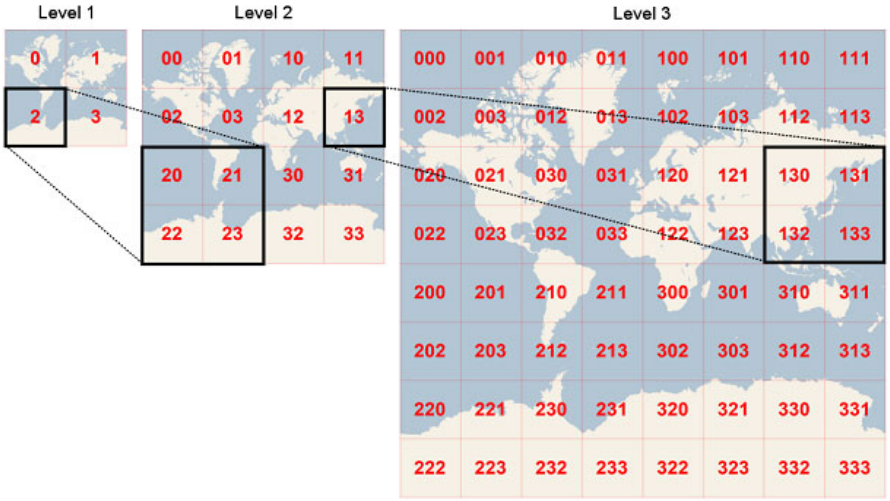


Ilustración 21: Concepto de tiles

La primera vez que se realiza una petición de una tile, para un zoom concreto, el servidor comprueba si ya dispone de la tile para ese zoom en su sistema de caché. Si el servidor dispone de esta tile, la recupera y la devuelve al cliente. En caso de que no disponga de la tile, realiza una petición al servidor de mapas para la extensión y el zoom de la tile que el cliente ha solicitado(petición mediante WMS). Una vez recuperada la imagen del servidor de mapas, MapCache guarda la tile en su sistema de caché. De este modo si se vuelve a pedir la misma tile para el mismo zoom, la tile ya estará generada. MapCache permite configurar la estructura de directorios donde se almacenan las tiles. La configuración de MapCache se explica en el apartado MapCache. Aclarar que para una imagen, el cliente realiza el cálculo necesario para saber que tiles ha de pedir al servidor para construir la imagen. En la ilustración 22 se muestra el diagrama de flujo del funcionamiento de MapCache.

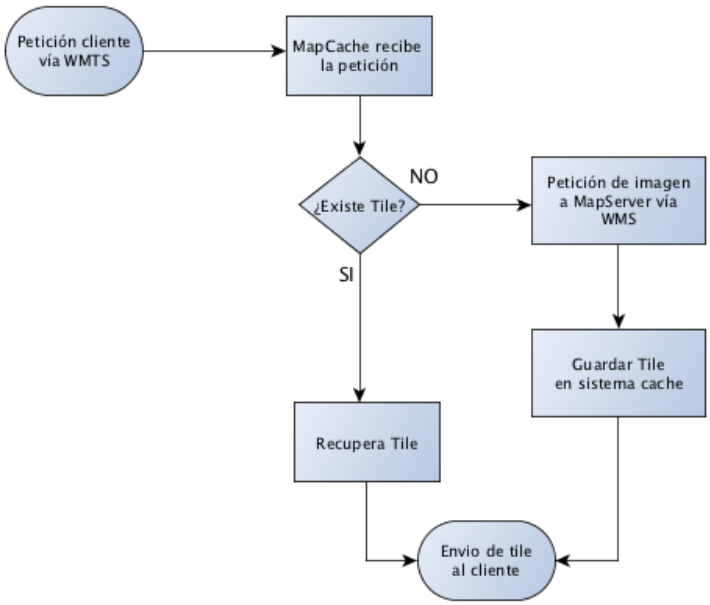


Ilustración 22: Diagrama de flujo MapCache

Otra característica importante de MapCache, es la posibilidad de pregenerar las tiles independientemente de las peticiones de usuario. Ésto se denomina seeds(semillas en inglés). La herramienta *seeds*, lo que permiten es hacer peticiones por linea de comando que pregeneran tiles para capas determinadas, que se almacenan en el sistema de caché. Ésta funcionalidad se utilizará para pregenerar tiles. En el apartado MapCache se describe la herramienta con más detalle.

La comunicación entre la aplicación cliente y MapCache será a través de la interfaz estándar WMTS. Los clientes que soportan éste servicio son capaces de determinar qué tiles se han de solicitar para una imagen en concreto, y posteriormente será capaz de unificar las tiles para mostrar una única imagen.

Con todo ésto el sistema agiliza las respuestas de la capa de servidor. Básicamente por dos motivos:

- En lugar de realizar una sola petición para una imagen de gran tamaño, el cliente realiza múltiples peticiones de imágenes más pequeñas y de menor tamaño. Ésto optimiza la comunicación entre el cliente y el servidor.
- Al tener las imágenes pregeneradas, no es necesario procesar todas las peticiones de los clientes.
- Aumenta la sensación de velocidad del cliente, ya que permite a la aplicación cliente ir mostrando las imágenes en pantalla según le van llegando del servidor.

La implementación de MapCache se describe en el apartado MapCache

8.4.2 MapServer

MapServer es un servidor de datos espaciales para aplicaciones Intranet/Intraweb. Se desarrolló en 1990 en la universidad de Minesota, implementado en C/C++. Actualmente es un proyecto de código abierto de OSGeo[12].

12 OSGeo es una fundación que promueve y da soporte al desarrollo de herramientas espaciales colaborativas. http://www.osgeo.org/

MapServer se ejecuta mediante tecnología CGI, y corre sobre un servidor web Apache. MapServer utiliza una serie de ficheros llamados MapFile, donde se especifica la configuración de las capas, proyecciones, extensión, origen de datos, simbología, etc. Las peticiones que recibe el servidor hacen referencia a un MapFile concreto, dentro de cada MapFile puede haber configurada una o varias capas, y cada capa a su vez puede ser visualizada mediante diferentes leyendas. A partir de los parámetros especificados en la petición y la configuración especificada en el MapFile, MapServer construye la información espacial requerida. Ésta información puede ser en forma de imagen o en formato texto(puede ser tanto en texto plano, HTML, XML, GML, etc.). Los MapFile se utilizan normalmente para configurar capas estáticas. Pero MapServer también nos permite acceder a capas o crearlas dinámicamente. Pera ello utiliza lo que se denomina MapScript. Éstos script se pueden implementar en diferentes lenguajes, como Python, PHP, Perl, Ruby, etc. Los MapScripts permiten la creación en tiempo de ejecución de objetos en memoria que representan a los ficheros MapFiles. Todo esto hace que MapServer sea más dinámico a la hora de poder consultar y generar información.

A modo de resumen estas son las principales características de MapServer:

- Posibilidad de utilizar scripting en diferentes lenguajes de programación, como Python, Perl, PHP, Ruby. Java, etc.
- Soporta consultas de grandes volúmenes de información, ya sea en formato raster, como vectorial.
- Se puede ejecutar en diferentes sistemas operativos, entre ellos linux, Mac OS X o Windows.
- Ofrece renderización de alta calidad.
- Soporta diferentes tipos de formato de salida, como PNG, JPG, GeoJSON, GML, ...
- Soporta prácticamente la totalidad de protocolos estándar definidos por OGC.
- Posibilidad de acceso a la mayoría de bases de datos espaciales, como Oracle Spatial o PostGIS.

En la ilustración 23 podemos ver un esquema de la arquitectura de MapServer.

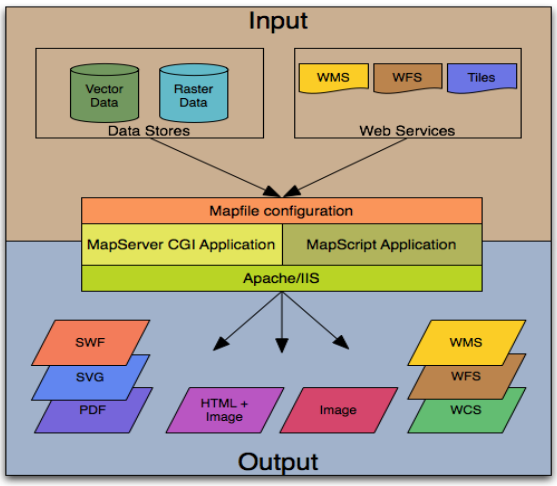


Ilustración 23: Esquema arquitectura MapServer.
http://mapserver.org/introduction.html

http://www.osgeo.org/home

8.5 Capa cliente

La capa cliente estará compuesta por una aplicación web, a modo de visor SIG, que permitirá al usuario visualizar información espacial. Será una aplicación de las denominadas RIA[13] y será una aplicación compuesta por una sola página web (SPA[14]). Las aplicaciones RIA son aplicaciones que ejecutan gran parte de lógica del lado cliente, y únicamente piden información al servidor cuando lo necesitan. La aplicación se encargará de visualizar y controlar la interacción del usuario con los datos visualizados. Se comunicará vía HTTP con los diferentes servicios web. Como toda aplicación web, será necesario el uso de un navegador web para su ejecución.

La aplicación estará instalada en un servidor web Apache, en una máquina virtual del entorno virtualizado del SMC. Los usuarios desde sus máquinas accederán a este servidor mediante una navegador web.

8.5.1 Comunicación de la capa cliente

El cliente se comunicará con la capa de servidor mediante HTTP. En ésta primera versión del sistema, la aplicación cliente se comunicará con tres servidores:

- Servidores de mapas que se implementarán y configurarán en este proyecto, MapServer y MapCache., ubicados en el SMC La comunicación con MapCache ser realizará mediante WMTS, y la comunicación con MapServer mediante WMS. La totalidad de las peticiones referentes a datos se realizarán mediante MapCache. La comunicación con MapServer se utilizará para realizar peticiones a las acciones de *GetLegendGraphic* y *GetCapabilities* del servicio WMS, que se usan para obtener la leyenda o paleta en forma de imagen, y para obtener las capas disponibles en el servidor, respectivamente.
- La API REST del SMC. Se utilizará para la consulta de los datos de la XEMA. La API dispone de un recurso que devuelve la información de las lecturas de las estaciones en formato JSON, donde se encuentra la información sobre las lecturas y los metadatos de las estaciones. Los metadatos poseen las coordenadas latitud/longitud de cada estación. Mediante estas coordenadas el visor pintará de forma vectorial las lecturas por pantalla. Esta petición se realizará mediante AJAX.
- Servicio WMS del ICGC[15], para la consulta de mapas y cartografía del territorio catalán.

En la ilustración 24 se muestra un esquema de los servicios que utiliza la capa cliente.

13 Rich Internet Applications. http://es.wikipedia.org/wiki/Rich_Internet_Applications

14 Single Page application, aplicaciones compuestas por una única página web. La aplicación recarga la información en una sola página. http://en.wikipedia.org/wiki/Single-page_application

15 Institut Cartogràfic i Geològic de Catalunya. http://www.icgc.cat/

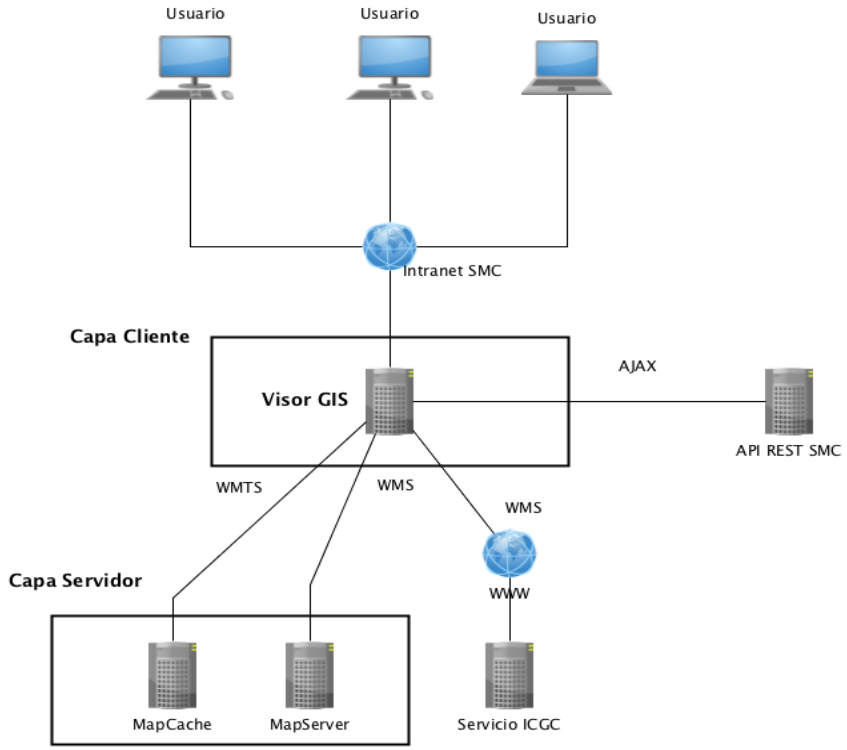


Ilustración 24: Esquema comunicación aplicación cliente

8.5.2 Librerías y frameworks de desarrollo

El visor se desarrollará utilizando los siguientes lenguajes:

- Javascript, permitirá que la aplicación sea dinámica. Controlará la interacción del usuario con la aplicación y gestionará la visualización de los datos.
- HTML, lenguaje que utiliza el navegador web para interpretar i visualizar la información.
- CSS, se utilizará para aplicar estilos a la aplicación.

Existen una serie de frameworks de desarrollo y librerías, que facilitan el desarrollo de aplicaciones web. Por norma general estas herramientas encapsulan y abstraen parte de las complejidades de la programación, haciendo más fácil la implementación. Las tecnologías a modo de frameworks y librerías necesarias para la implementación del presente proyecto, serán librerías javascript. El visor tiene como complejidades inherentes los siguientes elementos:

- Visualización de mapas y datos georeferenciados.
- Funcionalidades SIG básicas, que serian las típicas de cualquier cliente SIG de escritorio, como pueden ser realizar zooms, navegación, obtener información de un punto, etc.
- Interfaz de usuario apta para el uso de mapas.
- Control de las interacciones del usuario con la interfaz.

Las tecnologías que serán necesarias para la implementación de la capa cliente se pueden dividir básicamente en dos grupos:

- Librerías de visualización de datos georeferenciados.

- Frameworks de desarrollo.

A continuación se describen estos dos grupos y se enumeran las posibilidades existentes.

8.5.2.1 Librerías de visualización de datos georeferenciados

Librerías que abstraen de la complejidad de visualizar y trabajar con datos georeferenciados. En los últimos años debido al auge del uso de visualización de mapas en aplicaciones y páginas web, se han desarrollado varias librerías para éste tipo de uso.

En el desarrollo del visor, la elección de la librería de visualización de mapas es la parte más importante. La librería deberá ser capaz de proporcionar todas las funcionalidades que requiere la aplicación, incluidos los servicios de comunicación WMS, WMTS. Puesto que los datos georeferenciados con los que trabajará el sistema pueden ser de un gran volumen, la librería de visualización ha de ser capaz de trabajar con volúmenes de datos importantes. Los requisitos de la capa cliente se especificaron en el apartado Cliente.

Las características que ha de cumplir la librería de visualización son las siguientes:

- Posibilidad de realizar peticiones vía WMTS y WMS. Pensando en futuras ampliaciones de la aplicación de visualización, la librería deberá también permitir realizar peticiones WFS.
- Renderizar datos raster en forma de tiles.
- Permitir utilizar diferentes proyecciones. En la primera versión solo se implementará la visualización de los datos en una proyección. Pero en futuras versiones se deberán poder utilizar varias proyecciones.
- Permitir pintar y trabajar con datos vectoriales.
- Funcionalidades básicas de un visor SIG, como puede ser realizar zooms, navegar, zooms a regiones determinadas, realizar peticiones de un punto geográfica mediante *GetFeatureInfo, etc.*
- Cambiar la opacidad de una capa.
- Realizar clusters[16] de datos vectoriales. Con el objetivo de agrupar puntos de información que se encuentren relativamente cercanos, para evitar la aglomeración de información.
- Gestionar eventos sobre capas o mapas, como puede ser un click en un punto del mapa.
- Permitir la ocultación de la información sin eliminar la capa.
- Posibilidad de ordenar la información que se esta visualizando en pantalla.

Otros funcionalidades y características a tener en cuenta para la elección de la librería de visualización son:

- Buena gestión de la memoria del navegador. Ésto es muy importante dado el gran volumen de información que suelen tener los datos espaciales y mapas.
- Ha de tener el mayor número de funcionalidades posibles. En ésta primera versión las funcionalidades son las básicas de un SIG, y prácticamente la mayoría de librerías dispondrán de ellas. Pero se ha de pensar en futuras funcionalidades que se pueden querer añadir al visor.
- Buen manejo de datos espaciales, tanto vectorial, como rasters.
- Flexibilidad a la hora de poder extender o añadir nuevas funcionalidades.
- Velocidad de respuesta lo más pequeña posible.
- Poseer una buena documentación, en forma de API, ejemplos, implementación interna, etc.
- Madurez y robustez.
- Fácil seguimiento de su implementación para poder entender su funcionamiento.

16 En referencia a la visualización SIG, agrupación de cierta información en único punto de visualización.

Existen muchas librerías de visualización de mapas en el mercado. Sin entrar mucho en detalle, a continuación se enumeran tres de las librerías más extendidas y utilizadas, y que cumplen con las funcionalidades necesarias.

Leaflet

Leaflet es una de las librerías más recientes. Ofrece un look muy atractivo. Su principal virtud es la poca carga que genera en el navegador, lo cual lo hace que sea una muy buena opción a la hora de implementar aplicaciones web para móviles. Su uso es muy intuitivo y fácil. Su poco peso en cuanto a tamaño, se debe a que la librería esta pensada para cubrir funcionalidades básicas. Para paliar las pocas funcionalidades que posee, se han desarrollado una serie de plugins a modo de módulos externos que amplia las funcionalidades de la librería.

Google Maps API

Ésta librería desarrollada por Google, es una de las primeras que salió al mercado. De las tres librerías que se exponen en este proyecto, posiblemente sea la más fácil de utilizar. Se necesita una API key[17] para su uso. No es una librería de código abierto, puesto que no se puede modificar su código, aunque si que es gratuita. No aporta mucha flexibilidad en este aspecto. Poseen un número aceptable de funcionalidades, aunque al no ser de código abierto, las futuras actualizaciones de esta librería están bajo control de Google.

OpenLayers v2

Esta es la librería más madura y más utilizada de las tres expuestas. Posee multitud de funcionalidades, lo cual la hace algo pesada. Ésto hace que su uso para dispositivos móviles no sea muy aconsejable. Posee la mayoría de protocolos de comunicación desarrollados por el OGC. De las tres librerías, es la más flexible y más versátil. Gestiona muchos más eventos de usuario sobre el mapa que el resto de librerías. Otro aspecto importante es que es un proyecto de OSGeo[18]. Actualmente se está desarrollando la versión 3 de OpenLayers. Que mejora y optimiza la librería.

Criterio de puntuación - Muy Alta: 4, Alta: 3, Normal: 2, Baja: 1, No disponible : 0

	OpenLayers	Leaflet	Google API
Documentación	**Muy alta** • Muy Extensa • Muchos ejemplos de todas las funcionalidades. • API extensa y bien documentada.	Alta • API documentada. • Los plugins no están muy documentados. • Pocos ejemplos	Alta • API bien documentada • Gran cantidad de ejemplos
Protocolos	**Muy alta** WMS, WMTS, WFS,WFS-T, WCS, TMS, WPS, SOS, CSW	Normal WMS, WFS, WFS-T, WMTS, TMS	
Rendimiento	Normal	**Muy alta**	Alta
Interacción eventos usuario	**Muy alta**	Alta	Alta
Funcionalidades	**Muy Alta**	Normal	Alta

17 Clave a modo de registro o autorización necesario para poder utilizar la API de Google

18 Open Source Geospatial Foundation, organización que da soporte al desarrollo de aplicaciones Open Source en el campo geospacial. http://www.osgeo.org/

	OpenLayers	Leaflet	Google API
	Muchas y muy variadas	Necesita plugins para funcionalidades concretas	
Flexibilidad	**Muy alta**	Alta	Baja
Posibilidad de ampliación	**Muy alta**	**Muy alta**	No disponible No se puede
Tratamiento Raster	**Muy alta**	Alta	Alta
Tratamiento Vectorial	**Muy alta**	Alta	Alta
Soporte Tiles	Si	Si	
Proyecciones	**Muy Alta** • Utiliza de forma nativa proj4js. • Muy extensa	Alta Existen plugins para añadir proyecciones con proj4js	No disponible Solo permite trabajar con latitud / longitud

Tabla 15: Tabla comparativa de librerías JavaScript de visualización de mapas

En la tabla 15 se puede ver que OpenLayers sería la opción que más se acerca a las necesidades del proyecto. La mayor pega de esta librería es su tamaño. Los motivos a modo de resumen son los siguiente:

- Contiene multitud de funcionalidades SIG, tanto para datos vectoriales, como raster.
- Es maduro y robusto.
- Permite el uso de la mayoría de servicio y protocolos estándar actuales.
- Posee una gran cantidad de ejemplos en su página web. Que servirán de guía para la implementación.
- Posee una documentación extensa, completa y comprensible de su librería.
- Al ser una de las librerías más utilizadas, existen en la red multitud de tutoriales y explicaciones respecto al uso de la librería.

