UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Departamento de Electrónica e Computación

Tesis doctoral

DESARROLLO, DESPLIEGUE Y VALIDACIÓN DE UN LABORATORIO VIRTUAL OCEANOGRÁFICO BASADO EN COMPUTACIÓN GRID

Presentada por:

D. David Mera Pérez

Dirigida por:

Dr. D. José Manuel Cotos Yáñez

Dr. D. José Varela Pet

Noviembre de 2012

Dr. D. José Manuel Cotos Yáñez, Profesor Titular de Universidad del Área de Lenguajes y Sistemas Informáticos de la Universidad de Santiago de Compostela

Dr. D. José Varela Pet, Profesor Contratado Doctor de Universidad del Área de Lenguajes y Sistemas Informáticos de la Universidad de Santiago de Compostela

HACEN CONSTAR:

Que la memoria titulada **DESARROLLO, DESPLIEGUE Y VALIDACIÓN DE UN LABORATORIO VIRTUAL OCEANOGRÁFICO BASADO EN COMPUTACIÓN GRID** ha sido realizada por **D. David Mera Pérez** bajo nuestra dirección en el marco del Programa de Doctorado Interuniversitario de Investigación en Tecnologías de la Información del Departamento de Electrónica e Computación de la Universidade de Santiago de Compostela, y constituye la Tesis que presenta para optar al título de Doctor.

Este trabajo ha sido desarrollado en los siguientes centros de investigación de la Universidad de Santiago de Compostela:

- Centro Singular de Investigación en Tecnoloxías da Información (CITIUS).
- Instituto de Investigacións Tecnolóxicas (IIT).

Noviembre de 2012

Dr. D. José Manuel Cotos Yáñez
Director de la tesis

Dr. D. José Varela Pet
Codirector de la tesis

D. David Mera Pérez
Autor de la tesis

The Answer to the Great Question, of Life, the Universe and Everything [...] Forty-two, said Deep Thought, with infinite majesty and calm. [...] That quite definitely is the answer. I think the problem, to be quite honest with you, is that you've never actually known what the question is.

The Hitchhiker's Guide to the Galaxy.

Agradecimentos

Parece que fue ayer cuando me embarqué en este viaje pero la realidad es que ya ha pasado bastante tiempo y, sobre todo, muchas horas de trabajo. Es el momento de echar la vista atrás y pensar en todo el apoyo recibido de mis compañeros de trabajo, amigos, familia y organizaciones sin el cual esto nunca hubiese llegado a buen puerto.

En primer lugar, agradecer al Dr. José Manuel Cotos Yáñez y al Dr. José Varela Pet, directores del presente trabajo, la oportunidad de emprender esta aventura junto a ellos y por la confianza depositada en mí desde el comienzo. Gracias por apoyarme, guiarme y dedicarme tantas horas de vuestro tiempo para sacar esto adelante.

A mis compañeros de trabajo del Instituto de Investigaciones Tecnológicas, con los que he pasado la mayor parte de este tiempo, que me han aconsejado y ayudado siempre que lo he necesitado.

A Carmen Cotelo Queijo, mi compañera de armas en el proyecto de RETELAB, por echarme una mano desinteresada siempre que me ha hecho falta.

Al Dr. Pablo García Rodríguez por sus conocimientos y consejos pero especialmente por su apoyo y ánimos durante todo el trabajo.

Al Dr. Óscar García Pineda de la Universidad Estatal de Florida por su tiempo y conocimientos e indicarme el camino cuando éste estaba poco claro.

A mis amigos por aguantar mis «rollos frikis», por darme siempre ánimos y, sobre todo, por estar siempre ahí aunque los tenga abandonados.

A Leticia, «mi vida», por todas las horas que le he robado y que no podré devolverle. No sabría como expresar lo importante que has sido. Gracias por estar siempre a mi lado y por tu apoyo incondicional. Sin ti, esto no sería posible.

A mi familia y, en especial a mis padres, por todo el cariño y apoyo recibido durante toda mi vida y por confiar en que yo era capaz de cualquier cosa y animarme siempre a luchar y a esforzarme al máximo en todos los aspectos de la vida. Sin vosotros yo no sería quien soy. Gracias por todo.

Y a tantos otros que no he nombrado pero que llevo en la cabeza y en el corazón.

¡GRACIAS A TODOS!

Este trabajo no habría sido posible sin el apoyo y/o colaboración de las siguientes instituciones:

– Sociedad de Salvamento y Seguridad Marítima (SASEMAR), concretamente el Centro de Coordinación de Salvamento de Finisterre que nos facilitó los datos sobre los vertidos detectados en la costa gallega durante los últimos años.

– Ministerio de Educación y Ciencia a través de la financiación del proyecto «Laboratorio Virtual para la Red Nacional de Teledetección Oceanográfica» con referencia ESP2006-13778-C04.

– Xunta de Galicia a través de la financiación del proyecto «Análise e interpretación de imaxes de satélite a partires de técnicas de intelixencia artificial: aplicación a contaminantes no medio mariño» con referencia PGIDIT08SIN001236PR.

– Universidad de Santiago de Compostela mediante el «Programa de Contratos Predoctorales de la Universidad de Santiago de Compostela 2010».

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Las necesidades computacionales de la sociedad actual son cada vez mayores. Determinadas áreas de investigación como la física de altas energías, las ciencias de la Tierra o la meteorología, ven limitado su desarrollo por la necesidad de más y mejores recursos computacionales. Aunque el desarrollo tecnológico es la primera barrera que se encuentran los investigadores, también lo es el acceso a la propia tecnología, ya que, a menudo, la tecnología más puntera es económicamente inaccesible o su uso requiere de extensos conocimientos que limitan su generalización.

La computación distribuida surge con el objetivo de permitir la colaboración y compartición de recursos para reducir costes y maximizar la capacidad de almacenamiento y procesamiento. Su rápida evolución permitió a los investigadores resolver problemas complejos que hasta hace poco eran inasumibles y afrontar retos más ambiciosos incluso cuando los medios de los que disponían eran escasos.

La observación remota de la Tierra es un claro ejemplo en el que las necesidades computacionales avanzan a un ritmo difícil de igualar por el desarrollo tecnológico. Desde que en 1957 el Sputnik Soviético fuese el primer satélite en ser enviado al espacio, han sido muchas las misiones tanto civiles como militares que se han llevado a cabo para observar remotamente la Tierra. Año tras año el número de misiones espaciales, así como el número de proyectos de investigación relacionados con éstas, se ha ido incrementando y, al menos en un futuro a corto y medio plazo, la política a este respecto será continuista, ya que según un informe de Euroconsult, *Satellite-Based Earth Observation, Market Prospects to 2017*, se espera que cerca de 200 nuevos satélites estén operativos en el 2017.

Gracias a los avances tecnológicos, las misiones se han vuelto cada vez más sofisticadas y cuentan con instrumentación más especializada, generando un gran volumen de información sobre nuestro planeta que es necesario almacenar y procesar. La observación remota de la Tierra o teledetección se ha convertido en una técnica imprescindible en el seguimiento y estudio de determinados procesos ambientales críticos para nuestro planeta como, por ejemplo, el cambio climático, el deshielo o el deterioro de la capa de ozono. Del mismo modo, la teledetección también es esencial en el seguimiento de desastres, tanto naturales como provocados por el hombre, como son los incendios, los terremotos o los vertidos contaminantes. En particular, la teledetección es una herramienta fundamental en el estudio de los océanos y sus fenómenos asociados.

Los océanos cubren cerca de las tres cuartas partes del planeta y albergan al 97 % de las especies pero, a pesar de estas cifras, a menudo no somos conscientes de su importancia e influencia; por ejemplo: son el pulmón del planeta, ya que más de la mitad de la captura mundial de CO_2 y de la producción de oxígeno se realizan a través del mecanismo conocido como Producción Primaria (PP) [56], las actividades relacionadas con el mar sustentan al 10-12 % de la población mundial [42] y nuestra dependencia energética hacia ellos es incuestionable, puesto que aproximadamente la mitad de la producción mundial de gas natural y el 30 % de la de petróleo se extrae de los yacimientos presentes en el lecho marino [11].

El estudio de los océanos es una tarea interdisciplinar en el que grupos tan heterogéneos como los de los biólogos, físicos, meteorólogos, oceanógrafos o informáticos deben trabajar juntos para abordar problemas de naturaleza compleja y obtener los mejores resultados. A menudo, dichos grupos se encuentran dispersos y pertenecen a centros diferentes, lo que supone un problema, tanto organizativo como de seguridad y, en muchos casos, este tipo de dificultades puede obstaculizar la buena marcha de los proyectos o directamente imposibilitarlos. Además, la gran cantidad de datos, obtenidos a través de la observación remota y necesarios para la investigación, genera grandes dificultades en cuanto a su almacenamiento, procesamiento y distribución. Los centros necesitan disponer tanto de recursos de computación de altas prestaciones como de almacenamiento masivo, acordes a las necesidades de los proyectos pero esto no siempre es posible.

La computación Grid, paradigma de la computación distribuida, se posiciona como la tecnología más adecuada para cubrir las demandas tecnológicas de la investigación oceanográfica y proporcionar un acceso ágil y eficiente a los datos almacenados. La computación Grid permite desplegar entornos dedicados a compartir recursos entre diferentes centros a través de la

red y, además, permite administrar grupos virtuales de usuarios que se forman, en el marco de un proyecto, para maximizar sus posibilidades en torno a objetivos comunes.

1.1. Computación Grid

Tradicionalmente, los grupos y las organizaciones para la investigación han trabajado de forma local con sus propios recursos (computadores, sensores, bases de datos, etc.) y, a menudo, la heterogeneidad de los recursos, así como la ausencia en el uso de estándares, ha dificultado tanto la colaboración con otros centros como la compartición de la información. Este modelo de trabajo queda obsoleto desde el momento en que se afrontan problemas interdisciplinares, interorganizativos y que demandan cantidades ingentes de recursos que ningún centro puede asumir por sí solo. Un ejemplo de ello es el proyecto *Large Hadron Collider* (LHC) que genera alrededor de 15 Petabytes de datos cada año [99], los cuales deben ser almacenados, procesados y distribuidos. Proyectos de tal envergadura, que normalmente engloban a multitud de grupos, hacen necesario modificar la forma de trabajar y buscar alternativas que permitan compartir recursos para adaptarse a las necesidades de los proyectos.

La colaboración entre grupos interdisciplinares y dispersos geográficamente es una pieza clave en este nuevo modelo de trabajo, ya que deja obsoleto el concepto tradicional de organización y da paso a las llamadas Organizaciones Virtuales (VOs), donde diferentes grupos se unen durante un período de tiempo determinado y comparten sus recursos para acometer unas actividades concretas. La compartición de recursos en VOs debe entenderse en el sentido más general, es decir, acceso remoto a software, Bases de Datos (DBs), sensores, computadoras y cualquier otro tipo de medio tanto físico como lógico, por lo que se hacen indispensables métodos adaptados para su gestión.

Durante la década de los 90 aparece una nueva aproximación denominada computación Grid que nace con los objetivos de solucionar las demandas computacionales de la sociedad y de facilitar la adaptación a los nuevos conceptos organizativos. La computación Grid supone un nuevo paradigma de computación distribuida que permite agregar y compartir recursos heterogéneos y geográficamente distribuidos entre diferentes organizaciones.

En uno de los primeros esfuerzos por concretar el término Grid, Ian Foster y Carl Kesselman [46] lo definen como:

«Una infraestructura hardware y software que proporciona un acceso fiable, consistente, generalizado y de bajo coste a capacidades computacionales de alta calidad.»

Profundizando en la descripción tenemos que:

– Por infraestructura, Foster y Kesselman se refieren a un conjunto de recursos heterogéneos que incluye datos, sensores, ciclos de computación, etc., así como al hardware necesario para la interconexión de los recursos y el software utilizado para su monitorización y gestión.

– El acceso es considerado fiable si proporciona a los usuarios unos niveles de rendimiento altos, predecibles y estables.

– Por acceso consistente se entiende que se proporcionan servicios a través de interfaces estándar y utilizando parámetros normalizados. El uso de estándares beneficia a la generalización del paradigma y al desarrollo de aplicaciones de usuario.

– Se considerará que el acceso es generalizado si el usuario cuenta con un núcleo de servicios genéricos sea cual sea el sistema Grid al que se conecta.

– El acceso a bajo coste es determinante para su aceptación, ya que su uso deberá ser más rentable que el de sus alternativas.

A medida que la tecnología Grid fue evolucionando, la definición anterior fue manifestándose insuficiente e incluso se dudó de la necesidad de este nuevo paradigma; por ello, Foster et ál. [52] redefinen el concepto y lo asocian a un problema sin resolver:

«El problema real que subyace tras el concepto Grid es la coordinación de organizaciones virtuales dinámicas y multiinstitucionales en la resolución de problemas y la compartición de recursos.»

En la definición anterior, se entiende por una VO un conjunto de personas y/o instituciones que se agrupan a través de una política de compartición de recursos. Las VOs pueden variar en gran medida en cuanto a su propósito, ámbito, tamaño, duración y localización.

Los autores justifican la necesidad de este nuevo paradigma por no existir, hasta ese momento, una tecnología de computación distribuida que permitiese gestionar recursos compartidos basándose en relaciones tan flexibles como las de las VOs, con sofisticados controles de acceso que incluyesen una gestión desde el punto de vista más específico al más general, con delegación de credenciales o que integrase el uso de políticas tanto locales como globales y que, además, ofreciera un alto rendimiento, calidad de servicio, planificación de las tareas, múltiple asignación de recursos, monitorización y bajo coste.

La Figura 1.1 muestra un ejemplo de un sistema Grid y del uso de los recursos compartidos por parte de dos VOs. En este ejemplo, tres centros ponen a disposición de la comunidad parte de sus recursos pero gestionados por una política de acceso que filtra por la VO a la que pertenecen los usuarios. Cada centro está etiquetado con los recursos que comparte y la política de seguridad que se les aplica.

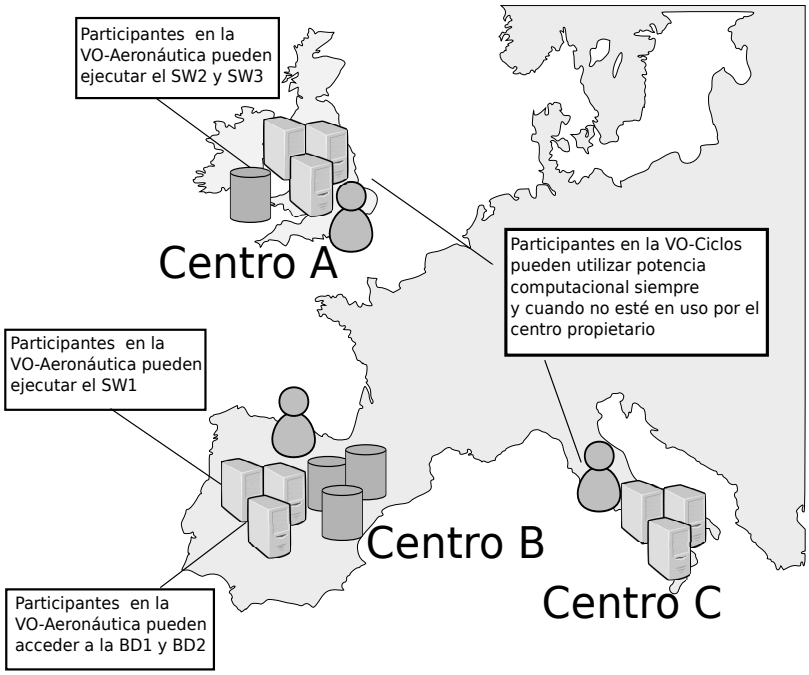

Figura 1.1: Ejemplo básico de un sistema Grid formado por 3 centros que comparten parte de sus recursos con dos VOs (VO-Aeronáutica y VO-Ciclos). Los participantes en la VO-Ciclos aprovechan la potencia computacional (ciclos) de los recursos compartidos por los centros 'A' y 'C'. Los participantes de la VO-Aeronáutica pueden ejecutar aplicaciones que han sido compartidas por los centros 'A' y 'B' y acceder a las DBs del centro 'B'. Elaboración propia a partir de [52].

A medida que las tecnologías Grid se fueron popularizando, el término Grid se empezó a explotar comercialmente diluyéndose su significado y utilizándose como eslogan para captar nuevos clientes. Las empresas incluyeron el término en muchos productos que difícilmente podían encajar dentro de su definición. En un intento de delimitar el concepto, Ian Foster [49] publicó una lista con 3 puntos básicos que recogen la esencia del concepto y que, en una publicación posterior junto con Carl Kesselman [8], agrupó para proponer una nueva definición:

«Un Grid es un sistema que coordina recursos distribuidos a través de protocolos e interfaces abiertas, estándar y de propósito general con el objetivo de proporcionar una calidad de servicio no trivial.»

Los elementos clave de la definición anterior y que formaban la lista de tres puntos básicos se detallan a continuación:

- Integración y gestión de recursos heterogéneos y distribuidos.
- Uso de protocolos e interfaces estándar, abiertos y de propósito general. Un sistema Grid gestiona un entorno formado por recursos heterogéneos, por lo que la implementación de protocolos estándar y libres permitirá que cualquier producto o aplicación, tanto *open-source* como comercial, pueda interoperar con el resto de componentes del sistema.
- Garantía de calidad de servicio en cuanto a tiempo de respuesta, disponibilidad, seguridad, múltiple asignación de recursos, etc.

1.1.1. Aplicaciones Grid

En general, la computación Grid se adapta a cualquier aplicación que por su complejidad no pueda ejecutarse de forma local y que potencialmente posea un alto grado de paralelización. Foster y Kesselman [46] identifican 5 tipos de aplicaciones que pueden obtener el máximo rendimiento de la computación Grid pero esperan la incorporación de otros tipos a medida que la computación Grid evolucione:

- Computación distribuida: aplicaciones con problemas de alta complejidad que utilizan la computación Grid para incorporar potentes recursos computacionales.
- Computación de Altas Prestaciones (HTC): se trata de aplicaciones que presentan problemas que requieren la ejecución de tareas secuenciales con baja interdependencia.
- Computación bajo demanda: se trata de aplicaciones que utilizarían las capacidades de la computación Grid para satisfacer necesidades puntuales de recursos. Son aplicaciones dirigidas por una política de costes y no de rendimiento.
- Computación intensiva de datos: este tipo de aplicaciones se centran en sintetizar nueva información a través del análisis de grandes cantidades de datos. Los datos son divididos

en segmentos que pueden ser procesados de forma independiente por los diferentes recursos del sistema. Posteriormente, se deberán unificar las salidas parciales en un resultado final.

– Computación colaborativa: aplicaciones centradas en permitir y mejorar las interacciones entre los usuarios, a menudo utilizando un espacio virtual común y compartiendo recursos computacionales.

La computación Grid aporta una serie de beneficios claros para los centros de investigación que también pueden trasladarse a la empresa privada [82] como por ejemplo:

– Permite compartir recursos heterogéneos a través de VOs (DBs, ciclos de computación, software, etc.).

– Provee un acceso transparente a recursos remotos.

– Reduce el número de servidores necesarios en las organizaciones.

– Permite agregar recursos bajo demanda. Ante una eventual emergencia, su flexibilidad permitiría alquilar y agregar recursos externos por un tiempo limitado.