Por contra también existen inconvenientes:

- OpenLayers carga toda la librería, aunque existan clases o módulos que no se utilicen.
- No tienen compatibilidad con HTML5. Ésto se ha implementado en la versión 3 de OpenLayers que en la actualidad esta en fase de desarrollo.

8.5.2.2 Frameworks de desarrollo.

Implementados en javascript. Ofrecen una marco de trabajo unificado, que hace que el código de la aplicación tenga coherencia y esté bien estructurado. Facilita la creación de ciertos elementos HTML, y abstrae la gestión de estos. También suelen facilitar y gestionar la comunicación con servidores mediante AJAX. Por estos motivos será útil el uso de alguno de estos frameworks de desarrollo para javascript en el desarrollo del visor.

Los requisitos fundamentales para la selección de una framework de desarrollo son los siguientes:

- Permitir comunicación vía AJAX con una API REST.
- Ofrecer una buena organización de código.
- Fácil de ampliar o extender.
- Buena documentación.

- Robusto y fiable

Actualmente existen multitud de frameworks javascript que cumplen con los requisitos expuestos. Un ejemplo de los más usados y contrastados son los siguientes:

- Dojo toolkits
- Exts JS
- Angular
- Jquery
- Backbones

Algunos como *Dojo* y *Ext Js* son fáciles de usar y están bien documentados. Otros como *Angular* y *Backbones*, son algo más potentes, pero su curva de aprendizaje es algo mayor. Para el desarrollo del proyecto cualquiera de los enumerados cumple con los requisitos. Se escoge **Dojo toolkits** como framework para el desarrollo del visor. Los motivos principales de esta elección son los siguientes:

- Buena documentación. Tanto de la API, como documentación acerca de su uso, con multitud de ejemplos.
- Permite la creación de módulos o widgets, que permiten encapsular a modo de objeto un componente con funcionalidades concretas.
- Permite la separación entre la lógica de aplicación y la presentación a modo de templates.
- Permite realizar herencias entre los módulos.
- Permite extender de forma sencilla los componentes del framework.
- Permite una buena organización de la información.
- Lleva una serie de componentes, como pueden ser calendarios, tablas, campos de selección, etc.

Resumen

Se decide utilizar las siguiente herramientas para el desarrollo de la aplicación:

- **OpenLayers** como librería de visualización
- **Dojo Toolkits**, como framework de trabajo javascript.

9 Capa servidor

A continuación se describe la implementación y configuración del componente de servidor.

9.1 Hardware

MapServer y MapCache se instalaran y ejecutarán en una misma máquina. Ésta máquina estará montada en el entorno virtual del SMC. El hecho de que se utilice un entorno virtual, hace que la configuración de las especificaciones de la máquina puedan aumentarse en el caso que el rendimiento no sea el deseado. Por lo tanto se podrá aumentar recursos de la máquina tales como unidades de CPU, memoria RAM o espacio en disco siempre que se necesite. Otro motivo importante por el que se ha decidido utilizar un servidor virtualizado, es la posibilidad de crear

snapshots[19] de un momento concreto de la máquina y también facilita la creación de una copia de seguridad de la máquina cada cierto tiempo.

Como detalle a tener en cuenta, debido a la gran cantidad de ficheros que puede llegar a generar el sistema de caché de MapCache, se utilizará un segundo disco con el único objetivo de almacenar las tiles que vaya creando MapCache. Para evitar que el disco se llene, se ejecutará un proceso en el crontab de la máquina que eliminará todas las tiles de las capas de los datos meteorológicos que tengan una antigüedad mayor de dos días.

Especificaciones de la máquina virtual que alberga el servidor:

- Sistema operativo Centos de 64 bits
- 4 procesadores de CPU
- 2GB de RAM
- Un primer disco de 25GB de espacio de almacenamiento
- Un segundo disco de 5GB de espacio de almacenamiento únicamente para el sistema de caché de MapCache.
- Sistema de ficheros XFS, debido a la gran cantidad de ficheros que el sistema de caché creará.

Software necesario en la máquina servidor:

- Servidor web Apache. Configurado para poder trabajar con FastCGI, que permite procesar peticiones a mayor velocidad.
- MapServer v6.4.1, y todas sus dependencias.
- MapCache v6.4.1, y todas sus dependencias.

Tanto MapServer como MapCache correrán a modo de CGI sobre el servidor web Apache.

9.2 MapCache

Antes de entrar en conceptos más concretos de la configuración de MapCache, hay tener en cuenta una serie de conceptos.

Parámetro tiempo

Esta variable es una de las más importantes del sistema SIG, y está presente en todos los datos meteorológicos. Mediante la fecha se indica la temporalidad del dato a consultar. Esta fecha está compuesta de varios parámetros. MapServer necesita los parámetros fecha para determinar que fichero consultar. por tanto MapCache también los tendrá configurados para poder realizar la petición a MapServer. Éstos parámetros específicos de cada capa en MapCache se configuran mediante el elemento <*dimension*>.

9.2.1 Configuración capas MapCache

Un aspecto muy importarte a la hora configurar las capas en MapCache es el paso del parámetro *style*(leyenda) a la hora de realizar las peticiones a MapServer. En principio para cada petición de tile que recibe MapCache, si no la tiene en caché, hace una petición al servicio WMS de MapServer pasando todos los parámetros configurados en el elemento <source> de mapcache.xml, junto con algunos elementos de la misma petición. Debido a un error en MapCache(detectado, pero no corregido en la versión 6.4.1), al hacer la petición a MapServer,

19 Copia de seguridad de la totalidad de una máquina virtual en un momento concreto

MapCache no pasa el parámetro *style* especificado en la petición WMTS. Ésto supone un problema, y requiere organizar las capas de MapCache de forma distinta a lo habitual. Para solventar el problema, se ha configurado en MapCache tantas tiles de una misma capa como estilos tiene una capa en MapServer. Es decir, si en MapServer tenemos configurada una capa de modelos con el nombre *WRF27_HGT_ISOLINEA_500-ISBL* y ésta se puede consultar con los estilos *negre*(que muestra las isolíneas de color negro) y *blanc*(que muestra las isolíneas de color blanco), en MapCache se configurarán dos capas distintas. Una con el nombre WRF27_HGT_ISOLINEA_500-ISBL_*negre* y otra con el nombre WRF27_HGT_ISOLINEA_500-ISBL_*blanc*. Y para cada una de estas capas el estilo se especifica en el <source>. En la ilustración 25 podemos ver una esquema del ejemplo descrito anteriormente.

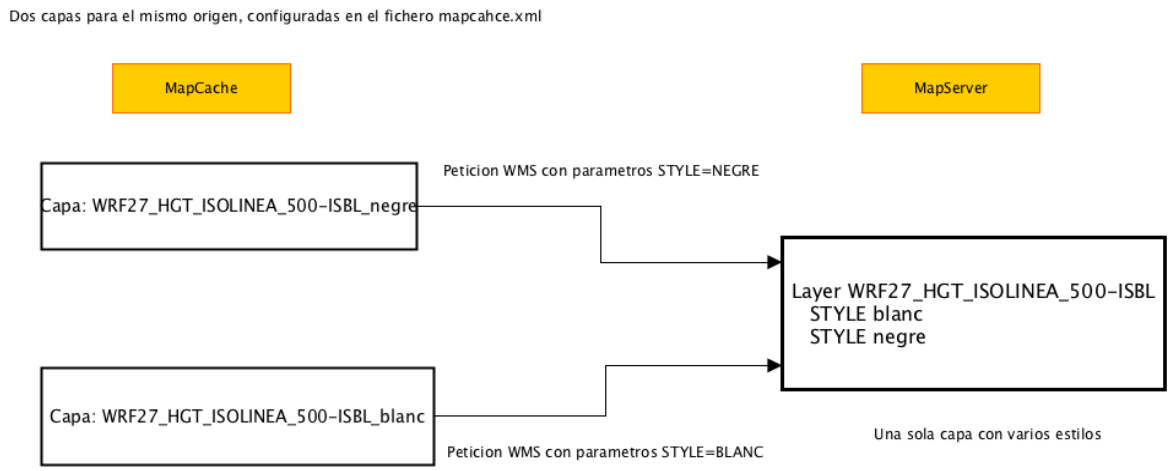


Ilustración 25: Paso del parámetro STYLE entre MapCache y MapServer

9.2.2 Peticiones WMTS

En apartados anteriores se ha indicado que las peticiones que realiza la aplicación cliente a MapCache, se hace a partir del servicio WMTS. En esta petición se especifican una serie de parámetros para que la petición pueda ser resultar por MapCache.

El patrón de una petición HTTP al servicio WMTS seria el siguiente:

```
http://10.116.16.133/mapcache/wmts/
?service=wmts&request=getTile&version=<wmts-
version>&layer=<layer>&style=<style>&tilematrixset=<tilematrixset>&tilematrix=<tilematrix>&tilerow=<tilerow>&tilecol=
<tilecol>&format=<format>
```

Y un ejemplo de petición para la capa de Meteosat para la fecha 16/04/2014 a las 12:30, sería el siguiente:

```
http://10.116.16.135/mapcache/wmts/?
ANY=2014&MES=04&DIA=16&HORA=12&MINUT=30&SERVICE=WMTS&REQUEST=GetTile&VERSION=1.0.0&LAY
ER=MSG3_IR_108_IRcolor&STYLE=default&TILEMATRIXSET=WGS84&TILEMATRIX=6&TILEROW=19&TILECOL=6
0&FORMAT=image%2Fpng
```

En la petición de ejemplo se pueden observar diferentes parámetros, unos son obligatorios en peticiones al servicio WMTS, y otros son parámetros específicos para cada capa. Respecto a los parámetros propios de la capa, cabe mencionar que cada capa puede tener configurados diferentes parámetros o ninguno, por ejemplo en los rayos se tiene que especificar la acumulación a consultar en un parámetro, y las capas estáticas no hay que especificar ningún parámetro propio de la capa. En las capas meteorológicas, los parámetros referentes al tiempo siempre están presentes. Dependiendo de si la capa proporciona datos diarios, horarios o minutales, se especifican más precisión en la fecha o menos. De forma que por ejemplo para un dato diario, solo habría que especificar año-mes-dia, y para un dato minutal habría que especificar años-mes-dia-hora-minuto.

A continuación se hace una descripción de los parámetros presentes en consultas WMTS:

Nombre parámetros	Descripción
Parámetros obligatorios en peticiones al servicio WMTS	
SERVICE	Especifica el tipo de servicio que se requiere
REQUEST	Indica el tipo de operación. • **GetTile**: devuelve la porción de una imagen en forma de tile. • **GetCapabilities**: la petición se realiza para obtener los metadatos del servicio, es decir las capas y los parámetros necesarios para las peticiones • **GetFeatureInfo**: acción que devuelve la información de una geometría(cualquier tipo de geometría) localizada en un pixel en concreto de una capa.
VERSION	Especifica la versión del servicio que se demanda. Lo habitual es utilizar 1.0.0
LAYER	Nombre de la capa a consultar. Este nombre ha de estar configurado en el fichero de configuración de MapServer.
STYLE	Indica el estilo con el que se puede consultar una determinada capa. Es obligatorio indicar un nombre, aunque puede tener como valor *default*, que indica que se requiere el estilo por defecto.
TILEMATRIXSET	Es la matriz o cuadricula que se asigna a un área, y consiste en una referencia espacial, extensión geográfica, resolución y tamaño de tile.
TILEMATRIX	Indica el nivel del zoom a consultar.
TILEROW	Indica la posición de la X a recuperar. De la cuadricula o grid que se crea para una imagen, este valor indica el número de fila.
TILECOL	Indica la posición de la Y a recuperar. De la cuadricula o grid que se crea para una imagen, este valor indica el número de columna.
FORMAT	Indica el formato de imagen que se quiere pedir.
Parámetros propios de la capa (estos varían según la capa)	
ANY	Especifica el año que se quiere consultar
MES	Especifica el mes que se quiere consultar
DIA	Especifica el día que se quiere consultar
HORA	Especifica la hora que se quiere consultar
MINUT	Especifica el minuto que se quiere consultar

Tabla 16: Parámetros petición WMTS

La construcción de la petición, la realiza y construye directamente el visor, mediante la librería OpenLayers, ya que ésta soporta peticiones vía WMTS y es capaz de construir las peticiones necesarias para una imagen.

9.2.3 Configuración

MapCache utiliza un fichero de configuración donde se especifican las capas para las cuales puede recibir peticiones, las proyecciones o grids que soporta, la ubicación, configuración del sistema almacenamiento de la caché, parámetros para las capas, origen de generación para las tiles (en el caso de este proyecto peticiones a MapServer vía WMS) servicios que soporta, etc. Éste fichero es *mapcache.xml*. En este fichero se especifica todo el comportamiento de MapCache. Una de las partes más importes de la configuración de este fichero, es la configuración de las capas.

Un aspecto muy importante en la configuración del fichero *mapcache.xml*, es la posibilidad de utilizar variables, dentro del fichero (también es posible en MapServer). De modo que MapCache será capaz de substituir estas variables por su valor. Las variables a substituir se indican mediante corchetes. Por ejemplo podemos utilizar la variables *{dia},* que se especifica como parámetro en una de las peticiones WMTS y utilizarla por ejemplo para construir la ubicación de las tiles cacheadas.

Dado que el fichero de configuración de MapCache para este proyecto es muy extenso, a continuación se explicarán los parámetros más relevantes de este fichero junto con algunas capas a modo de ejemplo. A continuación se muestra un ejemplo (no es el fichero *mapcache.xml* del sistema) de un fichero *mapacache.xml.*

```
<?xml version="1.0" encoding="UTF-8"?>
<mapcache>
   <cache name="disk" type="disk">
      <base>/var/www/html/mapcache/cache/</base>
      <symlink_blank>/var/www/html/mapcache/cache/</symlink_blank>
      <template>/var/www/html/mapcache/cache/{tileset}#{grid}#{any}#{mes}#{dia}#{hora}#{minut}/{z}/{x}/{y}.{ext}</template>
   </cache>
   <!-- CAPES DE FONS -->
   <source name="fonsclar" type="wms">
      <getmap>
         <params>
            <FORMAT>image/png</FORMAT>
            <LAYERS>fonsclar</LAYERS>
            <map>/var/www/html/mapserver/mapfile/fonsclar.map</map>
         </params>
      </getmap>
      <http>
         <url>http://localhost/cgi-bin/mapserv.fcgi</url>
      </http>
   </source>
   <tileset name="fonsclar">
      <source>fonsclar</source>
      <cache>disk</cache>
      <grid>WGS84</grid>
      <grid>UTM31N</grid>
     <grid>GoogleMapsCompatible</grid>
      <format>PNG</format>
      <metatile>2 2</metatile>
      <metabuffer>4</metabuffer>
   </tileset>
   <source name="MSG3_WV_062_paleta1" type="wms">
      <getmap>
         <params>
            <FORMAT>image/png</FORMAT>
            <LAYERS>MSG3_WV_062_paleta1</LAYERS>
            <MAP>/var/www/html/mapserver/mapfile/service_meteosat_includes.map</MAP>
             <STYLES>paleta1</STYLES>
         </params>
      </getmap>
      <getfeatureinfo>
         <info_formats>text/html,text/plain,application/vnd.ogc.gml</info_formats>
         <params>
            <QUERY_LAYERS>MSG3_WV_062</QUERY_LAYERS>
```

```
        </params>
      </getfeatureinfo>
      <http>
        <url>http://localhost/cgi-bin/mapserv.fcgi</url>
      </http>
    </source>
    <tileset name="MSG3_WV_062_paleta1">
      <source>MSG3_WV_062_paleta1</source>
      <cache>disk</cache>
      <grid>WGS84</grid>
      <grid>UTM31N</grid>
      <format>PNG</format>
      <metatile>2 2</metatile>
      <metabuffer>4</metabuffer>
      <dimensions>
        <dimension type="values" name="minut" default="00">00,15,30,45</dimension>
        <dimension type="values" name="hora" default="00">
            00,01,02,03,04,05,06,07,08,09,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24</dimension>
        <dimension type="values" name="dia" default="26">
            01,02,03,04,05,06,07,08,09,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31</dimension>
        <dimension type="values" name="mes" default="03">01,02,03,04,05,06,07,08,09,10,11,12</dimension>
        <dimension type="values" name="any" default="2013">2012,2013,2014,2015,2016,2017,2018,2019,2020</dimension>
      </dimensions>
    </tileset>
    <default_format>JPEG</default_format>
    <service type="wms" enabled="true">
      <full_wms>assemble</full_wms>
      <resample_mode>bilinear</resample_mode>
      <format>JPEG</format>
      <maxsize>4096</maxsize>
    </service>
    <service type="wmts" enabled="true"/>
  <lock_dir>/var/www/html/mapcache/cache/</lock_dir>
</mapcache>
```

Elemento <cache>

Elemento que especifica dónde y cómo se ubicarán las tiles, que cachea MapCache. Es posible configurar diferentes ubicaciones. En el presente proyecto solo se utilizará una ubicación. Se configura un directorio base mediante un path absoluto que apunta a var/www/html/mapcache/cache. A partir de este path base, se utilizan una serie de variables (que son las que se encuentran entre corchetes), que mapcache substituirá por su valor en cada petición. Éste elemento tendrá el nombre “disk”, que será referenciado por el elemento <tileset>

- **<cache nombre=”disk”>**: ubicación base donde se almacenarán las tiles. El sistema tendrá como ubicación base para el sistema de caché: */var/www/html/mapcache/cache/.*
- **<symlink_blank>**: especifica la ubicación de la tile por defecto que se utilizará cuando se haga referencia a una tile de una imagen que no contenga información. Es decir, cuando se construye una tile que no tenga información, no se crea una tile en blanco, sino un link a una imagen que no contiene información. Ésto optimiza la creación de imágenes sin información.
- **<template>**: indica la ubicación y la estructura que tendrá el sistema de caché. Primero se ha especificado un path absoluto base */var/www/html/mapcache/cache/*. A partir de este path, las tiles se irán creando en diferentes directorios a partir de una serie de variables. De esta forma MapCache sabrá donde ha de ir a buscar la tile demandadas y también lo utilizará para saber cómo almacenar las tiles que se han de cachear. Este path variable es : *{tileset}/{grid}/{any}/{mes}/{dia}/{hora}/{minut}/{z}/{x}/{y}.{ext}*. Donde:
  - tileset: es el nombre de la capa
  - grid: es la proyección
  - any – mes – dia – hora – minut: son los parámetros de fecha que se especifican en la petición. En el caso que la petición no contenga alguno de los parámetros, MapCache no lo tendrá en cuenta.

- Z: nivel de zoom de la petición
- X: columna de la cuadricula demandada.
- Y: fila de la cuadricula demandada.
- Ext: extensión de la tile.

Elemento <Source>

Especifica el servicio donde se consultarán las tiles. Aquí se indica que el servicio que se tiene que consultar es el WMS de MapServer (ubicado en la misma máquina por este motivo se pone localhost). Se crean tantos elementos source como capas se tengan configuradas en MapServer, teniendo en cuenta lo comentado en el apartado Configuración capas MapCache. Este elemento tendrá un nombre que servirá para referenciarlo desde un elemento <tileset>. Los elementos más relevantes son:

- **<getMap>**
  - **<params>**: parámetros que se adjuntarán en la petición al servicio WMS.
    - **<Layers>:** nombre de la capa configurada en MapServer, sobre la que se quiere realizar la consulta.
    - **<map>:** nombre del fichero MapFile dentro de MapServer, que tiene configurado la capa.
    - **<styles>:** estilo(paleta o leyenda) con el que se quiere recuperar la información.
- **<getfeatureinfo>:** este elemento especifica el formato que se devolverá cuando se realice una petición a la acción *getFeatureInfo* El formato que se utilizará es un texto plano.
- **<http>:** se indica la url del servicio WMS que se ha de consultar. Esta URL puede hacer referencia directamente a MapServer (http://localhost/cgi-bin/mapserv.fcgi), o bien a un MapScript (http://localhost/cgi-bin/[nombreMapScript].py).

Elemento <tileset>

Elemento donde se configuran las tiles que podrán ser consultadas. En presente proyecto se puede entender como la configuración de las capas a visualizar por el visor. Un parte importante de este elemento es la especificación del origen de los datos de las tiles, que en nuestro caso será el servicio WMS de MapServer. El servicio origen se indica mediante el elemento source, donde se indica que capa de MapServer se ha consultar, asi como los parámetros necesarios para la consulta. Se configuran tantos elementos tileset como capas se configuran en el sistema SIG, que serán las consultadas por el visor(de la misma forma que pasa con el elemento source hay que tener en cuenta lo comentado en el apartado Configuración capas MapCache). Los elementos más relevantes de <tileset> son los siguientes:

- **<source>**: nombre del elemento <source> de donde se obtienen las imágenes para el tile. El nombre del elemento source ha de estar presente en el mismo fichero.
- **<cache>:** nombre del elemento caché, donde se almacenarán las tiles generadas. Se utilizará un único elemento caché, que se ha comentado anteriormente.
- **<grid>:** nombre del elemento <grid>, para determinar la proyección. Éste puede estar presente en el mismo fichero o puede estar directamente creado en la configuración que tiene por defecto MapCache. En esta primera versión se utiliza el grid *WGS84*, que esté en la configuración por defecto de MapCache.
- **<dimensions>:** aquí se especifican los parámetros propios y específicos de cada capa y sus posibles valores. Como se ha comentado anteriormente la totalidad de capas

meteorológicas tendrán como mínimo algún parámetro que hará referencia a la fecha, y las capas estáticas o temáticas no tendrán parámetros puesto que no dependen del tiempo.