– Optimiza el uso de recursos organizativos compartiéndolos cuando estén infrautilizados.

– Reduce el tiempo de ejecución de aplicaciones complejas.

– Provee tolerancia a fallos.

– Permite la virtualización de centros de datos.

– Permite balancear la carga entre los diferentes recursos.

1.2. Motivación

Como se ha visto en los apartados anteriores, el estudio de los océanos es de vital importancia y aglutina disciplinas con poca o ninguna relación. En el pasado, sólo los grandes centros de investigación podían contar con equipos multidisciplinares y recursos locales suficientes para afrontar cierto tipo de proyectos. La computación Grid facilitó la colaboración y la compartición de recursos de altas prestaciones entre centros más modestos.

Desde el punto de vista de las tecnologías, a menudo los usuarios finales son reacios a utilizar herramientas o sistemas software que, a pesar de ser adecuados para su trabajo, presenten una interfaz de usuario complicada o simplemente diferente a lo que están acostumbrados.

Los *middleware* Grid como, por ejemplo, Globus permiten desplegar sistema distribuidos Grid con relativa rapidez y facilidad pero muy limitados en cuanto a usabilidad y basados en una Interfaz de Línea de Comandos (CLI) poco intuitiva y amigable, lo que puede llevar a reducir su grado de aceptación .

Sería deseable disponer de un sistema Grid enfocado a la investigación oceanográfica que permita compartir recursos y posibilite la colaboración entre diferentes grupos, pero desarrollado desde la sencillez y la usabilidad, ya que, debido a la heterogeneidad de los grupos, los conocimientos tecnológicos no serán los mismos entre todos los integrantes.

1.3. Objetivos

El presente trabajo nace con el propósito de cubrir las necesidades de la Red Española de Teledetección Marina (RETEMAR) en materia de computación de altas prestaciones. Esta Red auna investigadores y tecnólogos de diferentes centros con el objetivo de maximizar recursos y conocimientos y dirigirlos hacia la consecución de objetivos comunes en el marco del desarrollo de proyectos de investigación. El desarrollo de un laboratorio virtual, así como de aplicaciones y herramientas enfocadas a la investigación oceanográfica, es clave en la búsqueda de la cooperación entre los diferentes socios de RETEMAR.

La Red permite afrontar proyectos de mayor envergadura de la que podría abarcarse de manera independiente. De entre los objetivos comunes, cabe destacar el interés mutuo en la lucha contra la contaminación marina que, desgraciadamente, afecta diariamente a las costas españolas.

Este trabajo presenta dos objetivos principales:

– El desarrollo y despliegue de un laboratorio virtual a través de un entorno de computación distribuida que permita a los investigadores colaborar y realizar proyectos de forma conjunta.

– Desarrollar e integrar en el laboratorio un sistema de detección de vertidos de hidrocarburos en el mar, a partir del análisis de imágenes de satélite.

El laboratorio deberá permitir a los investigadores acceder a recursos y datos distribuidos, así como a la potencia computacional y espacio de almacenamiento necesarios para desarrollar sus trabajos. El acceso deberá ser amigable y concebido para que los científicos puedan centrar sus esfuerzos en la investigación y no en el aprendizaje y/o utilización de las herramientas.

Dado que los recursos serán interorganizativos, la seguridad deberá ser una de las bases de la plataforma pero sin ser un impedimento para el trabajo, por lo que las políticas de seguridad deberían ser transparentes para los usuarios finales.

Para validar el uso de la plataforma, se desarrollará un sistema de detección de vertidos de hidrocarburos que permita monitorizar la costa a través de imágenes de satélite y, de forma semiautomática, identificar y georreferenciar los vertidos.

1.3.1. Objetivos concretos

Se propone el desarrollo de un entorno colaborativo y distribuido que constituya un laboratorio virtual para el desarrollo de proyectos de investigación interdisciplinares relacionados con la teledetección oceanográfica. Para ello se deberán alcanzar los siguientes subobjetivos:

1. Despliegue de un sistema de computación Grid que permita compartir recursos heterogéneos y distribuidos.
2. Desarrollo de una interfaz de usuario amigable como medio de acceso al Grid que permita a los usuarios finales centrarse en sus trabajos y no en las herramientas.
3. Implementación de un sistema de acceso y registro de usuarios basado en *Public Key Infrastructure* (PKI) y Control de Acceso Basado en Roles (RBAC).
4. Despliegue de un sistema de almacenamiento distribuido gestionado por metadatos.
5. Implementación de un sistema de envío y ejecución de trabajos optimizado para seleccionar, de forma dinámica y transparente al usuario, el mejor recurso Grid para ejecutar la tarea.
6. Desarrollo de un sistema de monitorización que permita visualizar el estado de cada tarea enviada al Grid.
7. Integración de herramientas de visualización que permitan al usuario acceder a las colecciones de datos almacenadas en el laboratorio.

8. Despliegue e integración de herramientas y aplicaciones del ámbito oceanográfico para la validación del laboratorio.

Con el doble propósito de validar el laboratorio y de desarrollar un sistema de apoyo a la lucha contra la contaminación marina, se implementará una aplicación para la detección de vertidos de hidrocarburos en la superficie oceánica mediante el análisis de imágenes de satélite. Los subobjetivos planteados son los siguientes:

1. Desarrollo de un algoritmo de segmentación basado en una umbralización adaptativa que permita separar los posibles vertidos del resto de la imagen.
2. Integración de datos meteorológicos en el algoritmo de segmentación.
3. Análisis y desarrollo de un vector de características que permita representar a los candidatos segmentados.
4. Implementación de diferentes algoritmos de clasificación que, a partir de los candidatos segmentados, separen los falsos positivos de los vertidos.
5. Optimización del tiempo de ejecución de los algoritmos para ser utilizados en tiempo casi real.
6. Selección del clasificador que proporcione los mejores resultados e integración del sistema de detección en el laboratorio virtual.

1.4. Estructura de la memoria

Después del actual capítulo introductorio en el que se describen los orígenes, motivación y objetivos del trabajo, la presente memoria se divide en dos grandes bloques que responden a los objetivos principales de la tesis doctoral. Cada uno de estos objetivos se corresponde con un proyecto de investigación financiado por un organismo diferente pero ambos con una meta común.

La primera parte detalla el desarrollo de RETELAB, «Laboratorio Virtual para la Red Nacional de Teledetección Oceanográfica». RETELAB fue desarrollado gracias a la financiación del Ministerio de Educación y Ciencia (MEC) con la referencia ESP2006-13778-C04 entre los años 2007 y 2010. Siguiendo el espíritu del propio proyecto, el desarrollo de RETELAB aunó el trabajo de varias organizaciones distribuidas y heterogéneas:

– La Universidad de Santiago de Compostela (USC) y el Centro de Supercomputación de Galicia (CESGA) se centraron en la implementación del laboratorio. Por un lado, la USC aportó su experiencia en teletedetección, análisis de imágenes y desarrollo de portales y, por otro lado, el CESGA aportó su amplia su experiencia en el desarrollo de entornos de Computación de Alto Rendimiento (HPC) y Computación de Altas Prestaciones (HTC).

– Los usuarios típicos del laboratorio estaban representados por el Instituto Canario de Ciencias del Mar (ICCM) y el Centro Tecnológico del Mar y los Alimentos del País Vasco (AZTI). La experiencia de los investigadores de estos centros fue vital para realizar el análisis de requisitos, así como para desarrollar las aplicaciones del laboratorio.

El Capítulo 2 está dedicado a la descripción de diferentes aspectos de la computación Grid: la Sección 2.1 muestra las diferentes fases en la evolución de la tecnología, mientras que la Sección 2.2 describe la arquitectura típica de un sistema Grid, para finalmente ocuparse en la Sección 2.3 del desarrollo de portales Web como medio de acceso a los sistemas.

El Capítulo 3 se centra en la implementación del laboratorio virtual de RETELAB y profundiza en los diferentes módulos incluidos en su desarrollo:

– La Sección 3.2 describe la implementación del módulo de acceso y gestión de usuarios y cómo la integración de diferentes tecnologías permitió desplegar un sistema de acceso usable y seguro basado en un control de acceso RBAC.

– La Sección 3.3 profundiza en el despliegue y desarrollo del sistema de almacenamiento distribuido utilizado en RETELAB. La integración con la interfaz Web, así como el empleo de metainformación para describir las colecciones de datos, son dos de los puntos fundamentales.

– La integración de diferentes herramientas para la visualización de las colecciones de datos presentes en RETELAB es descrita en la Sección 3.4.

– La ejecución de tareas en RETELAB, así como su monitorización, se aborda en la Sección 3.5. A pesar de que no se profundiza en los detalles de la implementación, ya que esta tarea fue desarrollada principalmente por el CESGA, este apartado introduce cómo se ha integrado y mejorado el metaplanificador GridWay para su utilización desde el laboratorio. GridWay facilita a los usuarios el envío de tareas a un Grid seleccionando automáticamente el mejor recurso disponible para su ejecución.

– Por último, la Sección 3.6 muestra diferentes aplicaciones integradas para validar el laboratorio virtual e introduce el sistema desarrollado para detectar vertidos de hidrocarburos denominado SENTINAZOS y que centra la siguiente parte de la memoria.

La segunda parte de la tesis doctoral describe en detalle el proyecto SENTINAZOS, «Análise e interpretación de imaxes de satélite a partires de técnicas de intelixencia artificial: aplicación a contaminantes no medio mariño». El proyecto, con referencia PGIDIT08SIN-001236PR, fue financiado por la Consellería de Innovación e Industria de la Xunta de Galicia entre los años 2008 y 2010 e implicó, en su desarrollo, a la Universidad de la Coruña (UDC) y la USC; los equipos de ambas instituciones abordaron el problema desde dos puntos de vista diferentes pero complementarios:

– La UDC centró su esfuerzos en el desarrollo de algoritmos para la detección grandes cantidades de vertidos ocasionados por catástrofes.