Elemento<service>

Elemento que configura el tipo de servicios que utilizará MapCache. Las únicas peticiones que MapCache recibirá del visor, serán *WMTS*. Por lo tanto este servicio ha de estar activado. Pero también se han incluido otro tipo de servicios para que puedan ser consultados por otras aplicaciones cliente o directamente desde un navegador vía HTTP.

9.2.4 MapCache Seed

Como se ha comentado en anteriores apartados, cuando MapCache recibe una petición de una tile que no tiene almacenada en su sistema de caché, realiza una petición a MapServer. Para que éste la genere y se la devuelva, una vez recibida la tile MapCache la almacena en su sistema de caché. Por tanto para que se puedan utilizar tiles cacheadas, previamente se ha tenido que realizar una petición de una capa, para una extensión y un zoom en concreto. Para evitarlo, MapCahce ofrece la herramienta *seeder*. Esta herramienta permite pregenerar tiles mediante una orden por linea de comando. Ésto será útil para agilizar el tiempo de respuesta del servidor. De esta forma se generarán las tiles sin necesidad de que se hayan solicitado con anterioridad. Un ejemplo de uso del *seeder* podría ser el siguiente:

```
sudo -u apache env LD_LIBRARY_PATH=/opt/instantclient_11_2/lib:/usr/local/lib /usr/local/bin/mapcache_seed -c
/var/www/html/mapcache/mapcache.xml -o 'now' -g WGS84 -z 7,9 -e '-12617, 4676214, 685627, 5463420' -t fonsclar
```

En el sistema se utilizará la herramienta *seeder* para pregenerar ciertas tiles. Las capas que hacen referencia a datos meteorológicos requieren una serie de parámetros que determinan la fecha para la que se quiere consultar la información. Por lo tanto, en cada ejecución del *seeder*, se tendrá que añadir los parámetros(*dimension* en MapCache) de la fecha a consultar. Para que el sistema siempre tenga en caché los últimos datos, el *seeder* creará las tiles de los últimos datos disponibles de cada capa. Ésto genera una complicación, ya que para cada capa la fecha del último dato es distinta, ni tampoco se puede asegurar la fecha de creación de los ficheros de origen, ya que dependen de varios factores(como volumen de datos, disponibilidad de CPU, etc.). Para solventar ésto, se han creado una serie de scripts en Python, que consultan en el sistema de ficheros cual es el último fichero disponible para cada producto(capa). Cuando el script recupera la última fecha válida para un dato(accediendo al repositorio de datos), ejecuta el *seeder* utilizando como parámetros la fecha del último dato que ha calculado anteriormente. Se crea un script por cada servicio que ofrece MapServer para un conjunto de capas. Éstos scripts se ejecutarán mediante un crontab cada cierto tiempo.

Los scripts que se configuran en el crontab son los siguientes:

```
39 09 * * * python /home/productes/mapcache_seeds/seed_Prescat.py
35 * * * * python /home/productes/mapcache_seeds/seed_XRAD_RAIN1H_COMP_CEXEMA.py
39 * * * * python /home/productes/mapcache_seeds/seed_XRAD_RAIN1H_COMP_CE.py
43 02 * * * python /home/productes/mapcache_seeds/seed_RAIN24H_COMP_CEXEMA.py
00 06 * * * python /home/productes/mapcache_seeds/seed_WRF27-00.py
00 18 * * * python /home/productes/mapcache_seeds/seed_WRF27-12.py
13,28,43,58 * * * * python /home/productes/mapcache_seeds/seed_METEOSAT.py
```

9.3 MapServer

MapServer será quien genere la información espacial. Ésta información por norma general será

servida en forma de imágenes, aunque también servirá información alfanumérica sobre los valores de las capas. Dado que se ha diseñado el sistema mediante una arquitectura cliente-servidor, la capa servidor puede ser consultada por cualquier cliente que sea capaz de realizar peticiones y consumir los servicios de MapServer. Por tanto, las capas y servicios de comunicación que se configurarán en MapServer, podrán ser consultadas por la aplicación de visualización que se desarrolla en el proyecto, y también podrá ser consulta por clientes SIG de escritorio o realizando directamente una petición mediante un navegador web.

MapServer se configura a partir de una serie de ficheros MapFile y MapScript. Mediante MapFiles se configurarán capas estáticas, y se indica su origen de datos, leyendas, y toda la información espacial relacionada con la capa. Los ficheros MapScripts, son scripts implementados en Python, y se utilizan cuando la capa requiere de algún tratamiento especial, como puede ser un suavizado especifico, la generación de un shapefile (como ocurre en la visualización del viento), etc. Estos dos conceptos se explican más adelante(MapScript).

En el sistema se utilizará básicamente el servicio WMS para la comunicación con el servidor. WMS soporta diferentes acciones en las petición. Dependiendo del tipo de acción, se requieren unos parámetros u otros, y dependiendo de la acción se obtienen diferentes tipo de información. En el servidor se configuran las siguientes acciones:

- **GetMap**: acción que devuelve una imagen de un origen de datos(fichero o base de datos). Ésta acción será la que devuelva todas la información en forma de imagen.
- **GetCapabilities**: acción que devuelve en forma de fichero XML los metadatos del servidor de mapas. Es decir, devolverá información de todas las capas que el servidor es capaz de servir, junto con los parámetros que requiere cada capa, las proyecciones de visualización, extensiones, formatos de salida, leyendas, etc. En definitiva la información que el cliente ha de tener para poder consumir los servicios que ofrece el servidor. Las peticiones a esta acción se realizan para cada MapFile individualmente.
- **GetLegendGraphic**: acción que devuelve la leyenda en formato de imagen para la capa requerida. Ésto será útil para mostrar la leyenda de cada capa en el visor. Las ilustraciones 26 y 27 muestran un ejemplo de las leyendas que ofrece MapServer.

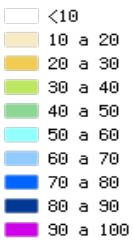


Ilustración 26: Leyenda Prescat

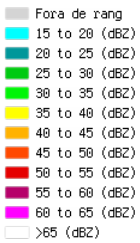


Ilustración 27: Leyenda CAPPI

Tabla 17: Ejemplo de leyenda obtenida mediante GetLegendGraphic

- **GetFeatureInfo**: acción que devuelve la información en valores alfanuméricos de una capa en un punto determinado. La información se puede devolver en diferentes formatos. El presente proyecto utilizará un formato en texto plano. *GetFeatureInfo* se utilizará cuando el cliente requiera el valor de una capa en un punto determinado.

Como información adicional, cabe aclarar que un servidor WMS ha de ofrecer como mínimo las acciones *GetMap* y *GetCapabilities*. Las acciones *GetFeatureInfo* y *GetLegendGraphic*, son opcionales en un servidor WMS.

9.3.1 MapFile

Los ficheros MapFile son ficheros de texto plano (con extensión ".map") dónde se configuran una o varias capas. Son los ficheros de configuración base con los que trabaja MapServer. Cuando un cliente realiza una petición a una capa, en la petición ha de indicar en que MapFile se encuentra la capa requerida. Por lo tanto, el cliente ha de conocer el nombre de estos MapFiles para poder realizar la petición. Para la visualización de imágenes, en el presente proyecto quien conoce el nombre y ubicación de estos MapFile, es MapCache, quien tiene configurado para cada capa, el nombre del MapFile en el que se encuentra (siempre que la capa no requiera la ejecución de un MapScript).

Un fichero MapFile puede tener configuradas varias capas y cada capa tendrá su origen de datos. MapServer ofrece la posibilidad de realizar uniones entre capas, ésto es útil si se quiere visualizar información que esta almacenada en diferentes fuentes de datos. También permite configurar la visualización de capas dentro de un "*union*", según el zoom o extensión de la petición. Ésto se utiliza por ejemplo en el mapa base del modelo digital del terreno. En una petición de una extensión muy grande se devuelve una capa que hace referencia a un mapa con poca resolución, mientras que cuando se realiza una petición a la extensión del territorio catalán, se muestra la capa que tiene como origen de datos una mapa con más definición.

Los ficheros MapFile contienen una serie de elementos, en una estructura que se puede entender como una representación abstracta de un mapa. La configuración y los elementos de estos ficheros son muchos y muy diversos. Un ejemplo de esta estructura puede ser la siguiente:

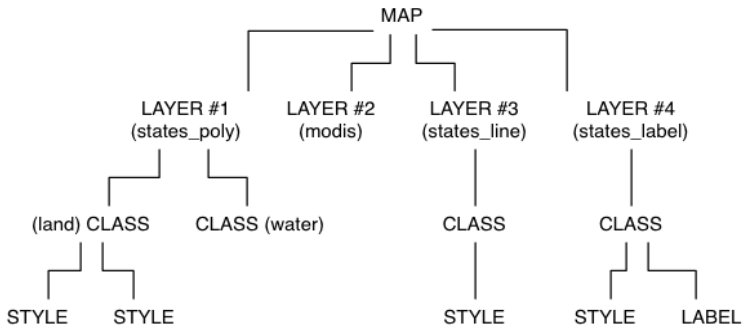


Ilustración 28: Ejemplo estructura MapFile

http://www.mapserver.org/tutorial/example1-5.html#example1-5

Ésta sería una estructura básica de un MapFile. Y es similar a la que se utilizará en la configuración del servidor.

Todos las capas o layers configuradas en el sistema tienen en común los siguientes elementos de configuración:

- **Proyección de salida**: es la proyección mediante la cual se visualizarán los datos. En la primera versión del sistema todas las capas tienen configuradas la proyección epsg:4326 con el datum WGS84.
- **Formato de salida de la operación GetFeatureInfo:** con valor text/html. Indica el formato de salida para la acción *GetFeatureInfo*.
- **Operaciones**: se permiten en todas las capas las operaciones *GetMap*, *GetCapabilites*, GetFeatureInfo y GetLegendGraphic. Ésto se configura mediante el elemento

*ows_enable_request = **.

9.3.2 Estructuración de los ficheros MapFiles

MapServer permite incluir ficheros dentro de otros MapFiles. Ésto permite evitar generar ficheros muy extensos y también posibilita reutilizar los ficheros. Ésta característica se realiza mediante *includes,* que apuntan a otros ficheros MapFile(que contienen parte de un fichero MapFile). Cuando MapServer haga la lectura del fichero principal (service_[categoria].map), internamente lo que hará será montar un único MapFile con la información de todos los ficheros incluidos(este montaje lo hace en memoria, de hecho todos los MapFile los carga en memoria tengan includes o no). Dada la gran cantidad de capas a configurar en el servidor, se utilizarán estos elementos *includes* para estructurar los MapFile en el servidor. Por norma general se crearan los siguientes *includes* para cada capa:

- **Service_[categoria].map**: Contiene todas las layers de un servicio
    - **map_[categoria].map:** Contiene la configuración genérica de las capas
        - **layer_[nombre_layer_1].map:** Configuración de la capa layer_1
            - **paleta_[nombre_paleta].map:** Configuración paleta
            - **paleta_[nombre_paleta].map:** Configuración paleta
        - **layer_[nombre_layer_2].map:** Configuración de la capa layer_2
            - **paleta_[nombre_paleta].map:** Configuración paleta
            - **paleta_[nombre_paleta].map:** Configuración paleta

A modo de ejemplo a continuación se muestra y explica la configuración del servicio de radar, configurado en el fichero *service_xrad_includes.map.* El resto de servicios y capas se configuran de forma similar.

9.3.2.1 Fichero SERVICE(Service_[categoria].map):

Fichero que recopila y estructura todos los ficheros que se han de incluir en el servicio. Primero se especifica un fichero *map*, después las *layers* y para cada una de éstas las leyendas o paletas que utiliza. Hay que tener en cuenta que siempre que se termina la definición de un elemento LAYER, hay que añadir un END. Ésto se puede ver al final de la especificación de cada paleta mediante la inclusión del *include end_layer.map,* que no es más que un fichero con un END.

A continuación se muestra el contenido del fichero *service_xrad.map*, fichero del servicio de capas de radar.

```
MAP

        INCLUDE  "./includes/maps/map_xrad.map"
################ Capa del producto CAPPI
### Radar Compuesto
        INCLUDE  "./includes/layers/xrad/CAPPI/layer_CAPPI_COMP_CE0m.map"
        INCLUDE  "./includes/paletes/xrad/Reflectivitat/xrad_reflectivitat_colorscomu.map"
        INCLUDE  "./includes/paletes/xrad/Reflectivitat/xrad_reflectivitat_colorsweb.map"
        INCLUDE  "./includes/layers/end_layer.map"

################# Capa del producto RAIN1H
### Radar acumulación precipitación en 1 hora
        INCLUDE  "./includes/layers/xrad/RAIN1H/layer_RAIN1H_COMP_CE.map"
        INCLUDE  "./includes/paletes/xrad/Pluja/xrad_pluja_EHIMI_NotesPremsa.map"
        INCLUDE  "./includes/paletes/xrad/Pluja/xrad_pluja_colorsTDT.map"
        INCLUDE  "./includes/layers/end_layer.map"

        INCLUDE  "./includes/layers/xrad/RAIN1H/layer_RAIN1H_COMP_CEXEMA.map"
        INCLUDE  "./includes/paletes/xrad/Pluja/xrad_pluja_EHIMI_NotesPremsa.map"
```

```
        INCLUDE  "./includes/paletes/xrad/Pluja/xrad_pluja_colorsTDT.map"
        INCLUDE  "./includes/layers/end_layer.map"

##################Capa del producto RAIN24H
### Radar corregido con XEMA acumulación precipitación en 1 hora
        INCLUDE  "./includes/layers/xrad/RAIN24H/layer_RAIN24H_COMP_CEXEMA.map"
        INCLUDE  "./includes/paletes/xrad/Pluja/xrad_pluja_EHIMI_NotesPremsa.map"
        INCLUDE  "./includes/paletes/xrad/Pluja/xrad_pluja_colorsTDT.map"
        INCLUDE  "./includes/layers/end_layer.map"
END
```

9.3.2.2 Fichero MAP

Este fichero incluye parámetros que son comunes a todas las capas del servicio. Los parámetros más importantes son:

- **NAME**: indica el nombre del servicio.
- **OUTPUFORMAT**: especifica un formato de salida para las imágenes generadas por el servicio.
- **OWS_SRS:** proyecciones que se pueden utilizar para servir las capas configuradas en el servicio. Hay que tener en cuenta que la proyección también se puede especificar individualmente en la capa y será prioritario.

El contenido del fichero *map_xrad.map*, es el siguiente:

```
NAME service_xrad
        SIZE 512 362
        STATUS ON

        PROJECTION
          "init=epsg:25831"
        END

        OUTPUTFORMAT
                NAME png
                DRIVER 'GD/PNG'
                MIMETYPE 'image/png'
                IMAGEMODE RGBA
                EXTENSION 'png'
                transparent ON
        END
        WEB
                IMAGEPATH '/tmp/'
                IMAGEURL 'http://localhost/tmp/'
                TEMPLATE "blank.html"
                METADATA
                        ######################### OWS
                        ows_title                       "Radar"
                        ows_abstract                    "Imatges Radar"
                        ows_keywordlist                 "Radar"
                        ows_contactorganization         "Servei Meteorològic de Catalunya"
                        ows_city                        "Barcelona"
                        ows_country                     "España"
                        ows_enable_request          "*"
                        ows_srs                         "EPSG:4326 EPSG:23031 EPSG:900913"
                        ows_encoding                    "UTF-8"
                        ######################### WMS
                        wms_feature_info_mime_type      "text/html"
                        wms_getmap_formatlist               "image/png,image/jpeg,image/gif,image/png"
                END
        END
```

9.3.2.3 Fichero Layer

Contienen la información especifica para cada capa. Una de las cosas más importantes a tener en cuanta es la forma en la que se configura el fichero que se consulta en cada petición. En la mayoría de las capas la información está almacenada en ficheros, que están almacenados en una estructura concreta y su nombre contiene la fecha válida para el dato.

Para las capas que tengan como origen de datos un fichero, en el atributo DATA, donde se

especifica el origen de los datos para la capa, se indica mediante un patrón con una serie de variables que indican la ubicación del fichero, por ejemplo:

```
'/DADES/RADAR/TIFF/CAP/CAPPI_0m/XXX/%any%/%mes%/%dia%/XXX_CAP_%any%%mes%%dia%_%hora%%minut
%_CAPPI_0m.tif'.
```

En la petición WMS, se especifican los parámetros que se substituirán en el patrón. En la ilustración 29 se muestra un ejemplo en forma de diagrama:

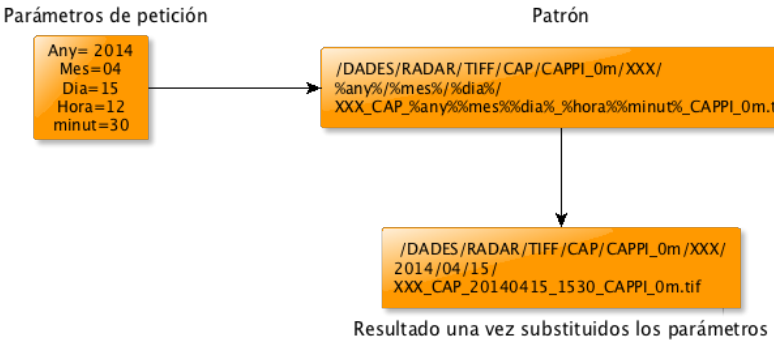


Ilustración 29: Esquema de substitución de parámetros

En el caso que el origen de los datos sea una base de datos, como es el caso de las capas del servicio de rayos, se especifica la ubicación de la base de datos y la query que se realiza para obtener los datos Utilizando las variables que se especifican por parámetro.

Un ejemplo de una query que se realiza en la layer de rayos seria la siguiente:

```
…....[ Configuración del resto de párametros].....

        CONNECTIONTYPE postgis
        INCLUDE  "/var/www/html/mapserver/mapfile/includes/connections/conn_xdde.map"

        DATA "the_geom FROM (
                SELECT xdd_secidx,xdd_coord_latlon as the_geom, xdd_signal, xdd_time, xdd_flags,
                        xdd_extended, xdd_chi_square, xdd_ell_semimajor_axis,
                        xdd_ell_semiminor_axis, xdd_ell_angle, xdd_risetime, xdd_maxraterise,
                xdd_freedom, xdd_multi, xdd_num_dfrs, xdd_cm_ine_municipi, xdd_is_cg, to_number(to_char(xdd_time,'HH24'),'99') as hora, oid
as sequence
                FROM xdde.xdde_dades_descarregues
                WHERE xdd_time>=timestamp without time zone '%any%-%mes%-%dia% 00:00:00'
                        AND xdd_time<timestamp without time zone '%any%-%mes%-%dia% 00:00:00'  + interval '1440 minute'
                        AND xdd_is_cg is true
                        AND xdd_ell_semimajor_axis < 10000
                        AND xdd_chi_square < 5
                        AND %any%<>'1000'
                        UNION  SELECT xdd_secidx, xdd_coord_latlon as the_geom, xdd_signal, xdd_time, xdd_flags,
                        xdd_extended, xdd_chi_square, xdd_ell_semimajor_axis, xdd_ell_semiminor_axis,
                xdd_ell_angle, xdd_risetime, xdd_maxraterise, xdd_freedom, xdd_multi,
        xdd_num_dfrs, xdd_cm_ine_municipi, xdd_is_cg, to_number(to_char(xdd_time,'HH24'),'99') as hora,oid as sequence
                                FROM xdde.xdde_dades_descarregues
                                WHERE xdd_time>=now() AT TIME ZONE 'GMT'
                                        AND xdd_time<now() AT TIME ZONE 'GMT'  + interval '1440 minute'
                                        AND xdd_is_cg is true
                                        AND xdd_ell_semimajor_axis < 10000
                                        AND xdd_chi_square < 5
                                        AND  %any%='1000') AS relampagos USING UNIQUE sequence USING SRID=4326"

        TYPE POINT
```

Los parámetros más importantes del elemento LAYER son:

- **NAME**: nombre de la layer. Éste nombre se ha de especificar en cada petición.
- **wms_feature_info_mime_type:** indica el formato en el que se devolverá la información cuando se haga una petición por GetFeatureInfo.
- **OWS_SRS:** proyección en las que se puede visualizar la capa.
- **EXTENT**: extensión total de la capa.
- **PROJECTION**: proyección de los datos de origen.
- **DATA**: patrón del origen de los datos.
- **TYPE**: indica la tipología en la que se encuentran o se recogen los datos. La mayoría de las capas tendrán valor RASTER, excepto la capa de rayos donde este parámetro tiene valor POINT; las capas administrativas que tienen como valor POLIGON; y las capas de modelos que muestran la información mediante isolíneas que tiene como valor LINE.
- **OWS_SRS**: proyección en las que se puede visualizar la capa.

Las capas del servicio de modelos que muestran la información de campos de presión mediante isolíneas, también tienen los elementos, PROCESSING y CONNECTIONTYPE, para indicar que los datos se visualizarán mediante curvas de nivel:

```
PROCESSING      "CONTOUR_ITEM=value_0"          # valor a tener en cuenta
PROCESSING      "CONTOUR_INTERVAL=400"          # cada cuanto se pinta una curva
CONNECTIONTYPE          CONTOUR         #indica que se quieren pintar contornos.
```

Como ejemplo, a continuación se muestra el contenido del fichero *layer_CAPPI_0m.map* a modo de ejemplo:

```
LAYER
        NAME 'CAPPI_COMP_CE0m'
        TYPE RASTER
        DUMP true
        TOLERANCEUNITS meters

        HEADER                          "templates/getfeatureinfo_header.html" # header html template
        TEMPLATE                        "templates/getfeatureinfo_content_radar.html" # content html template
        FOOTER                          "templates/getfeatureinfo_footer.html" # footer html template

        METADATA
                ows_title               "CAPPI Composada a 0 metres"
                ows_abstract    "CAPPI composada curt/llarg abast amb correccions EHIMI. Dades 6-minutals"
                ows_enable_request      "*"
                ows_extent              "51994.000 4273378.000 739994.000 4961378.000"
                ows_srs                         "EPSG:4326 EPSG:23031 EPSG:25831 EPSG:900913"

                wms_feature_info_mime_type  "text/html, application/vnd.ogc.gml"
                wms_include_items       "value_0"
                gml_constants           "UNITATS"
                gml_unitats_type        "Character"
                gml_unitats_value       "dBz"
                gml_value_0_alias        "VALOR"

                wcs_label                "Radar/GeoTiff"  ### required
                wcs_rangeset_name       "Range"
                wcs_rangeset_label      "value"

                'default_any'                   '2013'
                'default_mes'                   '07'
                'default_dia'                   '01'
                'default_hora'                  '20'
                'default_minut'                 '00'
        END
        VALIDATION
                'any'                           "[A-Z0-9._%+-|'\(\)]"
                'mes'                           "[A-Z0-9._%+-|'\(\)]"
```

```
                    'dia'                                   "[A-Z0-9._%+-|'\(\)]"
                    'hora'                                  "[A-Z0-9._%+-|'\(\)]"
                    'minut'                                 "[A-Z0-9._%+-|'\(\)]"
                END
                PROJECTION
                  "init=epsg:25831"
                END
                EXTENT 51994.000 4273378.000 739994.000 4961378.000
                DATA        '/DADES/RADAR/TIFF/CAP/CAPPI_0m/XXX/%any%/%mes%/%dia%/XXX_CAP_%any%%mes%%dia%_%hora%
%minut%_CAPPI_0m.tif'
                STATUS OFF
                PROCESSING "BANDS=1"
                PROCESSING "NODATA=OFF"
```

9.3.2.4 Fichero paleta

En este fichero se configura la leyenda de visualización. En cada leyenda se especifica con que color se pinta en la imagen un píxel según su valor. Se pueden especificar un valor o un rango de valores. La sección en negrita del ejemplo que se muestra a continuación indica que todos los píxels que tengan un valor entre 15 y 20 (ambos sin incluir) se han de pintar con el color RGB 0 255 255 (color azul claro). En las leyendas también se puede especificar una simbología en lugar de un color, como pasa en la capa de rayos.

```
        CLASSGROUP "colorscomu"
                CLASS
                        NAME "Fora de rang"
                        GROUP "colorscomu"
                        EXPRESSION ([pixel] = -9999)
                        STYLE
                                COLOR 154 154 154
                                OPACITY 40
                        END
                END
                CLASS
                        NAME "15 to 20 (dBZ)"
                        GROUP "colorscomu"
                        EXPRESSION ([pixel] > 15 AND [pixel] < 20)
                        STYLE
                                COLOR 0 255 255
                        END
                END
# [Se especifica para cada rango de valor del fichero un color, se ha acortado los rango intermedios ]
                .....................
                CLASS
                        NAME ">65 (dBZ)"
                        GROUP "colorscomu"
                        EXPRESSION ([pixel] >= 65 )
                        STYLE
                                COLOR 255 255 255
                        END
                END
        END
```

9.3.3 Organización de los ficheros MapFile

Se organizarán las capas en diferentes MapFile según su temática o categoría de producto. Por lo tanto en un MapFile estarán configuradas varias capas de una misma categoría. La configuración también permitirá ofrecer mediante la acción *GetCapabilities*, los metadatos de todas las capas de una misma categoría. Dentro del proyecto, utilizaremos el concepto *servicio de capas*, al conjunto de capas configuradas en un mismo MapFile.

El fichero donde se agrupan las capas es el fichero *service_[categoria].map*. Éste es el fichero que se indicará en las peticiones WMS. Todos los ficheros *service*, se ubican en */var/www/html/mapserver/mapfile/*. Un ejemplo de petición para una capa de Meteosat sería el siguiente:

```
Http://10.116.16.133/cgi-bin/mapserv.fcgi?VERSION=1.1.1&REQUEST=GetMap&SERVICE=WMS&STYLES=VIS_006&BBOX=-
0.175781,44.824219,22.675781,67.675781&WIDTH=520&HEIGHT=520&FORMAT=image/png&SRS=EPSG:4326&LAYERS=MSG3_
VIS_006&MAP=/var/www/html/mapserver/mapfile/service_meteosat_includes.map &minut=00&hora=15&dia=02&mes=05&any=2014
```

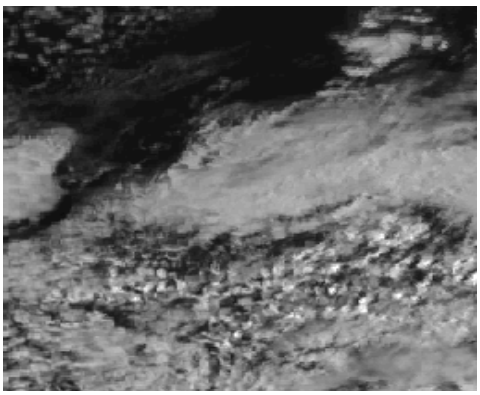


Ilustración 30: Resultado petición Meteosat

El resultado de la petición anterior daría como resultado la imagen que muestra la ilustración 30. En el ejemplo mostrado anteriormente, en negrita podemos ver, que en la petición se incluye el MapFile *service_meteosat_includes.map*, que es el fichero donde están configuradas todas las capas de Meteosat. Como en la petición es obligatorio indicar el nombre de la capa (MSG3_VIS_006), MapServer lo que hará será ir al fichero *service_meteosat_includes.map*, recorrer este fichero y construir un MapFile(en memoria) añadiendo el contenido de todos los incluyes que tenga el fichero, y buscar el layer MSG3_VIS_006.

Se organizan los siguientes servicios:

- **Servicio capas administrativas:**
  Configurado en el fichero *service_administratiu_includes.map*. Agrupa todas las capas que hacen referencia a información de tipo administrativo o referencia, como pueden ser límites de países, comarcas, municipios, carreteras, etc.
- **Servicio capas temáticas:**
  Configurado en el fichero *service_tematiques_includes.map*. Agrupa todas las capas estáticas que se usan como referencia a nivel meteorológico, como pueden ser medias anuales de alguna variable.
- **Servicio capas interpolaciones:**
  Configurado en el fichero *service_interpolacio_includes.map*. Agrupa todas las capas que hacen referencia a productos de algún tipo de interpolación, descritos en el apartado Productos interpolados.
- **Servicio capas Meteosat:**
  Configurado en el fichero *service_meteosat_includes.map*. Agrupa todas las capas que hacen referencia a los productos derivados de la información proporcionada por el satélite de Meteosat, descritos en el apartado Productos Meteosat
- **Servicio capas radar:**
  Configurado en el fichero *service_xrad_includes.map*. Agrupa todas las capas que hacen referencia a los productos que se generan a partir de la información proporcionada por los radares del SMC. Descritos en el apartado Productos radar.
- **Servicio capas Modelos:**
  Configurado en el fichero *service_models_includes.map*. Agrupa todas las capas que hacen referencia a modelos de predicción numéricos. Descritos en el apartado Productos de modelos.

- **Servicio capas rayos:**

Configurado en el fichero *service_xdde_includes.map*. Agrupa todas las capas que hacen referencia a información sobre las descargas eléctricas.El origen de datos de estas capas es una base de datos PostgreSQL.

Aclarar que todos los archivos MapFile y sus respectivos *includes* han de tener permisos de lectura y ejecución.

9.3.4 MapScript

En el apartado anterior se ha explicado que la configuración del servidor se realiza a partir de ficheros MapFile. Ésto es verdad, aunque no es la única forma. La configuración mediante MapFiles es una configuración que se puede denominar estática. Ésto quiere decir que en tiempo de ejecución no cabe la posibilidad de realizar ningún tipo de modificación en el fichero MapFile. Ésto puede ser un handicap dependiendo de la información que se quiera mostrar. Para solventar esta carencia, MapServer ofrece la posibilidad de crear o modificar MapFiles en tiempo de ejecución mediante MapScript. MapScript es una librería a modo de API que permite a los desarrolladores acceder a ciertas funcionalidades internas de MapServer. Cuando MapServer carga o lee un fichero MapFile, no hace más que cargar en memoria el fichero en una estructura de objetos. Mediante MapScript, se puede acceder a parte de la librería que trabaja con estos objetos, y por lo tanto se puede manipular una MapFile o crear uno nuevo en tiempo de ejecución. MapServer tiene diferentes módulos en forma de API que permite implementar estos scrips en diferentes lenguajes de programación como Python, PHP, Java, Perl, etc. En el sistema necesitaremos realizar algún calculo o modificación de MapFiles en tiempo de ejecución, y ésto se realizará mediante MapScripts implementados en Python.

En la petición WMS a MapServer se ha de indicar el MapFile donde se encuentra configurada la capa. En el caso de la ejecución de MapScripts lo que se tiene que indicar es la ubicación del MapScript a ejecutar. Un ejemplo seria el siguiente, que daría como resultado la imagen de la ilustración 31:

http://10.116.16.133/cgi-bin/models_bandes_FeatureInfo.py?VERSION=1.1.1&REQUEST=GetMap&SERVICE=WMS&STYLES=temperaturaSup&BBOX=11.140137,44.890137,16.984863,50.734863&WIDTH=532&HEIGHT=532&FORMAT=image/png&SRS=EPSG:4326&LAYERS=WRF27_TMP_2-HTGL&abast=00&sortida=00&dia=05&mes=05&any=2014

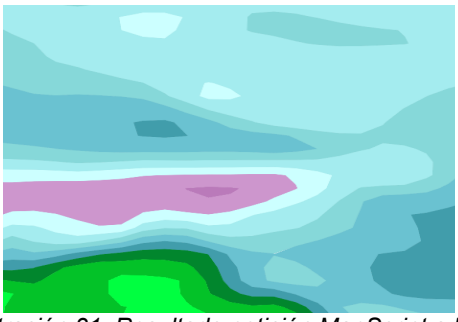


Ilustración 31: Resultado petición MapScript a la capa models

Aclarar que queda fuera del alcance de este proyecto la creación de estos scripts. Por lo tanto únicamente se comentará los MapScripts utilizados, para qué capas se utilizan y qué realizan los scripts.

A continuación se describen los scripts utilizados por el sistema, todos los scripts están ubicados en la carpeta */var/www/cgi-bin*, para que apache pueda ejecutarlos:

- **models.py**: las capas de modelos numéricos del modelo WRF, recuperan la información de ficheros GRIB. Los ficheros contienen la información de diferentes variables, a diferentes alturas, en unas estructuras denominadas bandas. Por tanto, cuando se quiera acceder a una variable a una altura determinada, el sistema tendrá que acceder al fichero GRIB de una fecha determinada y recuperar la banda que haga referencia a la variable requerida. MapServer es capaz de leer un fichero GRIB y recuperar una banda determinada. Pero la generación de los modelos no siempre guarda las variables en las mismas bandas, de forma que no se puede saber a priori en qué banda está configurada cada variable. Para solventar este problema se ha creado un MapScript que a partir del nombre de la variable, abre el fichero GRIB y recupera el numero de la banda que hace referencia a la información de esa variable. Una vez recuperada la banda, el script abre el MapFile(lo carga en memoria) y añade el número de banda.
- **models_vent.py**: En la visualización de la variable viento, lo que se requiere es poder visualizar las variables velocidad de viento y dirección en forma de barbas de viento[20]. Para poder generar una imagen con esta simbología, se necesita un MapScript que recupere la información de los modelos para las variables de viento, realice los cálculos necesarios y cree un fichero shapefile con la malla de los valores de velocidad y dirección del viento. Después realiza unos cálculos para generar la simbología necesaria en cada punto de la malla. La malla variará dependiendo de la extensión y del zoom requeridos. Un ejemplo de la imagene que genera el MapScript se puede observar en la ilustración 52.
- **models_bandes_FeatureInfo.py**: Los modelos de predicción tienen un paso de malla muy elevado, ésto quiere decir que cada pixel representa varios quilómetros. Ésto hace que la imagen generada tenga muy poca resolución y se vea la imagen pixelada. Para solventar este problema, se ha creado este script que realiza un suavizado(*vectorizando* la imagen) la información de los modelos. El suavizado se lleva a cabo creando una serie de isobandas[21](similares a las curvas de nivel), que permiten visualizar las imágenes suavizadas, tal y como se puede ver en las ilustraciones 32 y 33.

20 Símbolos que representan la velocidad y la dirección del viento.

21 Similar a una isolínea, pero donde la tipología en lugar de ser una linea es un polígono que engloba todos los valores dentro de un rango.

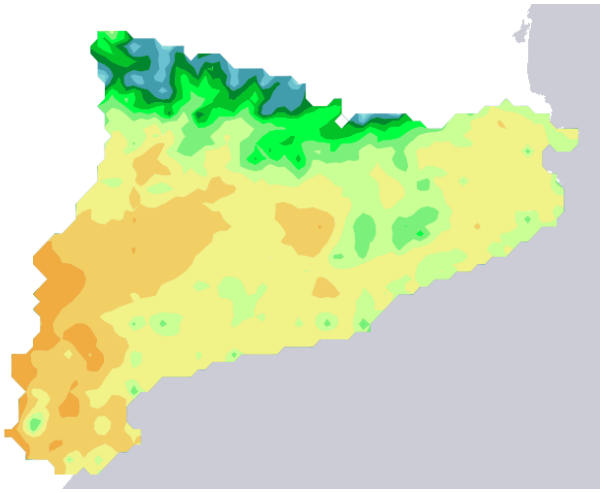


Ilustración 32: Imagen del modelo suavizado

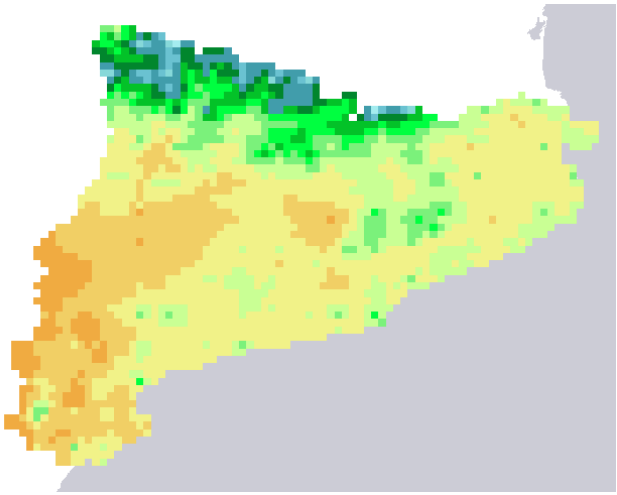


Ilustración 33: Imagen modelo original

9.4 Capas

En el componente servidor, se organiza y configura la información en forma de capas. Las capas se han de configurar tanto en MapServer como en MapCache. Para consultar las capas, el cliente deberá proveer en la petición los parámetros y acciones estándar del servicio utilizado(ya sea WMS o WMTS). A parte de estos parámetros estándar, cada capa puede estar configurada mediante diferentes parámetros necesarios para su consulta, como por ejemplo parámetros de fecha, minutos de acumulación o horizonte en el caso de los modelos.

En el siguiente apartado se describen las capas que se configuran en el sistema.

9.4.1 Capas temáticas

Son capas estáticas, es decir, que su información no varía. Las capas estáticas están pensadas para contener información meteorológica que no varía con el paso del tiempo, y si lo hace será en contadas ocasiones. En el caso que haya que modificar la información, solo habría que modificar el fichero de origen.

- **percentil P2**: se utiliza para establecer si una lectura de temperatura en los meses fríos están dentro de las mínimas esperadas.
- **percentil P98**: se utiliza para establecer si una lectura de temperatura en los meses de calor están dentro de las temperaturas máximas esperadas.

Configuración

Parámetros comunes a todas las capas:

- **Servicio**: service_tematiques_includes.map
- **Ubicación origen:** /DADES/RADAR/PRODUCTES/INTERPOLACIO/%any%/%mes%/%dia%/
- **Origen de datos:** Fichero
- **Tipo fichero**: GeoTIFF
- **Proyección**: epsg:23031
- **Tipología**: Raster

Capa	Tile	Producto	Fichero	Paletas
interpolacio_tempUltimes	interpolacio_tempUltimes_temperaturaSup	interpolacio_temp_ultimes	temperatura_%any%%mes%%dia%%hora%%minut%.tiff	temperaturaSup
interpolacio_hrUltimes	interpolacio_hrUltimes_hr	interpolacio_hr_ultimes	hr_%any%%mes%%dia%%hora%%minut%.tiff	hr

Tabla 18: Capas temáticas

9.4.2 Capas base

Son capas estáticas que contienen información cartográfica y que se utilizan como mapas base. Sobre estos mapas base es donde se mostrará la información. Las capas base, recuperarán la información de ficheros en formato GeoTIFF, ubicados en el mismo servidor. Dado que es información estática, no es necesario especificar ningún tipo de parámetro, a parte de los obligatorios por WMS o WMTS.

Se configurarán diferentes tipos de mapas base, con el objetivo de facilitar la visualización de cualquier tipo de dato. Los mapas que se configuran son los siguientes:

- **Mapa de fondo claro:** mapa con colores claros (blanco y gris) donde se muestra el continente de color blanco y el mar en color gris claro. éste mapa básico será útil cuando el usuario quiera superponer información con paletas de colores intensos.
- **Mapa de fondo oscuro:** igual que el anterior con la diferencia que los continentes y el mar se muestra con unos colores oscuros. Útil cuando se quiera superponer información con una paleta de colores claros.
- **Mapa con modelo digital de elevaciones**: mapa que muestra la orografía de la tierra, útil cuando se quiera visualizar el relieve de la tierra. Formado por diferentes ficheros GeoTIFF, que unificados forman un único mapa que muestra la orografía del terreno y ofrecen un efecto de relieve. También se han utilizado diferentes resoluciones dependiendo del zoom de visualización. De esta manera cuando el usuario esté en un zoom lejano la resolución será más baja, y cuando estemos en un zoom más cercano a Cataluña, se muestra el mapa con mayor resolución. Con ésto se gana en rendimiento, ya que los mapas con mayor resolución son más pesados. Otra característica de este mapa es la posibilidad de poder obtener la altura del terreno de un punto. Ofreciendo al usuario la posibilidad de hacer click en cualquier punto del mapa (únicamente en la zona de Cataluña) y obtener la altura en ese punto.

Configuración:

- **Ubicación origen:** /var/www/html/mapserver/datos/referencia/
- **Origen de datos:** Fichero
- **Tipo fichero**: GeoTIFF
- **Proyección**: epsg:4326
- **Tipología**: Raster

Capa	MapCache	Layer	Fichero
fonsclar	fonsclar	fonsclar_L	terra.tif

Capa	MapCache	Layer	Fichero
		fonsclar_zoom1_L	terrazoom1.tif
		fonsclar_zoom2_L	terrazoom2.tif
fonsfosc	fonsfosc	fonsfosc_L	terra.tif
		fonsfosc_zoom1_L	terrazoom1.tif
		fonsfosc_zoom2_L	terrazoom2.tif
fonsmde	fonsmde	batimetria	BatimetriaextT.tif
		Ombresmon	Ombresmon.tif
		ombresesp	Ombresesp.tif
		mdeext	Srtm30.tif
		ombrescat	sha90wgs84.tif
		mdecat	Srtm90wgs84.tif
		fonsclar_L	Terra.tif
		fonsclar_zoom2_L	terrazoom2.tif

Tabla 19: Capas de mapas base

9.4.3 Capas administrativas

Son capas estáticas que se utilizan como referencia en la visualización. Recuperan la información de ficheros shapefile. Para que la visualización de estas capas se pueda distinguir, se configurarán las capas para que puedan visualizarse con diferentes leyendas(paletas o estilos). En el sistema se configuran las siguientes capas de referencia:

- **Países**: capa que muestra los límites de cada país. La capa esta formada por la unión de tres capas distintas. Las capas contienen la misma información, pero a una resolución diferente, y se ofrece una u otra dependiendo de la extensión y el zoom demandado. Ésto se hace para agilizar la consulta, y se repite en otras capas similares que requieran visualizaciones de información muy extensa, como es el caso del modelo digital del terreno.
- **Municipios**: muestra el limite de los municipios de Cataluña. El usuario podrá consultar los nombres de los municipios haciendo click en uno de ellos. Ésto es posible dado que la información se recupera de un fichero shapefile que contiene la información propia para cada geometría, dispone del nombre del municipio. Dispone de diferentes leyendas para su visualización.
- **Comarcas**: muestra los límites de las comarcas de Cataluña. Al igual que la capa anterior, permite la visualización en diferentes colores y también consultar el nombre de la comarca.