– La USC se implicó en el desarrollo de un sistema de detección de pequeños vertidos (sentinazos) ocasionados, generalmente, por tareas de mantenimiento y en la integración de dicho sistema en RETELAB.

Después de introducir, en el Capítulo 4, el problema de la contaminación oceánica y el estado del arte de los sistemas de detección de vertidos, el Capítulo 5 se centra en el desarrollo del sistema implementado. La descripción detallada del problema a resolver (Sección 5.1) y las fuentes de datos empleadas (Sección 5.2) son la antesala para entrar en el detalle de las diferentes fases y algoritmos involucrados en el análisis e interpretación de las imágenes Radar de Apertura Sintética (SAR):

– Las diferentes acciones de preprocesado necesarias para preparar los datos SAR para su posterior análisis son descritas en la Sección 5.3.

– La segmentación de la imagen, relacionando las condiciones meteorológicas con la intensidad de la señal del satélite, se detalla en la Sección 5.4. Durante esta fase los posibles vertidos o candidatos son identificados y aislados.

– La Sección 5.5 describe el proceso por el que cada candidato es descrito empleando un conjunto de características.

– Cada candidato, a través de técnicas de Inteligencia Artificial (AI), es clasificado como falso positivo o vertido según sus descriptores. La implementación, uso y comparación de los diferentes clasificadores empleados es detallada en la Sección 5.6.

– La Sección 5.7 describe la aplicación de escritorio desarrollada para validar el sistema, así como su integración en RETELAB a través de un portlet.

La parte reservada al desarrollo de SENTINAZOS finaliza con un análisis detallado (Capítulo 6) y la discusión (Capítulo 7) de los resultados obtenidos.

Los capítulos finales de la tesis doctoral se reservan para mostrar los resultados y conclusiones del trabajo (Capítulo 8) y para perfilar las líneas de investigación abiertas por la tesis doctoral y que sería interesante desarrollar a corto y medio plazo (Capítulo 9).

Parte I

Laboratorio Virtual de Teledetección Oceanográfica

Capítulo 2

Evolución de la Computación Grid

2.1. Estandarización de las tecnologías Grid

La computación Grid (véase el Capítulo 1) es un paradigma de computación distribuida que surge como respuesta a dos necesidades: por un lado, la de disponer de entornos especializados en la resolución de problemas complejos con altas demandas computacionales y, por otro, la de gestionar VOs que permitan la colaboración entre grupos interdisciplinares y dispersos geográficamente.

El camino hacia su estandarización y globalización (véase la Figura 2.1) se inicia a principios de la década de los 90 a partir de implementaciones *ad hoc* que resolvían problemas concretos. Las aplicaciones resultantes aprovechaban la potencia computacional procedente de unir diferentes recursos distribuidos.

En esta primera etapa, la meta de los desarrolladores era hacer que funcionasen las aplicaciones y explorar nuevas posibilidades tecnológicas. La implementación se realizaba directamente sobre los protocolos de Internet sin seguir ningún tipo de estándar y, en general, las soluciones estaban muy limitadas en cuanto a seguridad, escalabilidad, robustez o interoperabilidad.

I-WAY [44] es considerado el primer Grid moderno que abrió la puerta al desarrollo y estudio de la computación Grid en profundidad [22] [9]. Se presentó en la conferencia ACM/IEEE Supercomputing del año 1995 (SC95) a través de una demostración en la que se conectaron 17 centros de supercomputación por medio de 11 redes de alta velocidad. Se creó un superordenador virtual sobre el que se probaron alrededor de 60 aplicaciones diferentes. I-WAY exploró varios aspectos que ahora se consideran esenciales en un Grid como la seguridad o

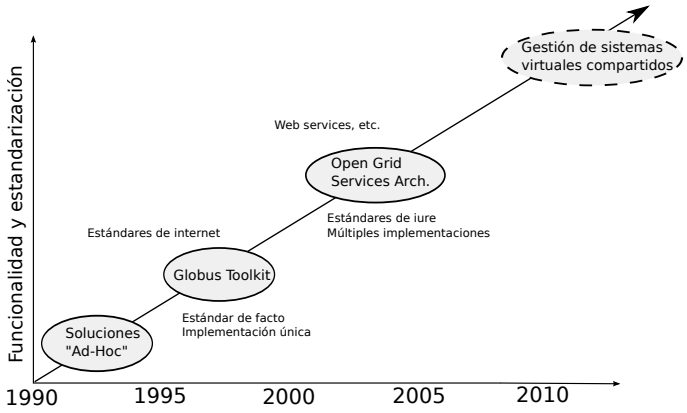

Figura 2.1: Evolución de las tecnologías Grid. Imagen adaptada de [50].

el envío y la gestión de trabajos. Su éxito demostró las posibilidades de la computación Grid y llevó a aumentar considerablemente los esfuerzos en el desarrollo de proyectos concebidos para compartir recursos de computación distribuidos.

I-WAY puede considerarse el germen del proyecto Globus [45], que surge con el objetivo de investigar y desarrollar tecnologías Grid fundamentales y marca el comienzo de la segunda etapa en la evolución hacia la estandarización. El proyecto, liderado por Ian Foster (codesarrollador de I-WAY y fundador del concepto de computación Grid), tiene uno de sus pilares en la implementación del *Globus Toolkit* (GT) [47]. El GT es un conjunto de componentes software diseñado para desplegar infraestructuras Grid y proporcionar servicios que gestionen aspectos tales como la seguridad, el envío de trabajos o la administración de los recursos.

El GT se sitúa en la capa de *middleware*, es decir, entre la capa de aplicaciones de usuario y la de recursos. Esta capa proporciona herramientas que permiten que los recursos participen de forma coordinada en un entorno unificado y aseguren el acceso transparente a los usuarios.

Además del GT, surgieron otros proyectos para el desarrollo de infraestructuras Grid como Legion [57] o *Uniform Interface to Computing Resources* (UNICORE) [94] pero, desde su primera versión en 1998, el GT se posicionó como el *middleware* dominante y, en 2002, su segunda versión (GT2) se estableció como el estándar *de facto* en el desarrollo de aplicaciones Grid.

El GT2 fue pionero en el desarrollo de sistemas Grid interoperables [50] pero, aunque algunos de sus componentes fueron desarrollados formalmente, en general, la implementación del GT2 carecía de directrices y no fue sometida a una revisión pública.

A medida que la computación Grid se establecía y se popularizaba (trascendiendo incluso al ámbito académico), se hizo más patente la necesidad de desarrollar estándares que aportasen total interoperabilidad no sólo a nivel de aplicaciones Grid, sino también de componentes fundamentales del sistema y permitiesen la integración de múltiples tecnologías en un sistema de recursos compartidos [73].

En el año 2002 surge la *Open Grid Service Architecture* (OGSA) [51], la cual define un estándar que incluye servicios básicos de computación Grid como la seguridad, gestión de recursos, administración de trabajos, etc. OGSA busca la convergencia entre la tecnología Grid y los servicios Web, por lo que define una arquitectura abierta basada en servicios para desarrollar entornos Grid. OGSA no aporta detalles para la implementación del estándar y, en consecuencia, surge una importante dificultad: OGSA requería servicios Web con estado pero el estándar de dichos servicios no lo proporcionaba.

En 2003 surge la *Open Grid Services Infrastructure* (OGSI), la cual detalla una posible implementación de OGSA y presenta una versión modificada de los servicios Web denominada servicios Grid [68]. Estos servicios proporcionaban, entre otras cosas, soporte para el estado gracias a una modificación del *Web Services Description Language* (WSDL) [30] llamada *Grid Web Service Definition Language* (GWSDL) [115]. La tercera versión del GT surge en 2004 y, en su camino hacia la estandarización, basa su implementación en el estándar definido por OGSI.

La comunidad Grid pronto consideró que OGSI era una aproximación poco conveniente [102], sobre todo por usar una versión modificada de los servicios Web en lugar de basarse en el estándar. Se continuó trabajando para encontrar una mejor solución para la implementación del estándar OGSA, en general, y para la implementación de servicios Web con estado, en particular. En 2004 se propuso la especificación *Web Service Resource Framework* (WSRF) con el objetivo de sustituir a OGSI y aportar una implementación de los servicios Web con estado aceptada tanto por la comunidad Grid como por la comunidad de servicios Web. WSRF integra OGSA con los servicios Web puros y es la base para la implementación de la cuarta versión del *Globus Toolkit* (GT4) (versión empleada en el desarrollo del laboratorio virtual de RETELAB).

El término de servicio Grid ha continuado utilizándose pero, a partir de la especificación de WSRF, se asocia a cualquier servicio que, de acuerdo a la nueva especificación, se ajuste a las convenciones de interfaz de una infraestructura Grid.

Continuando con el proceso de estandarización, Foster y Kesselman [50] anticiparon en el año 2003 una futura fase en la búsqueda de un sistema Grid global que denominaron gestión de sistemas virtuales compartidos. Partiendo de la especificación de OGSA, en esta fase se preveía un incremento del conjunto de servicios y sistemas estándar orientados a la interoperabilidad. Este nuevo conjunto de estándares estaría enfocado a la administración tanto de dispositivos móviles como de grandes sistemas con múltiples VOs, y todo ello aumentando el grado de virtualización, enriqueciendo los modos de compartición e incrementando la calidad de servicio.