Configuración

Parámetros comunes a todas las capas:

- **Servicio**: service_administrativas_includes.map
- **Ubicación origen:** /var/www/html/mapserver/datos/referencia/
- **Origen de datos:** Fichero

- **Tipo fichero**: shapefile
- **Proyección**: epsg:4326
- **Tipología**: Polygon (vectorial)

Capa	MapCache	Layer	Fichero	Paletas
paisos	- paisos_blanc - paisos_negre	- layer_administratiu_paisos.map • layer_administratiu_paisos1.map • layer_administratiu_paisos2.map • layer_administratiu_paisos3.map	./paisos1.shp ./paisos2.shp ./paisos3.shp	- linea blanca - linea negra
comarcas	- comarques_negre - comarques_blanc - comarques_gris	layer_administratiu_comarques.map	./comarca.shp	- blanco - negre - gris
municipis	- municipis_blanc - municipis_negre - municipis_gris	layer_administratiu_municipis.map	./municipis.shp	- blanco - negre - gris

Tabla 20: Capas administrativas

9.4.4 Capas de radar

Contienen la información de Productos radar.

Configuración:

- **Servicio**: service_xrad_includes.map
- **Ubicación origen:** /DADES/RADAR/TIFF/CAP/CAPPI_0m/XXX/*%any%/%mes%/%dia%/*
- **Origen de datos:** Fichero
- **Formato ficheros**: GeoTIFF
- **Parámetros**: *any, mes, dia, hora, minut*. Aclarar que para las capas horarias los minutos siempre sera '00', y para la capa diaria la hora y el minuto siempre sera '00'.
- **Proyección**: epsg:25831
- **Tipología**: Raster

Capa	Tile	Producto	Fichero	Paletas
CAPPI_COMP_CE0	CAPPI_COMP_CE0m_colorscomu CAPPI_COMP_CE0m_colorsweb	CAPPI 250m	XXX_CAP_%any%%mes%%dia%_%hora%%minut%_CAPPI_0m.tif	colorscomu colorsweb
RAIN1H_COMP_CE	RAIN1H_COMP_CE_EHIMI_NotesPremsa RAIN1H_COMP_CE_colorsTDT	RAIN1H_COMP_CE	XXX_RN1_%any%%mes%%dia%_%hora%%minut%_CMPAC1H_1h.tif	ColorsTDT EHIMI_NotesPremsa
RAIN1H_COMP_CEXEMA	- RAIN1H_COMP_CEXEMA_EHIMI_NotesPremsa - RAIN1H_COMP_CEXEMA_colorsTDT	RAIN1H_COMP_CEXEMA	XXX_RN1_%any%%mes%%dia%_%hora%%minut%_CMPAC1C_1h.tif	ColorsTDT EHIMI_NotesPremsa
RAIN24H_COMP	-	RAIN24H_COM	XXX_RNN_%any%	ColorsTDT

Capa	Tile	Producto	Fichero	Paletas
_CEXEMA	RAIN24H_COMP_CEXEMA_EHIMI_NotesPremsa - RAIN24H_COMP_CEXEMA_colorsTDT	P_CEXEMA	%mes%%dia%_%hora%%minut%_CMP24KG_24h.tif	EHIMI_NotesPremsa

Tabla 21: Capas de radar

9.4.5 Capas de Meteosat

Capas que contienen información de los productos generados por las imágenes del satélite de Meteosat(Productos Meteosat). Los productos se generan cada 15 minutos. El fichero del producto de Meteosat que se utiliza, es el mismo para las tres capas. El fichero contiene varias bandas, y cada banda hace referencia a un tipo de información.

Configuración

- **Servicio**: service_meteosat_includes.map
- **Ubicación origen:** '/DADES/METEOSAT/GEOTIFF/%any%/%mes%/%dia%/
- **Origen de datos:** Fichero
- **Formato ficheros**: GeoTIFF
- **Proyección**: "proj=merc" "lon_0=-0" "lat_ts=44" "x_0=0" "y_0=0" "ellps=WGS84" "units=m" "no_defs"
- **Parámetros**: any , mes, dia, hora , minut
- **Tipología**: Raster
- **Origen**: HRIT_MSG3_%any%-%mes%-%dia%_%hora%-%minut%.tif'

Capa	MapCache	Descripción	banda	Paletas
MSG3_IR_108	MSG3_IR_108_IRcolo	Imagen del infrarrojo térmico del Meteosat	4	MSG3_IRcolor
MSG3_WV_062	- MSG3_WV_062_paleta1 - MSG3_WV_062_paleta2	Imagen de vapor de agua del Meteosat	7	MSG3_VIS_006
MSG3_VIS_006	MSG3_VIS_006_VIS_006	Imagen visible del Meteosat	6	MSG3_WV_062

Tabla 22: Capas de Meteosat

9.4.6 Capas de Modelos de predicción

Capas que contienen la información de los modelos de predicción(Productos de modelos).En esta primera versión se configuran los modelos PRESCAT y WRF, para las variables y alturas más útiles para los técnicos del SMC. Dentro de cada fichero hay configuradas una serie de bandas que contienen la información para toda la extensión(nos referimos a extensión territorial) de una variable determinada. En los modelos PRESCAT, estas bandas son fijas, y por lo tanto se puede fijar la banda en el fichero MapFile. En cambio en los modelos WRF, las variables varían de banda en cada ejecución. Por este motivo se utiliza el MapScript models.py.

La visualización de las variables del modelo WRF, se pueden realizar a partir de los ficheros originales, o se pueden visualizar mediante un suavizado de los valores para evitar el pixelado del fichero original. El suavizado se realiza utilizando el MapScript models_bandes_FeatureInfo.py.

Existen un par de variables que tienen un comportamiento o visualización distinta al resto:

- **Viento**: la variable viento se visualiza mediante barbas de viento. Ésta visualización permite ver la velocidad y dirección del viento mediante símbolos.

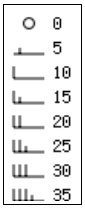


Ilustración 34: Barbas de viento de la leyenda de la variable viento

- **Presión**: variable que se visualiza mediante isolíneas. MapServer permite la visualización mediante lineas en forma de curvas de nivel. Las isolíneas de un mapa de presión, no dejan de ser curvas de nivel similares a los utilizados en un mapa de elevación, la única diferencia radica la separación entre las curvas. En las isolíneas se añade una etiqueta que indica el valor de presión de cada curva.

Configuración Modelo PRESCAT:

- **Servicio**: service_models_includes.map
- **Ubicación origen:** DADES/MODELS/PRESCAT/%sortida%.27/%any%/%mes%/%dia%/
- **Origen de datos:** Fichero
- **Formato ficheros**: GeoTIFF
- **Proyección**: epsg:4230
- **Parámetros**: any , mes, dia, sortida , abast.
- **Tipología**:
  - Raster: visualizado original
  - Vectorial: visualización suavizado
- **Origen**: PRESCAT-%any%%mes%%dia%%sortida%_%abast%.tiff

Capa	MapCache	Producto	Acceso	Tipologia	Paletas
PRESCAT_TMAX_SFC	PRESCAT_TMAX_SFC_temperaturaSup	PRESCAT_TMAX_SFC	MapFile	Raster	temperaturaSup
PRESCAT_TMIN_SFC	PRESCAT_TMIN_SFC	PRESCAT_TMIN_SFC	MapFile	Raster	temperaturaSup
PRESCAT_PRECIPITACIO_SFC	PRESCAT_PRECIPITACIO_SFC_probPrecip	PRESCAT_PRECIPITACIO_SFC	MapFile	Raster	probPrecip
PRESCAT_ESTATCEL_SFC	PRESCAT_ESTATCEL_SFC_estCel	PRESCAT_ESTATCEL_SFC	MapFile	Raster	estCel
Isobandas (suavizado)					
PRESCAT_TMAX_SFC	PRESCAT_TMAX_SFC_ISOBANDA_temperaturaSup	PRESCAT_TMAX_SFC	MapScript	Vectorial	temperaturaSup
PRESCAT_TMIN_SFC	PRESCAT_TMIN_SFC__ISOBANDA_temperaturaSup	PRESCAT_TMIN_SFC	MapScript	Vectorial	temperaturaSup
PRESCAT_PRECIPITACIO_SFC	PRESCAT_PRECIPITACIO_SFC_ISOBANDA_probPrecip	PRESCAT_PRECIPITACIO_SFC	MapScript	Vectorial	probPrecip
PRESCAT_ESTA	PRESCAT_ESTATCEL_SFC_	PRESCAT_ESTATCEL_SFC	MapScri	Vectorial	estCel

Capa	MapCache	Producto	Acceso	Tipologia	Paletas
TCEL_SFC	ISOBANDA_estCel		pt		

Tabla 23: Capas modelo PRESCAT

Configuración Modelo WRF:

- **Servicio**: service_models_includes.map
- **Ubicación origen:** /DADES/MODELS/WRF/run.%sortida%.27/%any%/%mes%/%dia%/
- **Fichero origen**: wrf-27.%any%%mes%%dia%%sortida%_%abast%.grib
- **Origen de datos:** Fichero
- **Formato ficheros**: GRIB
- **Proyección**: "proj=longlat" "a=6371200" "b=6371200" "no_defs"
- **Parámetros**: any , mes, dia, sortida , abast.

Capa	MapCache	Producto	Acceso	Tipologia	Paletas
WRF27_TMP_2-HTGL	WRF27_TMP_2-HTGL_temperaturaSup	WRF27_TMP_2-HTGL	MapFile	Raster	temperaturaSup
WRF27_TMP_500-ISBL_	WRF27_TMP_500-ISBL_temperatura500hP	WRF27_TMP_500-ISBL	MapFile	Raster	emperatura500hP
WRF27_TMP_850-ISBL	WRF27_TMP_850-ISBL_temperatura850hP	WRF27_TMP_850-ISBL	MapFile	Raster	temperatura850hP
WRF27_RH_2-HTGL	WRF27_RH_2-HTGL_hr	WRF27_RH_2-HTGL	MapFile	Raster	Hr
WRF27_RH_500-ISBL	WRF27_RH_500-ISBL_hr	WRF27_RH_500-ISBL	MapFile	Raster	hr
WRF27_RH_850-ISBL	WRF27_RH_850-ISBL_hr	WRF27_RH_850-ISBL	MapFile	Raster	hr
WRF27_MCDC_0-SFC	WRF27_MCDC_0-SFC_nuvolositat	WRF27_MCDC_0-SFC	MapFile	Raster	nuvolositat
	WRF27_MCDC_0-SFC_nuvolositat8			Raster	nuvolositat8
WRF27_HGT_ISOLINEA_500-ISBL	WRF27_HGT_ISOLINEA_500-ISBL_negre	WRF27_HGT_500-ISBL	MapScript	Vectorial	negre
	WRF27_HGT_ISOLINEA_500-ISBL_blanc			Vectorial	blanc
WRF27_HGT_ISOLINEA_850-ISBL	WRF27_HGT_ISOLINEA_850-ISBL_negre	WRF27_HGT_850-ISBL	MapScript	Vectorial	negre
	WRF27_HGT_ISOLINEA_850-ISBL_blanc			Vectorial	blanc
WRF27_PRMSL_ISOLINEA_0-MSL	WRF27_PRMSL_ISOLINEA_0-MSL_blanc	WRF27_PRMSL_0-MSL	MapScript	Vectorial	blanc
	WRF27_PRMSL_ISOLINEA_0-MSL_negre			Vectorial	negre

Capa	MapCache	Producto	Acceso	Tipologia	Paletas
WRF27_UGRD_VGRD_10-HTGL	WRF27_UGRD_VGRD_10-HTGL_negre	WRF27_UGRD_VGRD_10-HTGL	MapScript	Vectorial	negre
	WRF27_UGRD_VGRD_10-HTGL_blanc			Vectorial	blanc
WRF27_UGRD_VGRD_500-ISBL	WRF27_UGRD_VGRD_500-ISBL_negre	WRF27_UGRD_VGRD_500-ISBL	MapScript	Vectorial	negre
	WRF27_UGRD_VGRD_500-ISBL_blanc			Vectorial	blanc
WRF27_UGRD_VGRD_850-ISBL	WRF27_UGRD_VGRD_850-ISBL_negre	WRF27_UGRD_VGRD_850-ISBL	MapScript	Vectorial	negre
	WRF27_UGRD_VGRD_850-ISBL_blanc			Vectorial	blanc
Isobandas (suavizado)					
WRF27_TMP_2-HTGL	WRF27_TMP_2-HTGL_ISOBANDA_temperaturaSup	WRF27_TMP_2-HTGL	MapScript	Raster	temperaturaSup
WRF27_TMP_500-ISBL_	WRF27_TMP_500-ISBL_ISOBANDA_temperatura500hP	WRF27_TMP_500-ISBL	MapScript	Raster	emperatura500hP
WRF27_TMP_850-ISBL	WRF27_TMP_850-ISBL_ISOBANDA_temperatura850hP	WRF27_TMP_850-ISBL	MapScript	Raster	temperatura850hP
WRF27_RH_2-HTGL	WRF27_RH_2-HTGL_ISOBANDA_hr	WRF27_RH_2-HTGL	MapScript	Raster	Hr
WRF27_RH_500-ISBL	WRF27_RH_500-ISBL_ISOBANDA_hr	WRF27_RH_500-ISBL	MapScript	Raster	hr
WRF27_RH_850-ISBL	WRF27_RH_850-ISBL_ISOBANDA_hr	WRF27_RH_850-ISBL	MapScript	Raster	hr
WRF27_MCDC_0-SFC	WRF27_MCDC_0-SFC_nuvolositat	WRF27_MCDC_0-SFC	MapScript	Raster	nuvolositat
	WRF27_MCDC_0-SFC_nuvolositat8			Raster	nuvolositat8

Tabla 24: Capa modelo WRF27

9.4.7 Capas de rayos

Capas que contienen información de las descargas eléctricas(Producto rayos). La información de las descargas eléctricas está almacenada en una base de datos PostgreSQL. En los parámetros de consulta, a parte de la fecha, se indica un parámetro acumulación. Que sirve para especificar los minutos de acumulación de la consulta. La consulta de datos se realiza mediante una query SQL usando las funciones proporcionadas por PostGIS.

Configuración

- **Servicio:** service_xdde_includes.map

- **Origen de datos:** Fichero
- **Proyección**: epsg:4326
- **Parámetros**: any , mes, dia, hora , minut, acum
- **Tipologia**: Vectorial

Capa	Tile		Paletas
llampscg_sm	llampscg_sm_color	Descargas nube–tierra sin multiplicidad.	color
	llampscg_sm_negre	Descargas nube–tierra sin multiplicidad.	negre
	llampscg_sm_4intervals	Descargas nube–tierra sin multiplicidad en 4 periodos.	4intervals
Llampscg_sm_el	Llampscg_sm_el_color	Descargas nube–tierra sin multiplicidad visualizado con elipsoide.	color
	Llampscg_sm_el_negre	Descargas nube–tierra sin multiplicidad visualizado con elipsoide.	negre
	Llampscg_sm_el_4intervals	Descargas nube–tierra sin multiplicidad visualizado con elipsoide de error y en 4 periodos.	4intervals
Llampscg_sm_diari	Llampscg_sm_diari_12intervals	Descargas nube–tierra sin multiplicidad diarias visualizado en 12 inervalos.	12intervals

Tabla 25: Capas rayos

9.4.8 Capas de interpolaciones

Capas que contienen información de productos interpolados(Productos interpolados) En la primera versión se configuran dos capas de las variables temperatura y humedad relativa, que se generan a partir de los datos de la XEMA. Las interpolaciones tienen en cuenta la elevación del terreno y la cercanía al mar, para realizar correcciones. Cada 30 minutos se recuperan los últimos datos de las estaciones para la cada variable y se realiza la interpolación.

Configuración

- **Servicio:** service_interpolacio_includes.map
- **Ubicación origen:** /DADES/RADAR/PRODUCTES/INTERPOLACIO/%any%/%mes%/%dia%/
- **Origen de datos:** Fichero
- **Formato ficheros**: GeoTIFF
- **Proyección**: init=epsg:23031
- **Parámetros**: any , mes, dia, hora , minut
- **Tipología**: Raster

Capa	Tile	origen	Paletas
interpolacio_tempUltimes	interpolacio_tempUltimes_temperaturaSup	temperatura_%any%%mes%%dia%%hora%%minut%.tiff	temperaturaSup
interpolacio_hrUltimes	interpolacio_hrUltimes_hr	hr_%any%%mes%%dia%%hora%%minut%.tiff	hr

Tabla 26: Capas interpolación

10 Capa cliente

Como ya se comentó en otros apartados para la implementación del visor se utilizarán las librerías *Dojo Toolkit* y *OpenLayers*. La aplicación se implementará en una única página web. En ella se mostrará un mapa de fondo, y una serie de componentes o módulos en forma de paneles o menús, que permitirán al usuario interactuar con la aplicación.

Para la visualización y gestión de las capas, se usaran objetos de la librería OpenLayers. Al cargar una capa WMTS, OpenLayers realizará las llamadas necesarias vía WMTS para componer la imagen.

10.1 Visualización linea temporal

Como se ha comentado en varias ocasiones, la variable tiempo y la manera de visualizar los datos en una linea temporal es muy importante. El visor abstrae al usuario de la necesidad de conocer la forma en la que se etiqueta cada producto y también de la necesidad de conocer la temporalidad de cada producto o capa. El usuario introducirá la fecha que desea consultar, y será el visor quien realice los cálculo necesarios para realizar la petición al servidor con la fecha correcta. Un ejemplo gráfico del manejo de la temporalidad de las capas, se puede ver en la ilustración 41.

El visor trabajará con fechas mediante el estándar UTC.

10.2 Fichero de configuración

El visor dispone de una serie de ficheros de configuración. Éstos ficheros permiten agilizar ciertos aspectos de la implementación y el despliegue en producción.

- **Configuracio.js:** fichero javascript que contiene variables de carácter estático a modo de constantes. Contiene diferente tipo de información, desde el nombre de los servicios de capas del servidor de mapas(ficheros *service_[categoria]_includes.map*), zoom inicial del mapa, proyecciones, parámetros de configuración del mapa y del las capas, etc.
- **ConfiguracioURL.js**: fichero con variables de tipo constante, que contiene las direcciones URL, de MapCache, MapServer y la API RET del SMC. Éste fichero se modifica automáticamente mediante un script a la hora de desplegar la aplicación en los diferentes entornos del SMC(entorno de pruebas y producción).
- **Ficheros de productos**: ficheros en formato JSON, que se utilizan para configurar los valores de los diferentes formularios. Existe un fichero de configuración para cada categoría. Los formularios contienen una serie de elementos(combos, radio buttons, etc.), que se rellenan automáticamente con la información almacenada en estos ficheros. Por ejemplo en el formulario de modelos, el elemento que muestra las diferentes variables, obtienen el nombre y su valor a partir de este fichero. De esta forma si se quiere añadir una nueva variable solo se ha de modificar este fichero y no haría falta modificar el código. Ésto hace que la aplicación se ha flexible y fácil de mantener. Ademas contiene el intervalo de tiempo para los diferentes productos para poder calcular la fecha válida a consulta al servidor a partir de cualquier fecha. A continuación se muestra el contenido del fichero de configuración de las capas de radar.

```
{
   "title" : "Fitxer configuració de radar",
   "items":[
      {
         "nom":"CAPPI 0m",
         "id":"CAPPI_COMP_CE0m",
         "interval" : 360
      },
      {
```

```
        "nom":"PPT 1h corregida EHIMI amb XEMA",
        "id":"RAIN1H_COMP_CEXEMA",
        "interval" : 3600,
        "etiquetatOriginal" : "DARRERE"
    },
    {
        "nom":"PPT 1h corregida EHIMI",
        "id":"RAIN1H_COMP_CE",
        "interval" : 3600,
        "etiquetatOriginal" : "DARRERE"
    },
    {
        "nom":"PPT 24h corregida EHIMI amb XEMA",
        "id":"RAIN24H_COMP_CEXEMA",
        "interval" : 86400,
        "etiquetatOriginal" : "DARRERE"
    }]
}
```

10.3 Dojo Toolkit

Dojo funciona mediante módulos o *widgets*, estos componentes permiten agrupar ciertas funcionalidades en un único módulo. Estos módulos utilizan páginas HTML a modo de *template* para visualizarse por pantalla, tal y como podemos observar en la ilustración 35. Dojo permite que el desarrollador cree nuevos módulos, y también permite la herencia entre los módulos y de esta extender los módulos existentes en el core de Dojo.

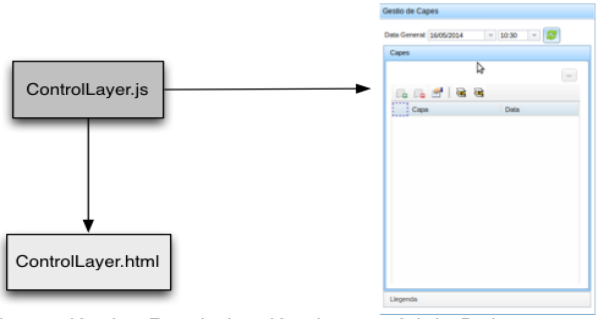


Ilustración 35: Renderización de un módulo Dojo

La aplicación se organizará mediante módulos, que agruparán funcionalidades similares. Los módulos que se visualicen por pantalla, accederán a ficheros template, que se usarán como plantilla para la visualización. Éstos módulos interaccionarán entre si para ofrecer todas la funcionalidades de la aplicación. Se puede hacer la equivalencia entre módulos y objetos en los lenguajes orientados a objetos. Aunque hay que aclarar que javascript no es un lenguaje orientado a objetos al uso.

10.3.1 Patrón subscribe/publish

Dojo nos ofrece la posibilidad de utilizar el patrón *subscribe/publish*. El patrón ofrece la posibilidad de desacoplar los objetos o módulos. Un módulo se subscribe a un evento determinado, normalmente la subscripción va asociada a la llamada a una función, un elemento realiza la tarea de mediador y almacena esta subscripción. Cuando en alguna parte del código se realiza una publicación(publish) para ese evento, el objeto mediador notifica a todos los subscriptores de ese evento que sea realizado una publicación. Entonces cada subscriptor realiza la tarea especificada en la subscripción, que normalmente es la ejecución de una función. Este patrón se utiliza en varios puntos de la aplicación para facilitar la independencia de los módulos y de esta manera desacoplar los módulos. Sobre todo en los módulos que realizan la función de controladores, como son *Mapa.js, PanelCapes.js*, etc. En la ilustración 36, se muestra el diagrama de secuencia del funcionamiento del patrón.

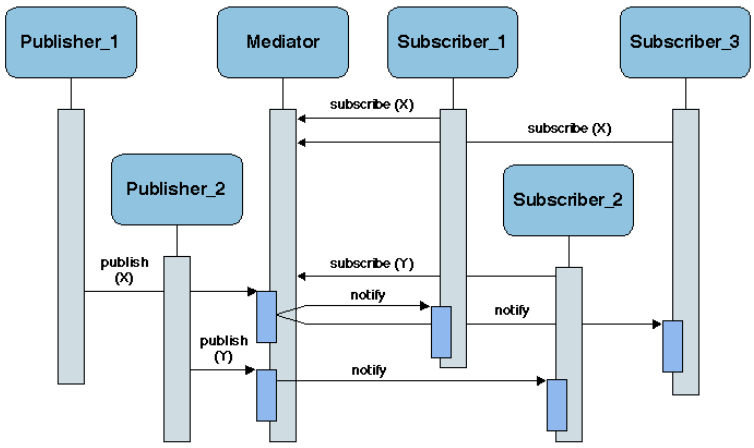


Ilustración 36: Diagrama de secuencia del patrón publish/subscribe

http://proton.inrialpes.fr/~krakowia/MW-Book/Chapters/Events/events.html

10.3.2 Diagrama conceptual

A continuación se muestra el diagrama conceptual de los módulos implementados en el visor.

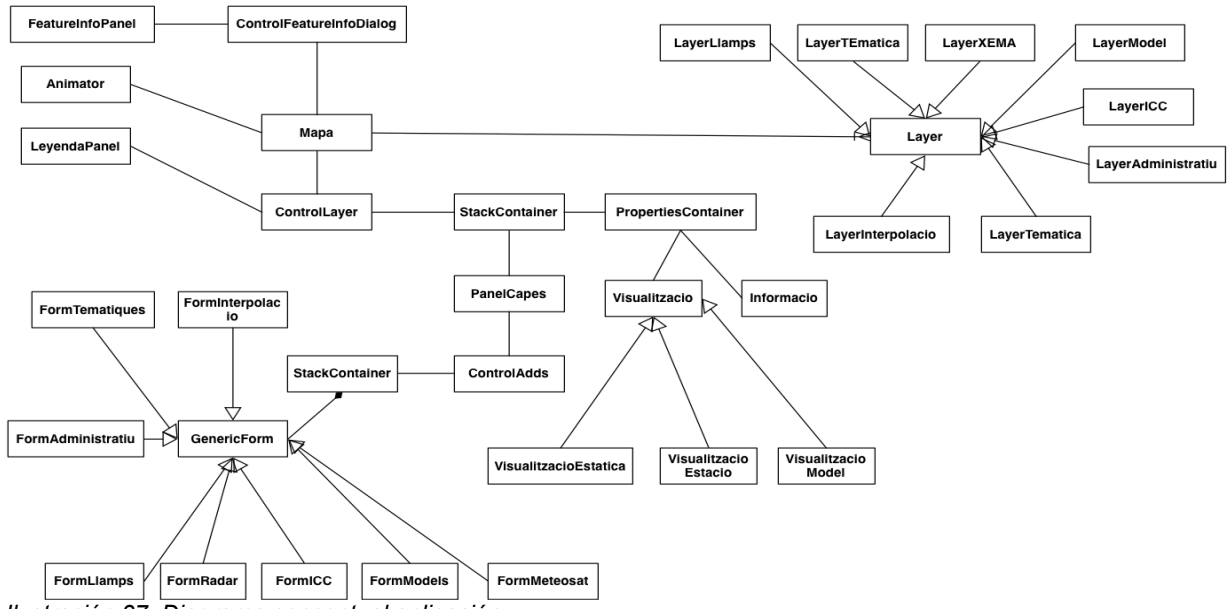


Ilustración 37: Diagrama conceptual aplicación

10.4 Módulos

Los módulos son los componentes que se han implementado y que agrupan las funcionalidades y la visualización de los paneles del visor. A continuación se describen los más importantes.

10.4.1 Mapa.js

Módulo principal de la aplicación. Uno de sus atributos es un objeto *OpenLayers.Map*, que es el objeto que visualiza la información. La función principal del módulo *Mapa.js*, es la de gestionar las

acciones sobre el mapa(*OpenLayers.Map*). Ésto lo realiza mediante el patrón *publish/subscribe*, subscribiéndose a una serie de mensajes. Cuando algún módulo realiza un *publish* sobre alguno de estos mensajes, el módulo *Mapa.js* realiza las acciones pertinentes sobre el mapa o cualquier otro módulo que contenga. Las subscripciones se agrupan dentro de la función *_inicialitzaSubscripcions*().

El módulo contiene una referencia de todos los módulos Layer.js que se han cargado. Con la finalidad de poder modificar este objeto cuando el usuario realiza alguna acción que lo requiera.

A continuación se muestra el contenido del método *_ inicialitzaSubscripcions().*

```
/**
        * Subscriu el widget als topics on publiquen els diferents components.
        * D'aquesta forma es redueix i es simplifica la comunicacio entre components.
        */
       _inicialitzaSubscripcions: function() {
           topic.subscribe("visor/capa/afegir", lang.hitch(this, this._afegirCapa));
           topic.subscribe("visor/capa/esborrar", lang.hitch(this, this._esborrarCapa));
           topic.subscribe("visor/capa/moure", lang.hitch(this, this._moureCapa));
           topic.subscribe("visor/capa/visibilitat", lang.hitch(this, this._canviarVisibilitatCapa));
           topic.subscribe("visor/capa/opacitat", lang.hitch(this, this._canviarOpacitat));
           topic.subscribe("visor/capa/data", lang.hitch(this, this._actualitzaDataCapa));
           topic.subscribe("visor/capa/zoomExtent", lang.hitch(this, this._zoomExtentCapa));
           topic.subscribe("visor/capa/style", lang.hitch(this, this._canviaStyleCapa));

           // Subscribim listener per detectar canvis de data en l'animacio
           topic.subscribe("visor/animacio/data", lang.hitch(this, this._actualitzaDataCapes));
       },
```

10.4.2 PanelCapes.js

Módulo que gestiona las capas dentro de la lista de capas que se visualizan. Éste módulo, entre otras funciones realiza la inserción/eliminación de capas, orden de visualización y activación/desactivación de capas. Cuando se realiza cualquiera de estas acciones el módulo emite un evento *publish* para el mensaje que corresponda a la acción, enviando la capa y los parámetros necesarios para que el subscriptor, que en este caso seria el módulo *Mapa.js*, realice las acciones pertinentes. A parte el módulo también realiza una subscripción al mensaje "onAfegirCapa", para añadir una capa a la lista. La emisión del evento publish para el mensaje “onAfegirCapa” la realiza el módulo *ControlAdds*.

Al cargarse una capa en el listado, se muestra un intervalo de validez, que informa al usuario de las fechas para la cual es válida la información que muestra capa. La selección y visualización de las capas están etiquetadas por detrás.

En la ilustración 38 podemos ver el diagrama de secuencia de la funcionalidad de añadir capa.

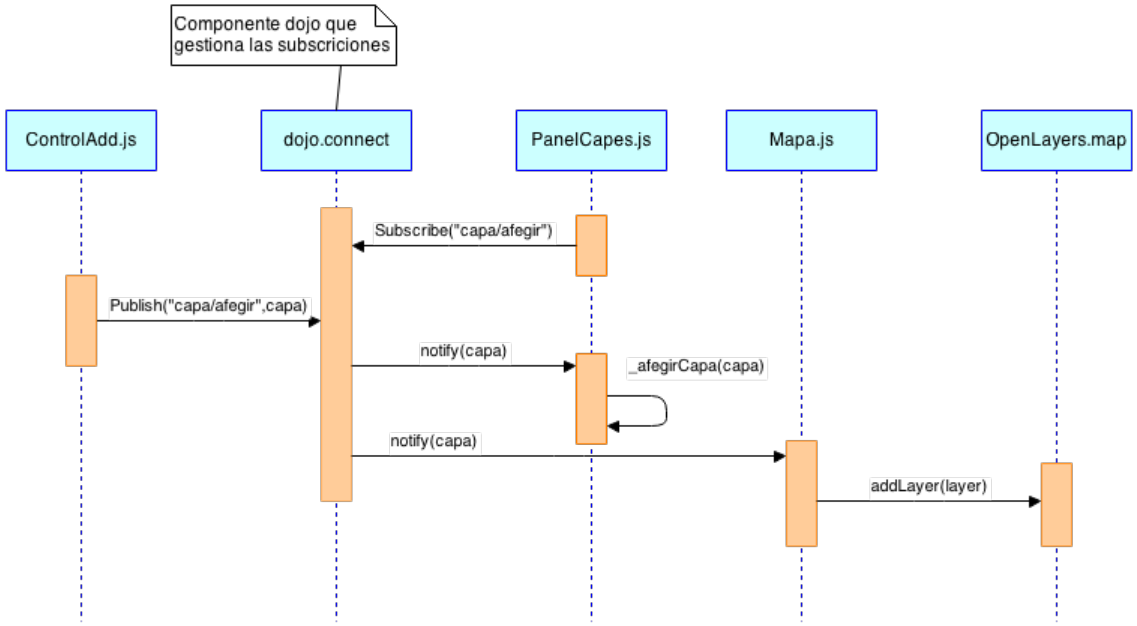


Ilustración 38: Diagrama de secuencia de la acción "añadir capa"

10.4.3 GenericForm.js

El componente *ControlAdds* gestiona los formularios. El componente contienen un elemento combo que muestra las diferentes categorías(o servicios de mapas) que se pueden consultar. El componente contienen un *stackContainer*(pila de componentes, que muestra el elemento que esta al inicio de la pila) de componentes formulario. Cada vez que se selecciona una categoría mediante el combo de *categoría* se visualiza el componente formulario de esa categoría, ofreciendo la posibilidad de añadir un nuevo formulario. *ControlAdds* también contiene el botón que genera las capas. Cuando hace click en el botón aceptar, el componente hace una llamada al método *getCapa()* del componente formulario, que crea una nueva *Layer*.

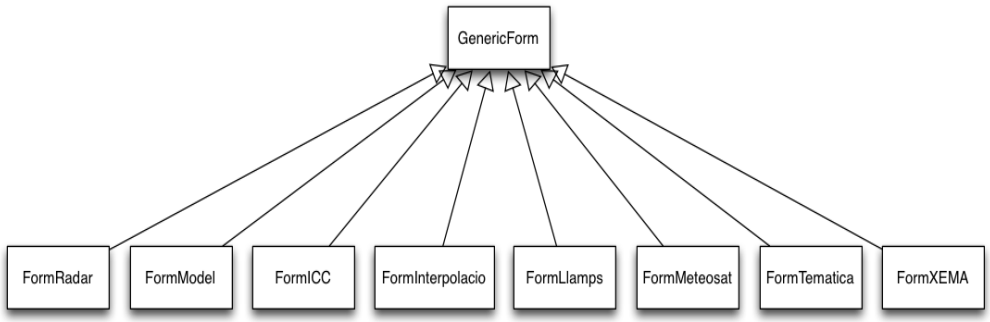


Ilustración 39: Jerarquía formularios

Los componentes formulario se utilizan para cargar las capas en el visor. Los componentes heredan de un componente superior que contiene funciones y métodos genéricos, ilustración 39. En cada formulario se pueden cargar capas de una misma categoría (servicio de capas en MapServer). Los campos del formulario se utilizan para construir el nombre de la capa o los parámetros a incluir en la petición. Por ejemplo, el campo fecha se utilizará para crear los parámetros fecha y el componente leyenda para crear el nombre de la leyenda a consultar. Dado que las peticiones para las capas son diferentes entre ellas, los componentes Formulario disponen

de campos distintos según el servicio de capas que se quiera consultar.

Un ejemplo de ésto se puede ver con la ilustración 40. La construcción de la petición de la capa se haría de la siguiente forma:

Partimos que el patrón del nombre de una capa de modelos dentro de MapCache es el siguiente: *[nombre_capa]_[variable]_[nivel]_[leyenda].*

- Campo "producte": aquí se indica el nombre de la capa. Valor de [nombre_capa]
- Campo variable: indica la variable a visualizar. Valor de [variable]
- Campo nivell: altura. Valor de [nivel]
- El campo leyenda, también formará parte del nombre de la capa que se pedirá. Recordar que la petición se hace a MapCache, y la leyenda forma parte del nombre de la capa.

En el ejemplo de la ilustración 40 el nombre de la capa que se construiría seria: *WRF27_TMP_2HTGL_temperaturaSup*.

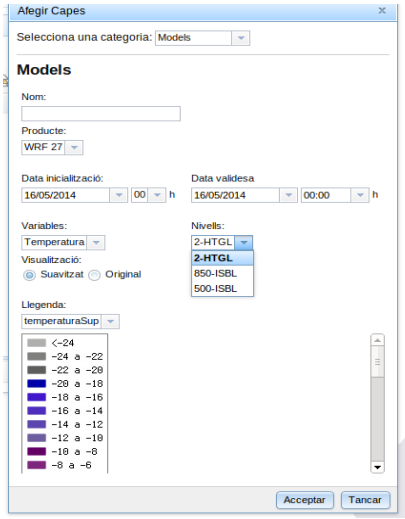


Ilustración 40: Formulario modelos. Construcción nombre de capa

Para hacer más dinámicos los formularios, las leyendas se rellenan automáticamente, sin necesidad de utilizar un fichero de configuración o especificándolas en el código de forma estática. Para rellenar la información de las leyendas se utiliza el fichero de metadatos que se obtiene mediante la llamada a un servició de capas mediante a la operación *GetCapabilities* del servicio WMS. La operación devuelve la información de las capas y las posibles leyendas para cada capa. OpenLayers ofrece la posibilidad de realizar la llamada a esta operación.

El método *getCapas()* realiza las acciones necesarias para recopilar la información del formulario y crear una *Layer*, con la información necesaria para que el componente *Mapa* pueda hacer la llamada al servidor.

10.4.3.1 FormEstacions.js

Tal y como se ha comentado anteriormente, la aplicación carga la información para la capa XEMA a partir de la API REST del SMC. La API ofrece un recurso para consultar todas las lecturas de las estaciones de la red XEMA para una fecha determinada. Como la capa de estaciones no esta configurada en el servidor de mapas, el formulario no puede obtener la leyenda del servidor. Por tanto la leyenda para el formulario de la XEMa se creará en el mismo módulo a partir del fichero de configuración, mediante javascript. La información en el mapa también se muestra de forma distinta que el resto de capas. Mientras que el resto de capas se muestra información mediante

una imagen, este formulario crea una capa vectorial a partir de la información que obtiene de la API.

10.4.3.2 FormICC.js

El formulario *FormICC.js* carga capas del ICGC, y tiene un comportamiento diferente al resto de formularios. Este formulario no muestra la opción de leyenda. El formulario ofrece la posibilidad de cargar tres capas del ICGC. El método *getCapa()* del formulario crea una capa WMS que se conecta directamente a los servidores del ICGC.

10.5 Animacio.js

El módulo Animacio.js, se renderiza en un panel flotante que contiene las funcionalidades necesarias para realizar la animación de la información. La animación consiste en establecer un intervalo de fechas, para las que se quiere realizar la animación. El panel contiene un cuadro de botones de animación que permite iniciar/para la animación, ir al intervalo anterior/siguiente e ir al inicio/final del intervalo de animación.

Cada vez que la animación cambia de fecha, el módulo emite un evento publish para el mensaje “visor/animacio/data”, para indicar a los subscriptores que las fechas de las capas han de modificarse. La fecha de validez(es decir fecha que se quiere visualizar), es la misma para todas las capas. Ésto supone que para cada fecha de la animación, cada una de las capas activas ha de calcular la fecha correcta, mediante el mismo método (*Layer._dataValidaConsultaGetter(data)*) que se utiliza para calcular la fecha de la capa al añadir una capa nueva. Los principales subscriptores son Mapa.js y PanelCapes.js. El primero accede a su propio listado de módulos Layer cargados en el mapa, calcula la nueva fecha a consultar para cada capa mediante un método del módulo Layer, y luego actualiza las layers del objeto OpenLayers.Map. El segundo actualiza el intervalo de fechas de validez para cada capa en el listado de capas.

10.6 Layer.js

El módulo Layer.js se crea al cargar una capa nueva. Este módulo es quien contiene el objeto *OpenLayer.Layer* que finalmente se añade al mapa mediante el patrón *publish/subscriber*. Por tanto se podría decir que es un envoltorio del objeto *OpenLayer.Layer*. El módulo *Layer* contiene un método importante en el sistema, que calcula a partir de una fecha, la fecha válida que se le ha de pasa al servidor de mapas, para que coincida con una fecha válida para el fichero que contiene la información de la capa a cargar. Hay que aclarar que para abstraer al usuario de conocer que fechas puede consultar para cada capa, la aplicación deja que el usuario informe el campo fecha del formulario con cualquier fecha. Por ejemplo, al cargar una capa del producto de radar *CAPPI* que se genera cada 6 minutos, la aplicación no puede hacer una petición para la fecha 01/05/2014 12:02h. Ya que no existe ningún fichero del producto *CAPPI* que tenga como nombre esa hora. Lo que hace el módulo *Layer* en este caso es calcular cual es la fecha del fichero que contiene la información válida para las 12:02h, es decir la fecha anterior más cercana. En este caso seria el fichero de las 12:00, que contiene la información del producto para el intervalo de las 12:00h – 12:06h. Para realizar el cálculo utiliza la información del fichero de configuración de capas anteriormente descrito. Éste ejemplo queda representado en la ilustración 41.

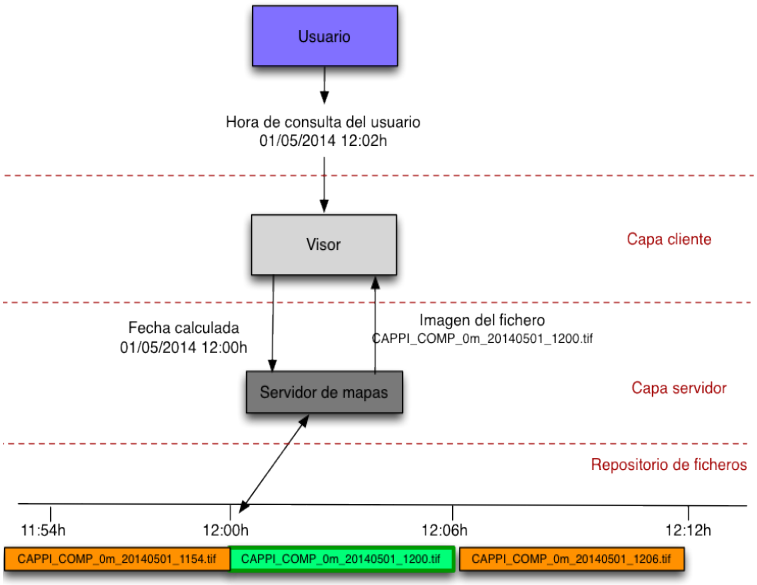


Ilustración 41: Uso de fechas en el visor

Todos los formularios heredan del módulo *GenericLayer*, éste contiene los métodos necesarios para crear un objeto *OpenLayer.Layer.Wmts*. Se ha establecido unos criterios genéricos para la creación de capas, y los métodos del módulo *GenericLayer* pueden crear una *Layer* de tipo WMTS siguiendo estos criterios. En el caso que la capa a cargar siga un comportamiento distinto a la hora de crear una layer, se crea un nuevo módulo para dicha capa que hereda de GenericLayer. Como por ejemplo *LayerModels*, que sobrescribe el método que calcula la fecha y los parámetros de consulta, puesto que se ha de calcular el alcance de la predicción para una fecha de predicción.