2.2. Arquitectura de un sistema Grid

La arquitectura de un sistema Grid desarrollado con GT (véase la Figura 2.2) se divide en capas y está construida sobre los principios del modelo de reloj de arena (*hourglass model*) [50], en el que las capas intermedias, situadas en el cuello del reloj, contienen un conjunto de protocolos e interfaces que los desarrolladores de servicios y aplicaciones de más alto nivel (capas superiores) utilizan y que les permiten abstraerse de los desarrollos y tecnologías de las capas inferiores que gestionan el uso de los recursos locales.

- Capa de infraestructura (*Fabric layer*): proporciona las operaciones básicas sobre los recursos que serán utilizadas como base para el desarrollo de operaciones de más alto nivel.

- Capa de conectividad (*Connectivity layer*): define el núcleo de los protocolos utilizados para la comunicación y la autenticación requeridos en las transacciones.

- Capa de recursos (*Resource layer*): contiene los protocolos necesarios para gestionar de forma remota el acceso e interacción con los recursos individuales.

- Capa colectiva (*Collective layer*): contiene protocolos para la gestión de recursos múltiples. Utiliza los protocolos definidos en las capas inferiores para implementar sus operaciones.

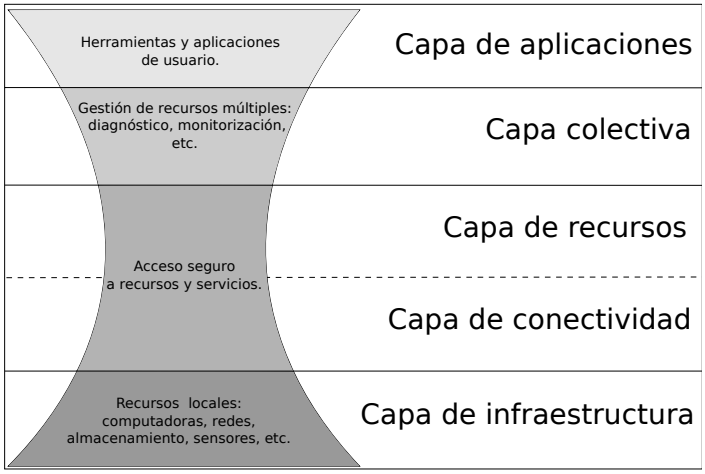

Figura 2.2: Arquitectura Grid basada en capas y construida sobre los principios del modelo de «reloj de arena». Figura adaptada de [50].

– Capa de aplicaciones (*Application layer*): esta capa contiene las aplicaciones de usuario que se implementan de forma específica para dar respuesta a un problema dentro del entorno de una organización virtual. Para su desarrollo se utilizan las interfaces desarrolladas en las capas inferiores.

2.3. Interfaces Grid

Existen dos tipos de usuarios relacionados con los sistemas Grid, desarrolladores y usuarios finales. Es habitual que los usuarios no tengan grandes conocimientos tecnológicos y, en particular, la formación en este ámbito es muy escasa. Éstos ven al sistema como un medio para desarrollar su trabajo, por lo que la usabilidad del entorno ha de ser primordial. La CLI es la solución básica para el acceso a un Grid pero es muy poco amigable y requiere de un estudio previo de los comandos y sus formas de uso, por lo que habitualmente es rechazada por los usuarios finales. Es conveniente desarrollar entornos que faciliten el acceso y utilización de los sistemas y, en la medida de lo posible, que abstraigan a los usuarios de los detalles tecnológicos para que puedan centrarse en sus trabajos.

Con el desarrollo de la Web 2.0 surge la tendencia de migrar aplicaciones y sistemas a la Web, convirtiendo Internet en un entorno amigable y conocido para el usuario. La compu-

tación Grid no fue ajena a este cambio y surgieron los llamados portales Grid que actuaban como interfaces de acceso a los sistemas distribuidos. El acceso Web suponía una evolución en cuanto a la generalización y popularización de los sistemas Grid, ya que permitía acceder a los usuarios en cualquier momento y desde cualquier lugar a través de una interfaz ergonómica y sin tener que instalar software específico. Desde el punto de vista de su finalidad, los portales Grid pueden dividirse en dos categorías:

– Portales Grid especializados: proveen de un subconjunto de operaciones Grid en el marco de una aplicación específica.

– Portales Grid de propósito general: proveen de un acceso básico y sin propósito concreto a servicios y recursos de un Grid.

Los desarrollos de portales Grid pueden dividirse en dos generaciones dependiendo de su implementación y tecnologías utilizadas [69]: la primera generación está caracterizada por desarrollos fuertemente acoplados con el *middleware* (habitualmente GT2), mientras que la segunda generación hace uso de tecnologías como, por ejemplo, los portlets, que permiten soluciones más configurables y dinámicas.

2.3.1. Portales Grid: primera generación

Surgen a mediados de los años 90 como una forma de hacer más accesibles y usables los sistemas Grid. Se realizaron grandes esfuerzos para implementar entornos de computación Grid que ofreciesen a través de la Web los mismos servicios que un sistema Grid clásico. Estos desarrollos, previos a los estándares (véase la Sección 2.1), son soluciones personalizadas poco modulares y fuertemente acopladas con el *middleware*. Esta primera generación aportó soluciones pero sobre todo un valioso conocimiento de la problemática que permitió allanar el terreno para desarrollos posteriores. A continuación se detallan algunas de las limitaciones más importantes [69]:

– Los portales tienen poca o ninguna posibilidad de personalización por parte del usuario final.

– El alto acoplamiento entre los portales y el *middleware* Grid hace que sea muy complicado integrar servicios suministrados por otro *middleware*.

– En esta época, el desarrollo de la computación Grid era muy dinámico. La integración de nuevas funcionalidades Grid no contempladas desde un primer momento era muy complicada.

Aunque no son los únicos, los ejemplos presentados a continuación son una muestra relevante de sistemas de primera generación. Es importante destacar que se describen las primeras versiones importantes pero que éstos han continuado evolucionando y cambiando, por lo que las limitaciones mencionadas no tienen que estar presentes en las sucesivas versiones.

UNICORE Uno de los primeros ejemplos importantes de portales Grid de propósito general fue UNICORE [94]. No dependía de ninguna implementación específica ni de servicios ni de un middleware Grid, sino que se trataba de una solución completa para la gestión y uso de una red de recursos heterogéneos y distribuidos. En los primeros prototipos, el acceso a UNICORE se proporcionaba a través de un applet disponible a través de la Web. Implementaciones posteriores descartaron el uso del applet en beneficio de una aplicación Java de escritorio [38].

GridPort 2.0 Es un *toolkit*, basado en *scripts* CGI de Perl, para desarrollar y desplegar rápidamente portales Grid [107]. Fue implementado pensando en potenciar la compatibilidad, a expensas de sacrificar rendimiento y flexibilidad, y por ello se utilizaron tecnologías básicas y se descartaron otras más complejas y actuales como los applets (en el lado del cliente) o *Java Server Pages* (JSP) y servlets (en el lado del servidor). Se utiliza GT para acceder a los recursos de alta computación y sus comandos son invocados a través de los *scripts*. La interfaz de usuario se construye a través de páginas HTML que contienen llamadas a los *scripts*. Tanto el uso de *scripts* y de HTML para modelar la interfaz, como la encapsulación de los servicios en los propios *scripts*, facilita el despliegue pero complica el mantenimiento y personalización del portal.

***Grid Portal Development Kit* (GPDK)** Es un *toolkit* que provee de una serie de componentes reutilizables que permiten interactuar con los recursos de un sistema Grid y desarrollar portales y aplicaciones con interfaces personalizables [85]. En la medida de lo posible, GPDK fue desarrollado a través de software libre y abierto. GT2 es el *middleware* utilizado para el acceso y gestión del sistema distribuido, mientras que el desarrollo del portal y las aplicacio-

nes se basan en JSP y servlets. La interacción entre Java y GT2 se realiza a través de *Java Commodity Grid Kit* (CoG) [113].

2.3.2. Portales Grid: segunda generación

El inicio de la segunda generación de portales Grid viene marcado por la aparición de los portlets o componentes reutilizables para portales Web y por el desarrollo de los estándares Grid basados en servicios.

Tecnologías empleadas

Portlets Como hemos visto en el apartado anterior, los portales de primera generación estaban muy limitados en cuanto a personalización y esto se solucionó, en gran medida, gracias a la aparición de los portlets. Según la primera versión de la especificación del estándar de desarrollo de portlets de Java (JSR168) [10]:

«*Un portlet es un componente Web desarrollado en Java y gestionado por un contenedor de portlets que procesa peticiones y genera contenido dinámico. Los portales utilizan portlets como componentes de la interfaz de usuario que se pueden activar o desactivar y que proporcionan una capa de presentación a los sistemas de información.*»

El usuario final ve al portlet como una ventana dentro de un portal que proporciona información o acceso a un servicio concreto como, por ejemplo, un calendario, noticias, o el acceso a un buzón de correo [61]. Un portlet genera un fragmento de código HTML que es incrustado en una sección del portal. La unión de los fragmentos generados por los diferentes portlets forman la interfaz del portal. Una característica importante de este tipo de portales es la posibilidad de personalización que ofrecen al usuario final, ya que el contenido generado puede variar dependiendo de la configuración e incluso es posible administrar qué portlets están activos y cuáles no (véase la Figura 2.3). Es fácil imaginarse un portal como un puzzle que puede tomar cualquier forma (la elige el usuario final) y en el que se pueden intercambiar piezas, siendo cada pieza un portlet. Lo interesante es que estas piezas pueden ser reutilizadas para formar parte de nuevos portales, pues lo único que se exige es que el portal de destino sea compatible con el *Application Programming Interface* (API) estándar de desarrollo de portlets. Cada portlet funciona de forma independiente respecto al resto y tiene su propio ciclo de vida que es gestionado por el contenedor de portlets.