10.6.1 LayerVector.js

Carga las lecturas de las estaciones mediante el objeto *OpenLayers.Layer.Vector*. La información que se obtiene de la API, es un JSON con la información de las lecturas de todas las estaciones. El módulo *LayerVector* recorre la información obtenida y crea objetos *OpenLayers.Feature.Vector* de tipo punto, para cada lectura. Por lo tanto se crea una capa de tipo vector que contiene features de tipo punto con las lecturas de las estaciones. Antes de crear los objetos Feature, se realiza una reproyección del punto, dado que la visualización se realiza mediante la proyección epsg:4326 y las coordenadas de las estaciones que ofrece la API están en la proyección epsg:23031. Para aplicar la leyenda a los datos se utiliza una regla de estilo que se aplica para cada feature de la capa.

Otro aspecto importante de este módulo es la posibilidad de agrupar las features en clusters. Un cluster es la agrupación de ciertos elementos en uno solo. La agrupación permite que la visualización de las lecturas sea más limpia ya que permite agrupar en una sola feature un grupo de features(valores de las estaciones) próximas entre si. La diferencia de visualización se puede observar en las ilustraciones 42 y 43.

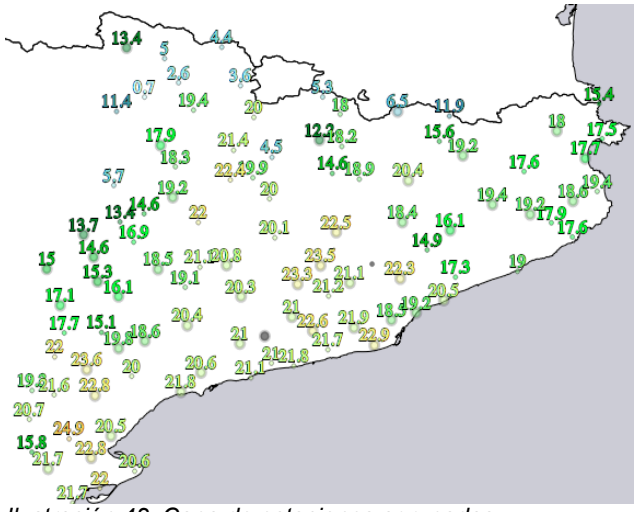


Ilustración 42: Capa de estaciones agrupadas

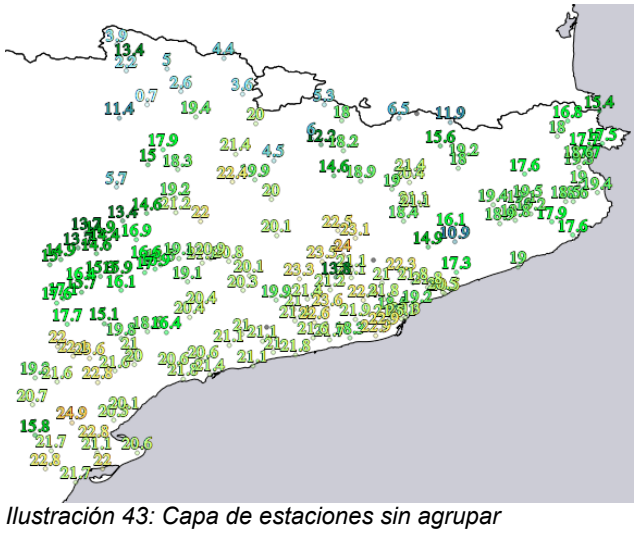


Ilustración 43: Capa de estaciones sin agrupar

10.6.2 LayerICC.js

El módulo LayerICC, tiene la particularidad que crea un objeto layer de tipo WMS en lugar de uno de tipo WMTS, como el resto de capas. Otra característica es que carga la información directamente de los servidores públicos del ICGC, sin pasar por el servidor de mapas implementado en el proyecto.

La creación de layers WMS se realiza mediante la siguiente función:

```
_creaWmtsLayer: function() {
        this.olLayer = new OpenLayers.Layer.WMS("Topo ICC",
            "http://mapcache.icc.cat/map/bases_noutm/service?",
            {
                transparent: true,
                layers: this.nomWmsCapa,
                format: "image/png",
                exceptions: "application/vnd.ogc.se_xml"
            },
            {
                buffer: 0,
                transitionEffect: 'resize',
                opacity: 1,
                singleTile: true}
        );
    },
```

10.7 Interfaces y funcionalidades

A continuación se describen las diferentes funcionalidades e interfaces del visor.

10.7.1 Pantalla de inicio

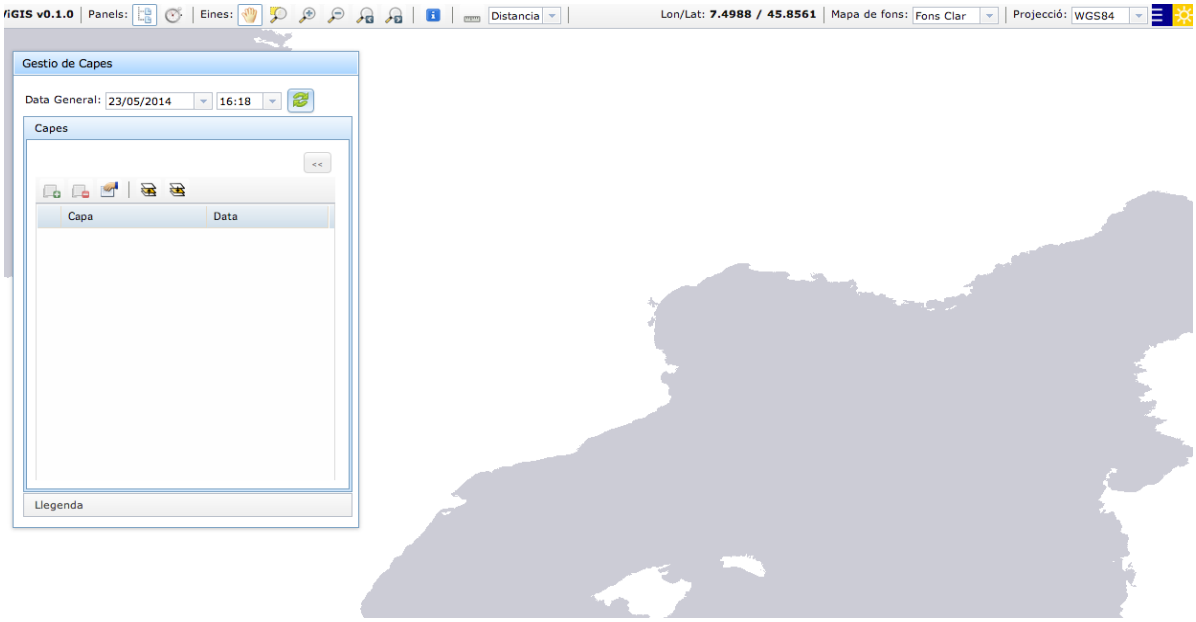


Ilustración 44: Interfaz inicial

En la interfaz inicial se muestra el panel flotante de gestión de capas y en la parte superior un menú fijo con las funcionalidades genéricas de la aplicación, tal y como muestra la ilustración 44. En el menú nos encontramos con botones agrupados en tres grupos: *Panels*, que ocultan/muestran los paneles flotantes de gestión de capas y animación; *Eines*, que ofrecen básicamente funcionalidades de navegación como zooms, activar navegación, histórico de zooms, etc; un botón para consultar la información de un punto; un botón para realizar medidas de distancias o áreas. Después encontramos un desplegable que donde se selecciona el mapa de fondo, y por último un desplegable para seleccionar la proyección de visualización.

10.7.2 Formulario de capas

Desde el panel de gestión de capas se accede a los formularios que permiten cargar las capas. Las capas se agrupan en diferentes grupos según su temática y características. Al pulsar en el botón de añadir capa se abre un panel flotante que muestra un formulario por defecto y un desplegable. El desplegable permite al usuario seleccionar la categoría que quiere consultar. En función de la selección de la categoría se muestra un formulario u otro. Dentro de cada formulario se muestran diferentes campos que variarán según la categoría. En las capas dinámicas (capas meteorológicas) existen varios campos que estarán presentes en todos los formularios: campo fecha, campo producto y campo leyenda. Al aceptar el formulario, la aplicación recopila los datos informados en los campos y construye un objeto OpenLayers.Layer que contendrá la información necesaria para realizar la petición al servidor.

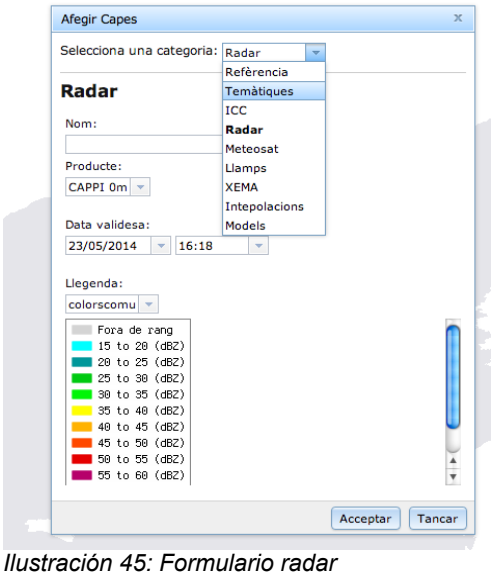


Ilustración 45: Formulario radar

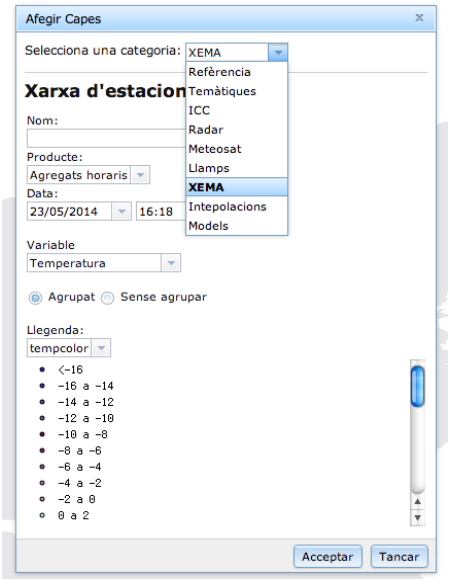


Ilustración 46: Formulario XEMA

Cada vez que se acepta el formulario se añade una capa a la lista de capas del panel "*Gestió de capes*", y también se visualizará en el mapa. Ésto se puede ver en la ilustración 47. La lista de capas muestra un intervalo de fechas que indican el intervalo de tiempo para el cual son válidos los datos de la capa.

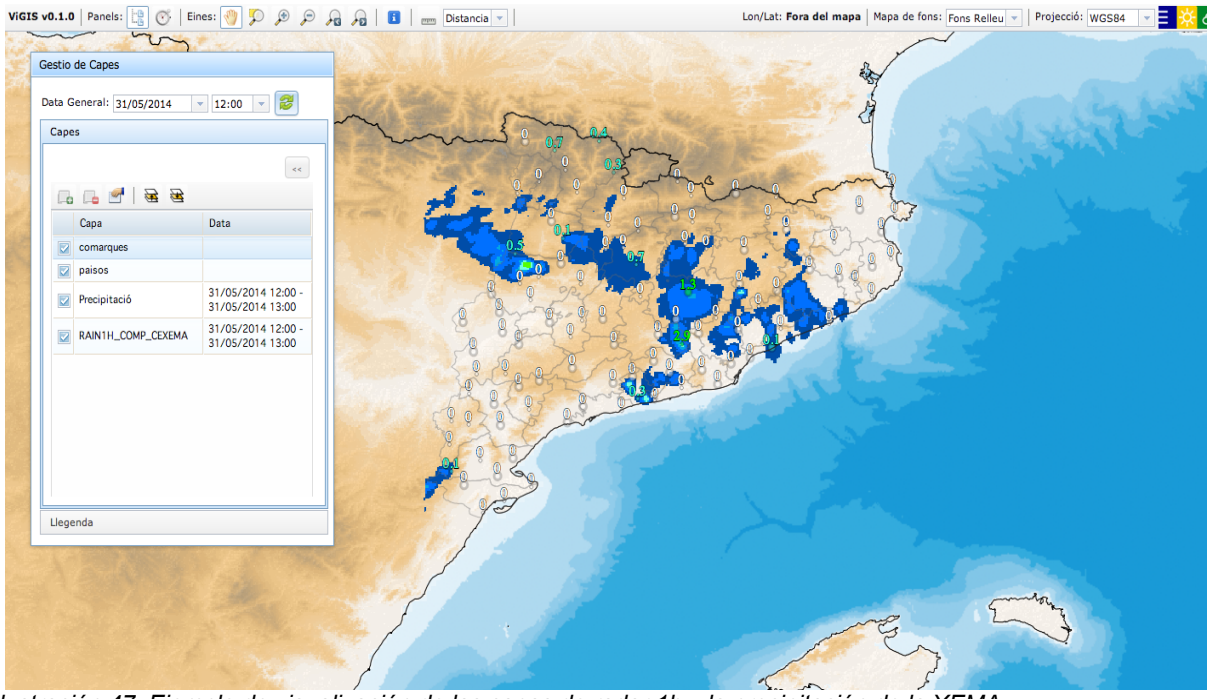


Ilustración 47: Ejemplo de visualización de las capas de radar 1h y la precipitación de la XEMA.

En la ilustración 49 podemos ver una imagen de la capa de rayos, que se obtiene como resultado de al cargar la información del formulario que se muestra en la ilustración 48.

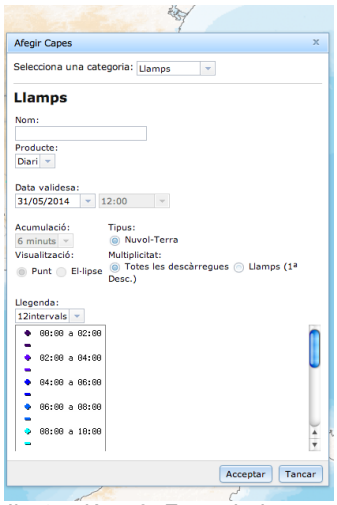


Ilustración 48: Formulario para cargar capa rayos 24h

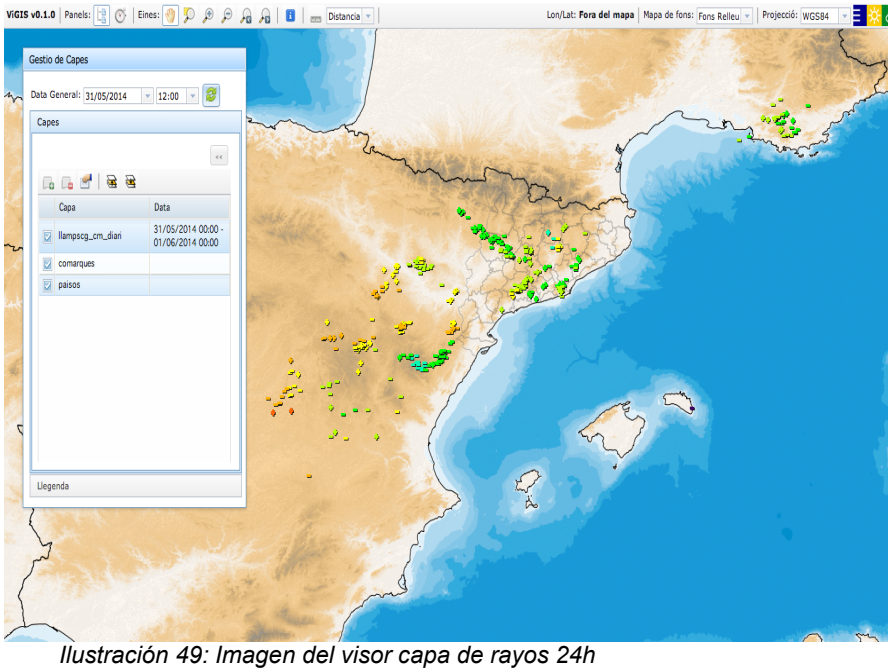


Ilustración 49: Imagen del visor capa de rayos 24h

10.7.3 Información de un punto

Mediante el botón  ubicado en el menú superior, se puede consultar la información alfanumérica de las capas que se están visualizando. En el ejemplo de la ilustración 50, se puede ver la información de un punto de las capas cargadas. Cuando se selecciona cualquier punto de mapa, la aplicación muestra la información en un panel flotante de todas las capas cargadas que permiten la acción de consulta a través de un punto(*GetFeatureInfo*). Cabe recordar que para la capa base de relieve, a esta resolución también se puede consultar la altura en ese punto. La ubicación del punto se visualiza mediante un punto no muy visible para no entorpecer la visualización de los datos.

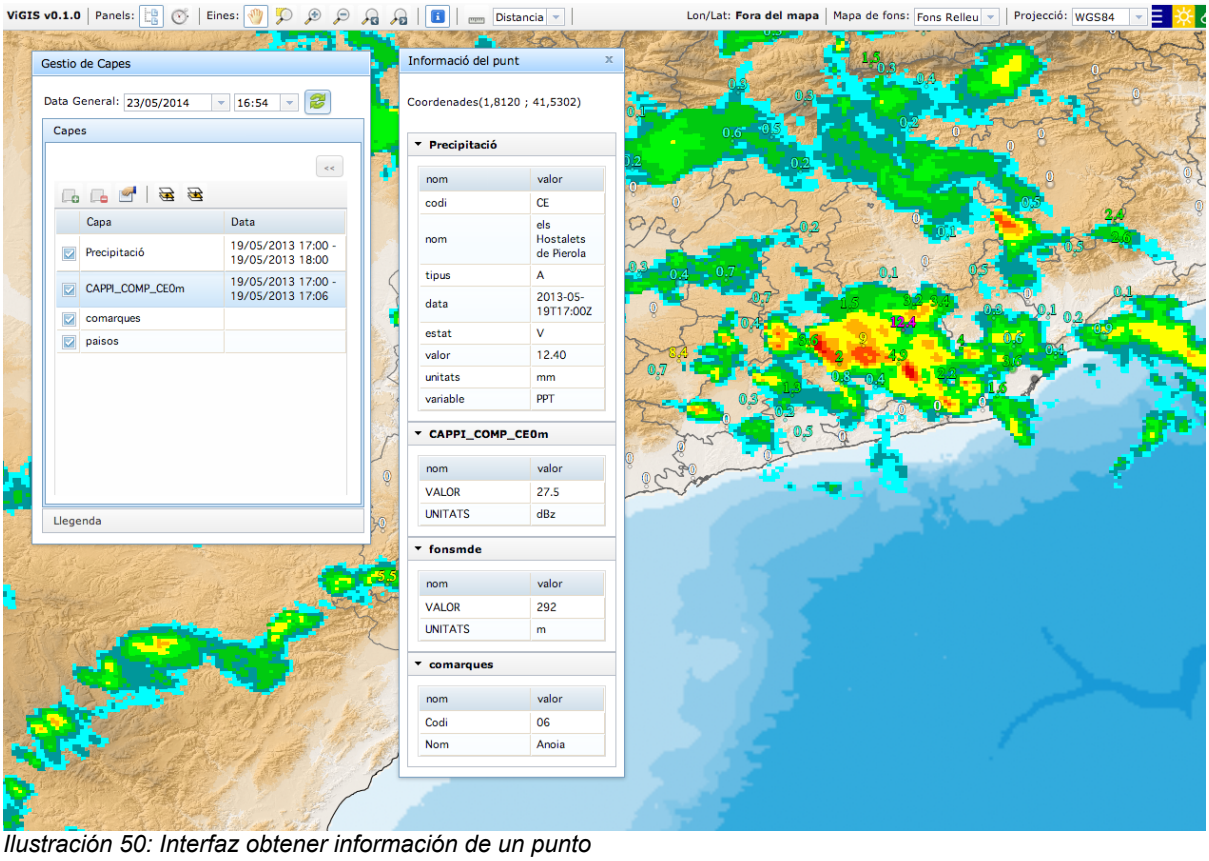


Ilustración 50: Interfaz obtener información de un punto

10.7.4 Gestión de capas

La función de este panel es la de gestionar las capas que se visualizan. Tiene dos paneles desplegables en forma de acordeón, “Capes” y “Llegenda”(Leyenda). El primero realiza la gestión de las capas y el segundo muestra las leyendas de las capas cargadas.A parte de permitir añadir o eliminar capas, desde este panel se puede gestionar el orden de visualización de las capas. Las capas se muestran en una lista, el orden de visualización en el visor viene determinado por el orden de las capas en esta lista. La capa que se encuentra más arriba en la lista se visualiza primero y así sucesivamente. Un ejemplo de esta funcionalidad se puede ver en las ilustraciones 51 y 52. En la ilustración 51 podemos ver dos capas cargadas del modelo WRF, una de temperatura y otra de viento. Como se observa en la lista de capas, la capa de viento está por debajo de la de temperatura, lo cual hace que la capa de viento no se vea en el mapa puesto que la capa temperatura la tapa. En la ilustración 52 podemos ver cómo se ha subido una posición la capa de viento y se puede ver en el mapa.

Desde la lista de capas también se puede activar o desactivar la visualización de las capas mediante el checkbox que hay en la izquierda de la lista.

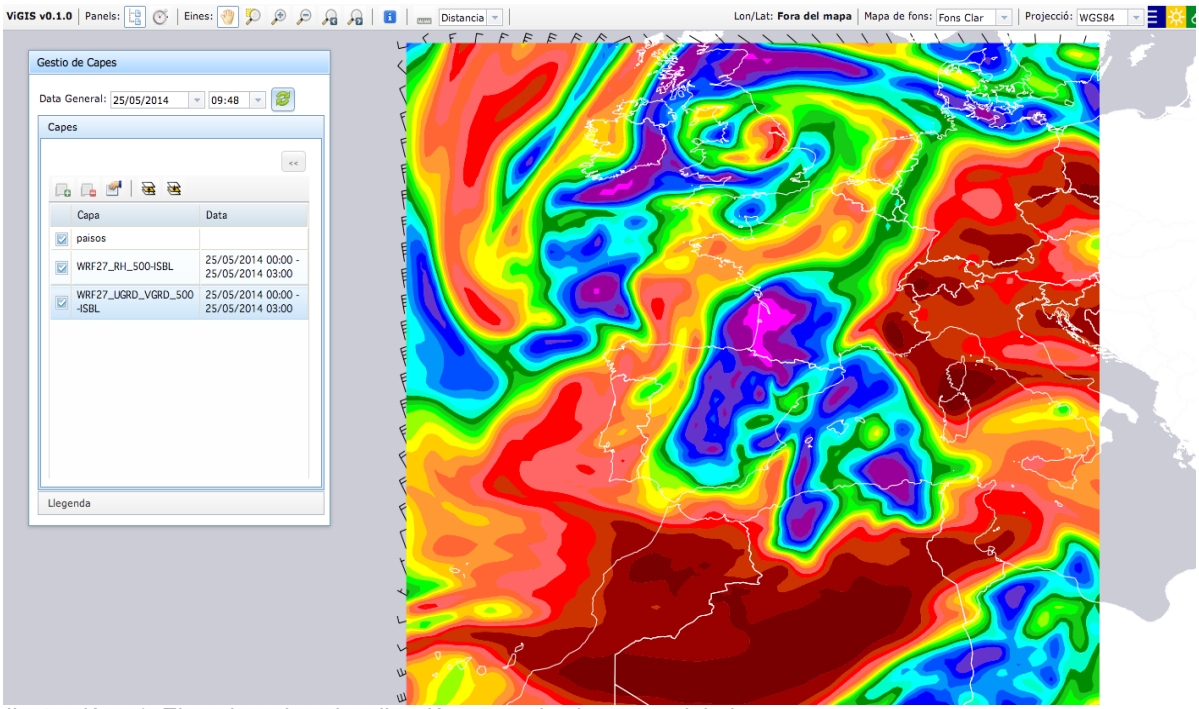


Ilustración 51: Ejemplo orden visualización, capa de viento por debajo

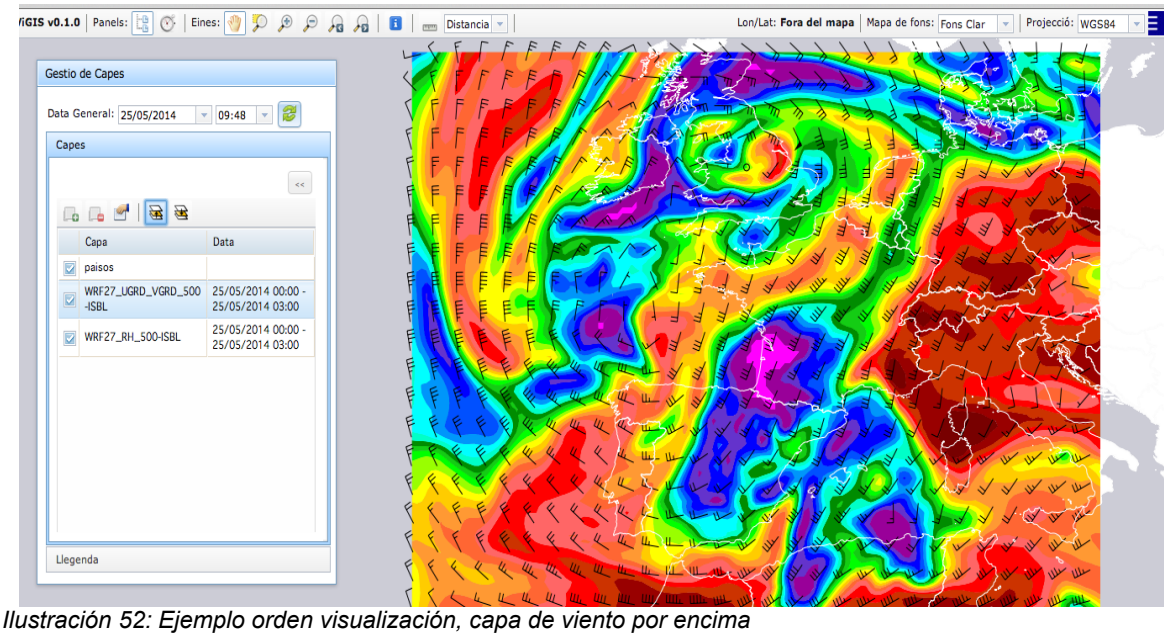


Ilustración 52: Ejemplo orden visualización, capa de viento por encima

10.7.5 Propiedades de visualización

Desde el panel de gestión de capas, también se puede acceder a las propiedades de las capas. Haciendo doble click en la capa o mediante el botón de acceso a propiedades. Éste panel muestra dos pestañas. La primera muestra las propiedades de visualización de la capa, ilustración 53. Donde se pueden modificar ciertos parámetros: la fecha de consulta, la opacidad de la capa, ofrece un botón para visualizar la capa en su extensión máxima(la que tenga configurada la capa en el servidor) y la leyenda de visualización. En las capas de la XEMA también se pueden agrupar o desagrupar las capas. La segunda pestaña muestra información de la capa, tal y como se muestra en la ilustración 54.

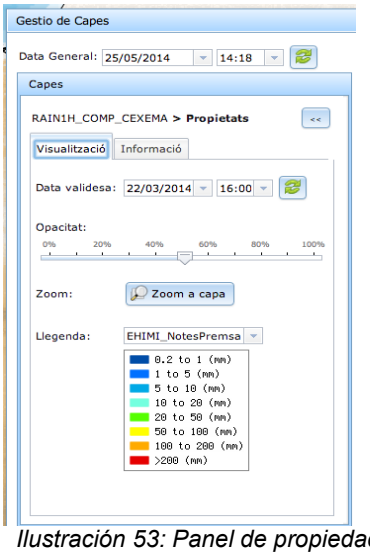


Ilustración 53: Panel de propiedades de la capa

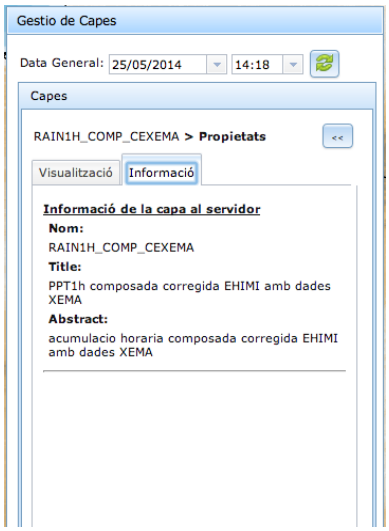


Ilustración 54: Panel de información de la capa

10.7.6 Leyenda

El visor ofrece la posibilidad de visualizar las leyendas de las capas cargadas. Desde el panel de gestión de capas se puede desplegar el panel (en forma de acordeón) que muestra las leyendas de las capas que se están visualizando. En la ilustración 55 podemos ver el panel leyenda, que contiene las leyendas de las capas CAPPI_0m y rayos.

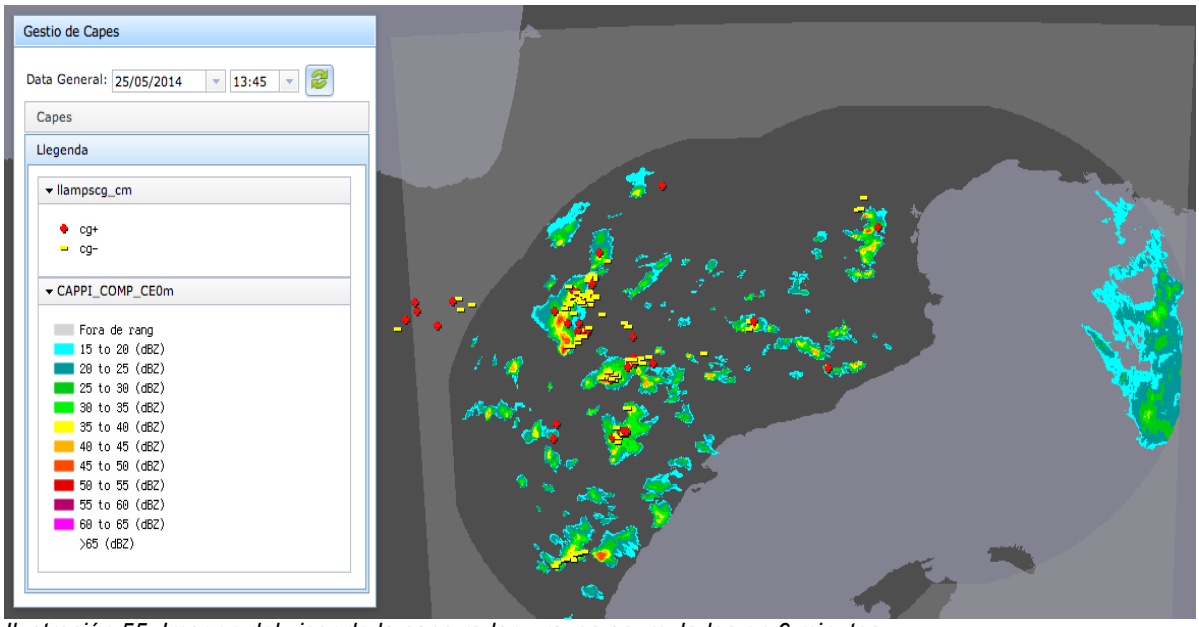


Ilustración 55: Imagen del visor de la capa radar y rayos acumulados en 6 minutos

10.7.7 Animación

El panel de animación se activa mediante el botón  ubicado en el menú superior. La animación ofrece la posibilidad a los usuarios de poder animar las capas cargadas en una linea temporal. Cuando se activa el panel de animación, todas las capas de la lista se actualizan a la fecha que indica la animación. En el panel de animación se muestran una serie de campos y parámetros que se utilizan para poder determinar el intervalo de tiempo a visualizar, la velocidad de la animación, el intervalo de salto entre animación y animación, y ofrece la posibilidad de realizar la animación en forma de bucle. También se ofrece un cuadro de mando para realizar la reproducción y una linea temporal. En la linea temporal se muestra la fecha que se esté visualizando.

A continuación se muestra una secuencia de 4 imágenes donde se puede observar una animación de la capa de radar junto con la de rayos.

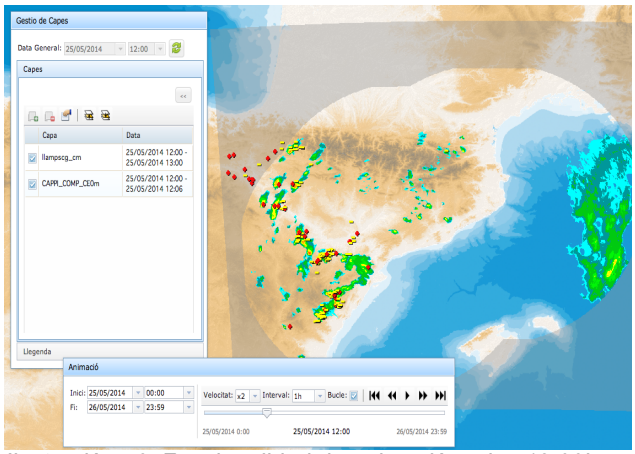


Ilustración 56: Funcionalidad de animación a las 12:00h

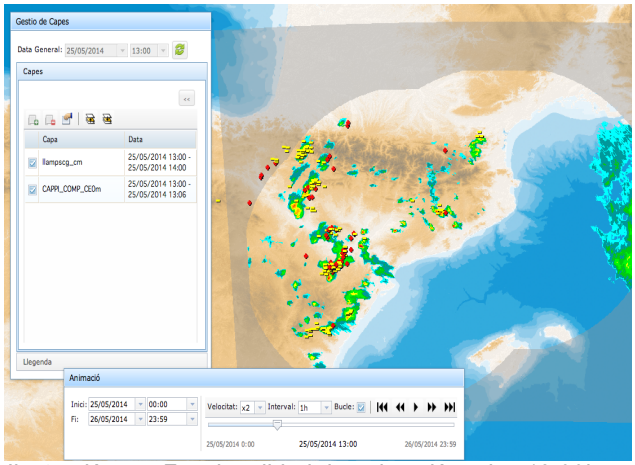


Ilustración 57: Funcionalidad de animación a las 13:00h

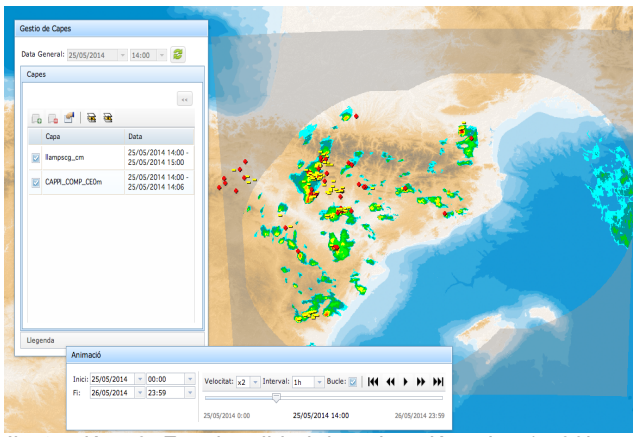


Ilustración 59: Funcionalidad de animación a las 14:00h

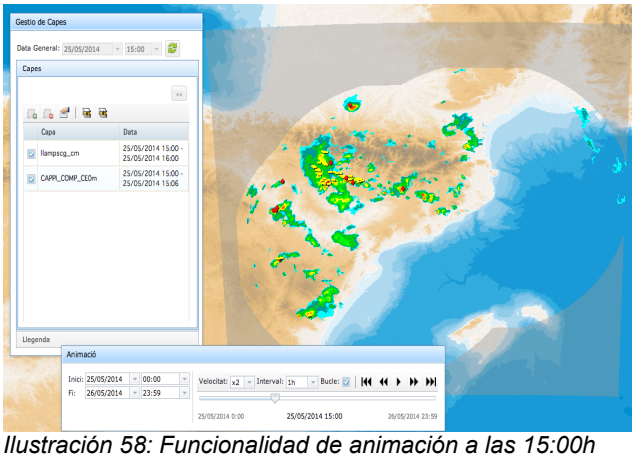


Ilustración 58: Funcionalidad de animación a las 15:00h

11 Futuras versiones

En esta primera versión se han establecido las bases del sistema SIG del SMC. Implementando el servidor de mapas con las capas más importantes y el visor con las funcionalidades básicas.

En futuras versiones la prioridad principal es añadir más capas tanto en la parte del servidor de mapas, como en el visor. Puesto que el visor se ha implementado de la forma más genérica posible, añadir más capas es relativamente fácil.

En la primera versión del sistema se ha añadido una proyección de visualización. En las siguientes versiones se añadirán más proyecciones. La primera proyección que se añadirá sera ETRS89. ETRS89 es la proyección que se ha establecido por la comunidad Europea mediante el programa INSPIRE. Añadir nuevas proyecciones en la capa de servidor es relativamente fácil, únicamente hay que añadir la proyección a los MapFiles y al fichero *mapcache.xml.* En el visor es algo más costoso, ya que hay que implementar la posibilidad de modificar la proyección de manera dinámica.

Realizar filtros de la información de las capas vectoriales, es otra de las funcionalidades a añadir. Ésto permitiría poder filtrar por ejemplo las estaciones de la XEMA mediante algún campo como por ejemplo la altura. O en la capa de municipios, poder buscar un municipio en concreto y centralizar la vista en ese municipio.

La posibilidad de realizar gráficas, meteorográmas y cortes verticales, es otra de las funcionalidades futuras a implementar en el visor. Ésta funcionalidad supone una implementación bastante importante. Ya que se requiere añadir nuevos módulos y definir una nueva interfaz para mostrar las gráficas.

Otra funcionalidad que se plantea realizar en el futuro es la ofrecer la posibilidad de exportación en forma de imagen la información que se esta visualizando. Con la idea de poder difundir en las redes sociales del SMC, utilizarla en los vídeos de predicción que realiza el SMC y también añadirlas en estudios meteorológicos.

Una funcionalidad interesante podría ser permitir cargar capas automáticamente a través de perfiles. Lo cual evitaría tener que cargar capas manualmente. Ésto es realmente útil cuando se quiere consultar datos que están relacionados, como por ejemplo la capa de radar con la capa de rayos.

12 Conclusiones

El desarrollo del proyecto se ha realizado dentro de los plazos establecidos en la planificación, respetando los hitos y entregas.

El sistema se ha desarrollado e implantado correctamente en el SMC. Se han implementado todos los requisitos descritos. Al finalizar el proyecto se ha obtenido un sistema de visualización SIG para los datos meteorológicos del SMC. Implementado y configurando un servidor de mapas, formado por MapServer y MapCache, y desarrollado e implementado una aplicación web de visualización SIG. Que permite a los técnicos del SMC consultar los productos que el propio SMC genera. Previo a la implantación se ha establecido un sistema de almacenamiento de los productos, para que su explotación sea lo más ágil y comprensible posible. También se han alcanzado todos y cada uno de los objetivos que se han descrito durante el proyecto.

En posteriores versiones se ampliarán funcionalidades, información a visualizar y se cubrirán las necesidades de los técnicos del SMC. La evolución del sistema se realizará mediante versiones que agrupen ciertas funcionalidades y/o mejoras.

A nivel personal el desarrollo del proyecto me ha permitido consolidar mis conocimientos en el ámbito SIG. También me ha permitido utilizar tecnologías y herramientas de web mapping. Durante el desarrollo del proyecto he podido conocer productos meteorológicos y conocer porque se generan y para que se aplican.

13 Glosario

API REST f arquitectura de desarrollo web que ofrece información mediante una serie de recursos.

Feature f en el ámbito SIG, hace referencia a una entidad geográfica vectorial. Ésta puede ser tanto un punto, un polígono, un multipolígono o una línea.

GeoTIFF m Es un estándar de metadatos que provee de información geográfica a una imagen en formato TIFF. Esta información geográfica incluye proyección de la imagen, sistema de coordenadas, elipsoide, datum y otro tipo de información geográfica útil para georeferenciar una imagen.

Grib m formato que se usa internacionalmente para transportar y manipular datos meteorológicos, sobre todo para el ámbito de la predicción. Almacena la información en una serie bandas a modo de variables.

JSON m formato ligero de intercambio de datos. Alternativa al uso de ficheros XML más pesados. Fácil de interpretar tanto para humanos como para procesos. Es un subconjunto de la notación literal de objetos de JavaScript

Leyenda f descripción de la información representada en una imagen o en objetos vectoriales. Se puede utilizar tanto símbolos como colores para la representación de la información.

MapFile m fichero de configuración de MapServer que permite configurar capas para diferentes servicios.

Metadatos m Definidos como “datos de los datos”. Conjunto estructurado de información que describe un conjunto de datos. Dentro de los SIG los metadatos aportan información como la proyección, extensión, formato, etc.

OGC m Open Geospatial Consortium, consorcio internacional de 479(en el año 2014) organizaciones, empresas e universidades, creado en 1994. Que tiene como objetivo la creación, definición y publicación de estándares abiertos dentro de los SIG.

PostGIS m modulo de PostgreSQL que ofrece funcionalidades SGI, para que la base de datos pueda almacenar y utilizarse como una base de datos espacial.

PostgreSQL f sistema gestor de base de datos relacional OpenSource. En la actualidad es el sistema gestor de base de datos de código abierto más potente en al actualidad.

Raster m fichero de datos representados mediante una matriz de pixels, donde cada pixel tiene como valor un color.

Shapefile m formato creado por ESRI, que se ha convertido en un estándar en sistema SIG. Permite almacenar información vectorial georeferenciada.

Tile f Al castellano se traduce como tesela. Generada a partir de la división en una cuadricula de una imagen mayor.

Vectorial m Representa un conjunto de información mediante lineas, polígonos, puntos o multipolígonos.

WGS84 m Sistema de coordenadas estándar mundial creado en 1984.

WMS m servicio estándar desarrollado por el OGC, que ofrece información dinámica de datos georeferenciados, a través de la web.

WMTS m servicio desarrollado por OGC en 2010. Permite servir información espacial de imágenes mediante teselas de datos a través de internet.

14 Bibliografía

Olaya, Victor (2012). “Sistemas de información Geográfica”. OSGeo.
http://wiki.osgeo.org/wiki/Libro_SIG

Muñoz Bollas, Anna. “Geodèsia i cartografia”. UOC

Wikipedia

http://es.wikipedia.org/wiki/Sistema_de_informaci%C3%B3n_geogr%C3%A1fica

http://es.wikipedia.org/wiki/Web_Map_Service

OGC

http://www.opengeospatial.org/

OSGeo

http://www.osgeo.org/

MapServer

http://mapserver.org/

Servei Meteorològic de Catalunya

http://www.meteo.cat/

Instituto Meteorológico Noruego

http://www.yr.no/

Institut Cartogràfic i Geològic de Catalunya

http://www.icgc.cat/

Dojo Toolkit

http://dojotoolkit.org/

OpenLayers

http://openlayers.org/

Otros

http://robinlovelace.net/software/2014/03/05/webmap-test.html