Contenedor de portlets Podemos definir un contenedor de portlets como un entorno de ejecución en donde los portlets pueden ser instanciados, usados y finalmente destruidos. Es responsabilidad del contenedor la gestión del ciclo de vida del portlet [69]. Un contenedor de portlets no puede ser usado de forma independiente, sino que se despliega dentro de un contenedor de servlets y aprovecha sus funcionalidades.

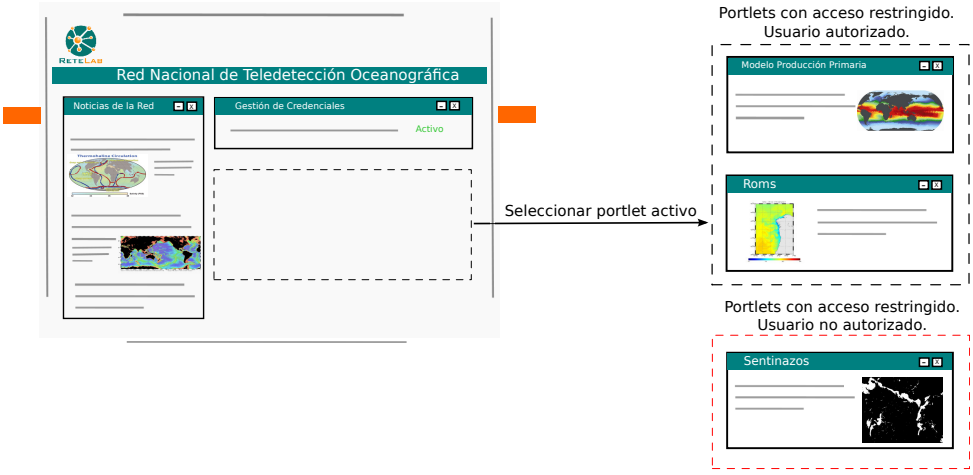

Figura 2.3: Ejemplo de un portal Web desarrollado a través de portlets.

Portlets Grid

Un portlet Grid es un tipo especial de portlet que está asociado a un servicio Grid proporcionado por un *middleware*. Gracias a la estandarización tanto de los portlets como de los servicios Grid, un portlet puede ser reutilizado e interactuar con un servicio Grid sea cual sea el *middleware* que lo hace accesible, por lo que se puede afirmar que los portales Grid desarrollados con portlets se encuentran débilmente acoplados al *middleware*.

Entornos para el desarrollo de portales Web basados en portlets

Los entornos diseñados para desarrollar portales basados en portlets no son específicos de la computación Grid, es decir, son genéricos y pueden contener portlets de cualquier tipo (incluido portlets Grid). WebSphere y GridSphere son, probablemente, los entornos de mayor popularidad:

IBM WebSphere Portal Es una aplicación J2EE que se ejecuta sobre un servidor de aplicaciones con el mismo nombre y que permite desarrollar y administrar portales Web centrados en el acceso a aplicaciones, contenidos y servicios a través de portlets [69]. Es una implementación propietaria que actualmente se encuentra en su octava versión. Durante sus primeras versiones, el desarrollo de portlets estaba basado en el API de IBM que, aunque fundamentalmente era similar al API JSR168, sus pequeñas diferencias [59] impedían la total compatibilidad de los portlets. El soporte para el API JSR168 fue incluido a partir de la versión 5.0.2 y la versión actual soporta tanto el API JSR168 como el API JSR286.

Con el portal se distribuye un *toolkit* que permite su administración y personalización, así como la creación, prueba, depuración y despliegue de portlets para un rápido desarrollo. El IBM WebSphere Portal también provee de mecanismos para la personalización de la vista del portal por el usuario final, permitiendo seleccionar las aplicaciones y contenidos, así como establecer la forma en que son mostrados.

GridSphere Es uno de los entornos de desarrollo de portales basados en portlets más conocidos y utilizados [87] y, concretamente, fue el empleado para desplegar el laboratorio virtual de RETELAB. Se trata de un proyecto de código libre desarrollado en el marco del proyecto GridLab que incluye un contenedor de portlets, un conjunto básico de portlets y librerías de desarrollo.

GridSphere es compatible con el API de WebSphere y con el API estándar para portlets, lo que permite una rápida integración del software desarrollado por terceros. Cuando el proyecto fue iniciado, la primera versión del estándar todavía no se había publicado, por lo que, de entrada, fue construido conforme al de IBM. La compatibilidad con el estándar fue añadida en una versión posterior.

El conjunto básico de portlets incluido en GridSphere facilita el desarrollo de nuevos portales y aporta funcionalidades básicas como la administración de usuarios y grupos, el acceso de los usuarios al portal, la gestión de las cuentas o la personalización del portal.

Entornos y herramientas para el desarrollo de portlets Grid

Para facilitar la implementación de portlets Grid y su despliegue en los portales, surgieron proyectos específicos que proporcionaban tanto entornos de desarrollo como *toolkits*.

GridPortlets y *Open Grid Computing Environments* (OGCE) son dos de los entornos de desarrollo más populares y extendidos. Ambos aportan herramientas básicas que permiten a los desarrolladores interactuar con el *middleware* facilitando el desarrollo de portlets Grid.

OGCE Es un proyecto de código libre que proporciona un ecosistema en el que se integran diferentes tecnologías y herramientas para la creación de portales Grid [12] [116].

Varias son las líneas en las que OGCE ha centrado sus esfuerzos:

- Establecer un entorno de desarrollo de portlets Grid: desde la segunda versión del OGCE, los portlets desarrollados cumplen con el estándar y pueden ser reutilizados. Para facilitar su implementación, OGCE proporciona una utilidad que permite la creación de portlets estándar a través de la herramienta de desarrollo Web denominada Velocity [4].
- Abstraer a los desarrolladores de aplicaciones del sistema Grid utilizado: OGCE no utiliza un único API para acceder a los diferentes servicios Grid, sino que se basa en un conjunto procedente de varios proyectos como CoG o GridPort. CoG se puede considerar el API principal y proporciona una capa de abstracción que independiza a los portlets del *middleware* utilizado.
- Desarrollar servicios Grid avanzados: la segunda versión de OGCE integra y proporciona servicios para, por ejemplo, monitorizar los recursos locales utilizando el *GridPort Information Repository* (GPIR), incorporar el repositorio semántico de contenido Tupelo [5] o generar interfaces de servicio Web para las aplicaciones científicas que se desee integrar en el portal.
- Desarrollar servicios para la colaboración de grupos: OGCE desarrolló servicios y diseñó interfaces portlet para integrar las herramientas colaborativas Sakai [3].

OGCE ha sido el entorno utilizado para el desarrollo de los portlets en RETELAB; además, también se incorporaron al laboratorio varios portlets ya implementados por OGCE como el portlet para la administración de credenciales o el dedicado a la gestión de ficheros remotos.

GridPortlets Surge dentro del proyecto GridSphere con el objetivo de proporcionar a los desarrolladores un entorno para la implementación de portlets Grid [97]. GridPortlets distribuye una colección de portlets básicos que pueden ser utilizados individualmente o junto a otras aplicaciones y que permiten desplegar un portal Grid básico. Este conjunto de portlets

está implementado a través de componentes JSP reutilizables que pueden ser integrados en nuevas implementaciones.

Al contrario de OGCE, que utilizaba varias APIs, GridPortlets proporciona una única API de alto nivel pensada para acceder a los servicios Grid e interactuar con los recursos y así desarrollar portlets y aplicaciones Grid específicas.

Los portlets desarrollados a través de GridPortlets no son completamente independientes de GridSphere y, aunque teóricamente son compatibles con el API estándar, no se pueden integrar fácilmente en otros contenedores de portlets [116]. GridPortlets, pensando en la usabilidad, proporciona una sencilla pero potente interfaz para el envío de trabajos que abstrae al usuario de los detalles de bajo nivel y almacena un detallado historial de cada trabajo procesado para que pueda ser consultado a posteriori.

2.3.3. Portales Grid: casos de estudio

Durante la primera década del siglo XXI, sobre todo en los primeros años, la incidencia de los sistemas Grid en la sociedad fue en aumento, no sólo en el campo académico, sino también en el empresarial. La estandarización de las tecnologías Grid y la mejora de las interfaces de usuario hicieron de esta tecnología una valiosa herramienta. Fueron muchos los sistemas Grid desplegados y numerosos también los portales Grid desarrollados para interactuar con los sistemas de una forma amigable. Una gran mayoría de estos portales estaban centrados en la visualización de información, monitorización de los recursos y sistemas y en la ejecución de tareas simples; no obstante, algunos otros profundizaron en la tecnología y desarrollaron aplicaciones Web que facilitaban la realización de tareas Grid complejas a los usuarios.

GENIUS Grid Portal La Organización Europea para la Investigación Nuclear (CERN), en general, y el proyecto LHC, en particular, han ejercido una gran influencia en el desarrollo de la tecnología Grid. El portal Grid GENIUS [19] [15], implementado dentro del marco del proyecto europeo DataGrid, fue desarrollado para proporcionar acceso a los datos del LHC y permitir su utilización en diferentes aplicaciones sobre un sistema Grid. Estaba diseñado para proporcionar un acceso seguro a funcionalidades Grid como el envío y ejecución de trabajos, la administración de datos o el acceso interactivo para el análisis de los datos generados.

GENIUS fue un portal Grid pionero basado en una versión extendida del GT. Este *middleware* modificado extendió la interfaz de Globus facilitando la interacción entre las aplicaciones software y el sistema Grid. La primera versión del portal se puede considerar dentro

de la primera generación de portales Grid (véase la Sección 2.3.1), ya que su desarrollo está fuertemente acoplado con el *middleware* utilizado.

El portal permitía el acceso remoto de usuarios pero mantenía la necesidad de tener una cuenta personal en el servidor del portal con la que se proporcionaba acceso al sistema de ficheros local. La autenticación y autorización de los usuarios sobre los recursos Grid se realizaba a través de la delegación de credenciales implementada con MyProxy (véase la Sección 3.2)

***Earth System Grid* (ESG)** En general, los proyectos relacionados con el estudio de la Tierra poseen una serie de particularidades que dificultan su desarrollo a través de medios tradicionales:

- Generan cantidades ingentes de información, bien sea a través de la observación directa de fenómenos o bien a través de la ejecución de modelos que los simulan.
- Los algoritmos y modelos utilizados suelen requerir una gran potencia computacional.
- Implican ciencias interdisciplinares en donde trabajan científicos de diferentes ramas que deben colaborar para obtener el máximo rendimiento de las investigaciones.

La computación Grid se adapta especialmente bien a este tipo de proyectos y, en concreto, el proyecto ESG [23] es uno los más relevantes. Centrado en la investigación climática, ESG proporciona un entorno colaborativo con el objetivo de almacenar, distribuir y procesar las enormes cantidades de datos utilizadas en el seno de sus proyectos.

Además de proporcionar la infraestructura necesaria para el almacenamiento de la información climática, gran parte de los esfuerzos del proyecto fueron destinados a desarrollar estándares y servicios de metainformación orientados a administrar, navegar, buscar y compartir los datos almacenados. Un portal Grid facilita el acceso a los datos y los servicios de búsqueda y recuperación de información.

El acceso a los recursos Grid se basa en el sistema de delegación de credenciales de MyProxy (véase la Sección 3.2) pero ESG desarrolló y empleó un novedoso sistema de registro de usuarios para la generación automática de credenciales. El *Portal-based User Registration Service* (PURSe) [53] permite el registro de usuarios y la creación y administración de sus Certificados Digitales (DCs) y respectivos *proxies* de forma automática. Originalmente fue desarrollado dentro del proyecto pero, posteriormente, evolucionó como sistema independiente e integrable en nuevos portales Grid.

PURSe encaja perfectamente con la política de facilidad de uso que dirige el proyecto RETELAB, por lo que fue integrado como medio de registro de usuarios. No obstante, el módulo de registro fue modificado y mejorado para dar un soporte más amplio que el proporcionado por defecto y así ajustarse a las necesidades del laboratorio virtual. Los detalles de implementación y de integración se muestran en detalle en la Sección 3.2.

P-Grade P-Grade es un portal Grid, basado en GridSphere, concebido para soportar formas complejas de ejecución de trabajos dirigidas por flujos de tareas (workflows). P-Grade está diseñado para trabajar con Globus como *middleware*, lo que le permite integrarse con cualquier sistema Grid que lo utilice y aportarle la capacidad avanzada de administración de flujos de tareas. Proporciona a los usuarios un editor de flujos que permite diseñar localmente las tareas y subirlas al servidor. Empleando *Java Web Start* (JavaWS), los usuarios pueden descargar el editor desde el portal y éste se instala automáticamente y de forma transparente en la estación de trabajo del usuario. Dentro de un mismo flujo de tareas, P-Grade soporta la asignación de trabajos a recursos que pertenezcan a diferentes Grids e incluso la creación de flujos colaborativos entre varios investigadores. Es tarea del usuario la asignación de los recursos en concreto en donde se vaya a ejecutar una parte del flujo, por lo que el usuario debe conocer la estructura del sistema.

CAPÍTULO 3

PROYECTO RETELAB

3.1. Descripción del proyecto

RETELAB es un laboratorio virtual, desplegado sobre un sistema distribuido, que proporciona un entorno de trabajo colaborativo enfocado al desarrollo de proyectos de teledetección oceanográfica [78] [33]. Nace en el seno de la red RETEMAR con el objetivo proporcionar un entorno para llevar a cabo proyectos de altas prestaciones entre sus integrantes.

Según las especificaciones iniciales, los requerimientos que debía cubrir el laboratorio virtual eran:

- Gestión de recursos compartidos: el laboratorio virtual debía poseer la capacidad de integrar y administrar recursos distribuidos pertenecientes a los diferentes centros de la Red.

- Soporte para el almacenamiento, administración e intercambio de datos distribuidos: el almacenamiento local tiene grandes limitaciones a la hora de administrar las ingentes cantidades de datos capturados a través de la teledetección, por lo que RETELAB debía disponer de un sistema de datos distribuidos que permitiese almacenar, procesar y compartir la información.

- Potencia computacional: en general, las aplicaciones científicas oceanográficas (modelos climáticos, detección de contaminantes, etc.) demandan una gran cantidad de recursos computacionales. El sistema proporcionaría la potencia necesaria para su ejecución.

– Seguridad: dado que el laboratorio proporcionaría acceso a recursos compartidos, la seguridad debía ser uno de los pilares en el diseño e implementación de RETELAB.

– Interfaz amigable: es necesario que la interfaz de acceso al sistema sea intuitiva y fácil de utilizar para que los usuarios puedan centrarse en sus trabajos sin tener que dedicar esfuerzos al aprendizaje de nuevas herramientas.

– Visualización: el laboratorio virtual debía proporcionar las herramientas necesarias para visualizar tanto los resultados de los trabajos como los datos almacenados.

Las especificaciones del sistema, junto con las características típicas de las aplicaciones que se desea ejecutar, hacen de la computación Grid la tecnología más adecuada para el desarrollo del laboratorio virtual y el acceso a través de un portal Grid, la opción más simple y amigable para el usuario final.

3.1.1. Arquitectura general del sistema

El diseño de RETELAB se corresponde con una arquitectura cliente-servidor multicapa (véase la Figura 3.1) en la que una aplicación cliente, típicamente un navegador Web, realiza peticiones a un servidor que las procesa y devuelve un resultado. A continuación se describen brevemente las principales capas del sistema:

– Capa de aplicaciones: en esta capa se sitúa el servidor Web (Tomcat + Apache) en el que está desplegado el portal Grid a través del cual se accede al sistema. El portal, basado en el contenedor de portlets GridSphere, contiene diferentes portlets que encapsulan y gestionan tareas sobre los recursos de alta computación.

– Capa de *middleware*: en esta capa está desplegada la cuarta versión del *Globus Toolkit* que contiene la tecnología y herramientas necesarias para gestionar un Grid. GT4 proporciona una colección de servicios que permiten a las aplicaciones (capa superior) operar sobre los recursos de alta computación (capa inferior).

– Capa de recursos: en esta capa se sitúan los recursos del sistema. Se trata de recursos distribuidos y heterogéneos que pertenecen a diferentes centros.

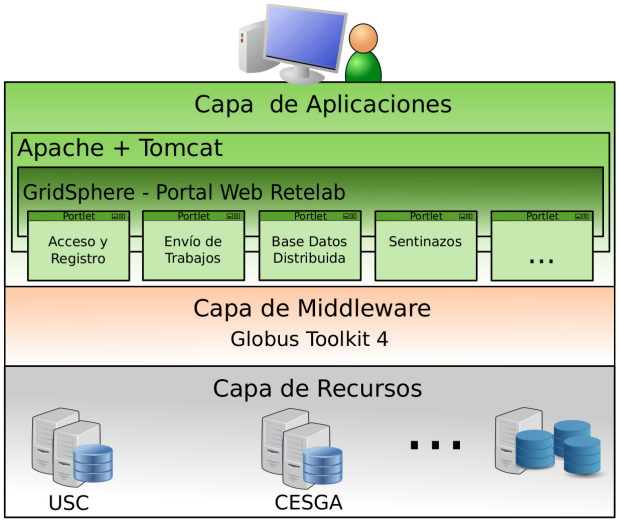

Figura 3.1: Arquitectura general del sistema RETELAB.

Recursos del sistema

El sistema Grid de RETELAB permite incluir recursos de cualquier centro participante en el proyecto pero en el despliegue inicial sólo se ponen a disposición de la comunidad recursos de la USC y del CESGA.

La USC colaboró con un cluster compuesto por 11 nodos, 10 AMD Opteron 2212 y 1 INTEL Xeon 5130, cada uno de los cuales presenta las siguientes características: arquitectura x86_64, 2 procesadores dual core, frecuencia de 2GHz, 4GB RAM e interconexión Giga-Ethernet.

El CESGA aportó un cluster compuesto de 4 nodos HP Proliant de las siguientes características: arquitectura x86_64, 2 procesadores Intel Xeon Quadcore X5355 (8 cores/equipo), frecuencia de 2,66GHz, 8 GB de RAM, 4 x 2MB caché (L2) e interconexión GigaEthernet.

Los recursos proporcionados por el CESGA se aprovecharon al máximo gracias a la plataforma de virtualización Xen [6]. Tal y como se muestra en la Figura 3.2, uno de los nodos del cluster fue virtualizado para alojar a 3 recursos virtuales diferentes:

– RETELAB UI: aloja la interfaz de acceso al laboratorio virtual, es decir, el portal Grid.

– CESGA CE: gestiona los trabajos que le llegan al cluster actuando como nodo principal o «cabeza».

– Nodo de trabajo: se ha virtualizado un nodo de trabajo que, si bien no tiene la potencia computacional del resto de los nodos, es aprovechado y utilizado como parte del cluster.

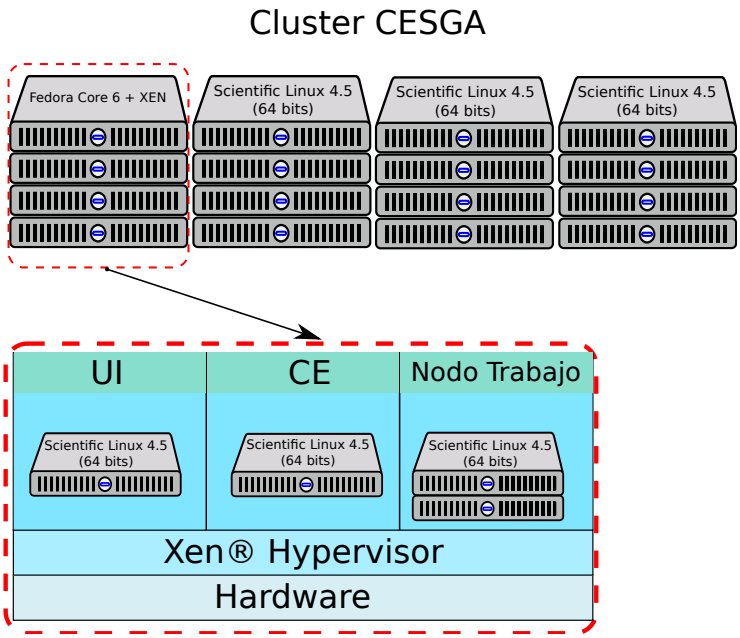

Figura 3.2: Recursos aportados por el CESGA.

3.2. Registro y acceso de los usuarios

El objetivo de esta sección es mostrar el módulo para el registro y acceso de los usuarios de RETELAB al sistema pero, para una mejor comprensión, es necesario introducir brevemente algunos conceptos sobre la infraestructura de seguridad del GT.

Security Grid Infrastructure (GSI)

La GSI es un conjunto de herramientas, librerías y protocolos usados por el GT para gestionar el acceso de los usuarios y aplicaciones a los recursos de altas prestaciones [70]. Los pilares fundamentales del GSI son la PKI y los Certificados Digitales (DCs) X.509.

Cada participante de un sistema Grid dispone de un DC X.509 que lo identifica y que, principalmente, está compuesto por:

- Un nombre que identifica a la persona o recurso al que representa el DC.
- La clave pública del propietario del DC.
- El nombre de la Autoridad Certificadora (CA) que ha firmado el DC y que garantiza la identidad del propietario. Es esencial que todos los participantes de un Grid confíen y tengan acceso a las claves públicas de las CAs utilizadas para firmar los DCs.
- La firma digital de la CA que valida el DC .

Es importante diferenciar dos términos en el ámbito de la seguridad Grid:

- Autenticar: proceso de identificar unívocamente a un usuario o a un recurso.
- Autorizar: proceso que identifica las acciones que puede realizar un usuario sobre un recurso concreto.

Autenticación El proceso de autenticación de participantes en un sistema Grid es mutuo, es decir, para establecer una comunicación entre dos participantes es necesario que cada uno de ellos se identifique ante el otro a través del siguiente método:

1. La entidad ‘A’ establece una conexión con la entidad ‘B’ y le envía su DC.
2. La entidad ‘B’ comprueba la autenticidad del DC utilizando la clave pública de la CA con la firma del certificado.
3. Una vez validado el certificado, ‘B’ trata de confirmar que la entidad que envió el DC es realmente su propietaria y para ello envía un mensaje aleatorio para que ‘A’ lo codifique utilizando su clave privada.
4. La entidad ‘A’ utiliza su clave privada para codificar el mensaje y éste es devuelto a ‘B’.

5. 'B' utiliza la clave pública de 'A', contenida en el DC, para descodificar el mensaje y comprobar que es el mismo que el original.

6. Una vez que 'A' es autenticada, 'B' iniciaría nuevamente el proceso enviando su DC a 'A'.

Cuando un usuario desarrolla una tarea en un Grid, ésta puede requerir varios recursos o puede que, durante su procesamiento, necesite solicitar servicios en nombre del usuario. Según el método de autenticación mostrado, cada vez que el usuario necesite un recurso o requiera un servicio debería autenticarse. El uso de la clave privada exige la atención del usuario, ya que se almacena cifrada y es necesario que el usuario introduzca su contraseña cada vez que sea necesaria su autenticación. Esto puede resultar tedioso en tareas complejas, sobre todo si se dilatan en el tiempo. Para solucionarlo, GSI introduce el concepto de delegación. Este mecanismo permite al usuario asignar un *proxy* a la tarea deseada y evita tener que presentar su DC cada vez que la tarea realiza una operación en su nombre. Los *proxies* son credenciales, con un tiempo de vida muy reducido, generadas por los usuarios a partir sus DCs. La identidad de los *proxies* es la misma que la de los DCs salvo una pequeña modificación que indica su naturaleza. La creación de un *proxy* implica la generación tanto de una clave pública como de una privada que se almacenan sin cifrar y a las que se puede acceder sin contraseña evitando la acción del usuario.

Autorización Por defecto, GSI proporciona unos sistemas de autorización muy básicos de entre los que cabe destacar el sistema gestionado por el GridMap. Cada recurso posee un fichero denominado GridMap que contiene una lista que relaciona nombres de usuario (identificadores presentes en los DCs) con cuentas de usuario locales. Si el usuario tiene una cuenta local, entonces puede utilizar dicho recurso. Es conveniente disponer de sistemas que ayuden a la creación y gestión de dichas listas y de las cuentas correspondientes.

La ventaja es que GSI también proporciona mecanismos para el desarrollo e integración de sistemas de autorización propios y esto será aprovechado para gestionar la autorización de los usuarios de RETELAB.

MyProxy

MyProxy [86] es un repositorio *on-line* de certificados cuya finalidad es proporcionar *proxies* temporales bajo demanda para que puedan ser utilizados en cualquier momento y

desde cualquier lugar. MyProxy puede ser utilizado para delegar credenciales a servicios que actúen en lugar del usuario y es el método más habitual para delegar y usar credenciales en los portales Grid. MyProxy puede ser utilizado de dos modos diferentes:

– Repositorio de credenciales de usuario: MyProxy actúa como almacén de los DCs y de las claves privadas de los usuarios. Abstrae al usuario del uso y almacenamiento de los certificados y permite generar *proxies* temporales bajo demanda. Este uso implica que el usuario delega la seguridad de sus DCs en el servidor.

– Repositorio de *proxies*: MyProxy almacena los *proxies* generados por los usuarios a través de sus DCs y genera nuevos representantes con una vida más corta que los originales. Es necesario que los usuarios generen y alojen nuevos *proxies* cada vez que los almacenados caduquen.

3.2.1. Autenticación y autorización

Uno de los requisitos fundamentales para la construcción del laboratorio virtual (véase la Sección 3.1) era el desarrollo de una interfaz de usuario sencilla y amigable que no demandase grandes conocimientos tecnológicos. La sencillez de un sistema debe empezar desde el registro y acceso de los usuarios pero, a menudo, esto entra en conflicto con los requerimientos de seguridad, por lo que fue necesario encontrar una solución equilibrada.

La seguridad en RETELAB está basada en el modulo GSI del GT4, lo que condiciona el acceso al sistema y supone que los usuarios deban disponer de un DC. El proceso para obtener y emplear un DC es, cuanto menos, tedioso para el usuario y puede producir rechazo al uso del sistema, por lo que es necesario adaptarlo para que sea más cómodo. Por otro lado, es importante que el sistema de autorización sea sencillo de administrar pero lo suficientemente potente para responder a las siguientes preguntas:

– ¿Quién puede acceder?

– ¿Qué recursos puede utilizar?

– ¿Qué tareas puede realizar sobre un recurso concreto?

Un RBAC [91] parece ser la mejor alternativa. En RBAC los permisos se encuentran asociados a roles y los usuarios son poseedores de uno o más roles, por lo que adquieren los permisos asociados a éstos. Los roles permiten reflejar con mayor veracidad la estructura de una VO y el

puesto o tareas desempeñadas por un investigador concreto dentro de ella. La administración del control de acceso es considerablemente más sencilla que si se gestiona la seguridad a nivel de usuario.

Registro y Acceso

El registro y acceso de los usuarios de RETELAB se ha implementado integrando y mejorando diferentes sistemas para conseguir una solución amigable sin sacrificar la seguridad [76] [77]. La Figura 3.3 muestra un diagrama en donde se aprecian los diferentes sistemas integrados.

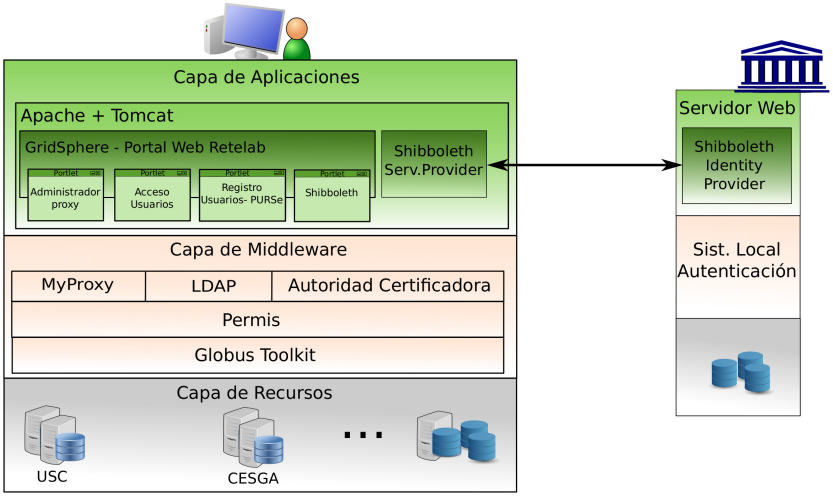

Figura 3.3: Arquitectura del sistema de registro y acceso de usuarios de RETELAB.

El acceso al portal puede contemplarse desde tres escenarios diferentes (véase la Figura 3.4): acceso de un usuario no registrado, acceso de un usuario registrado y acceso de un usuario perteneciente a una red de centros de confianza.

Usuario no registrado

El registro de un nuevo usuario se realiza a través de un portlet de registro basado en el sistema PURSe [53] [2]. Este sistema permite registrar a un usuario y solicitarle un DC de forma automática en el mismo momento en que envía sus datos. Por cada registro, el

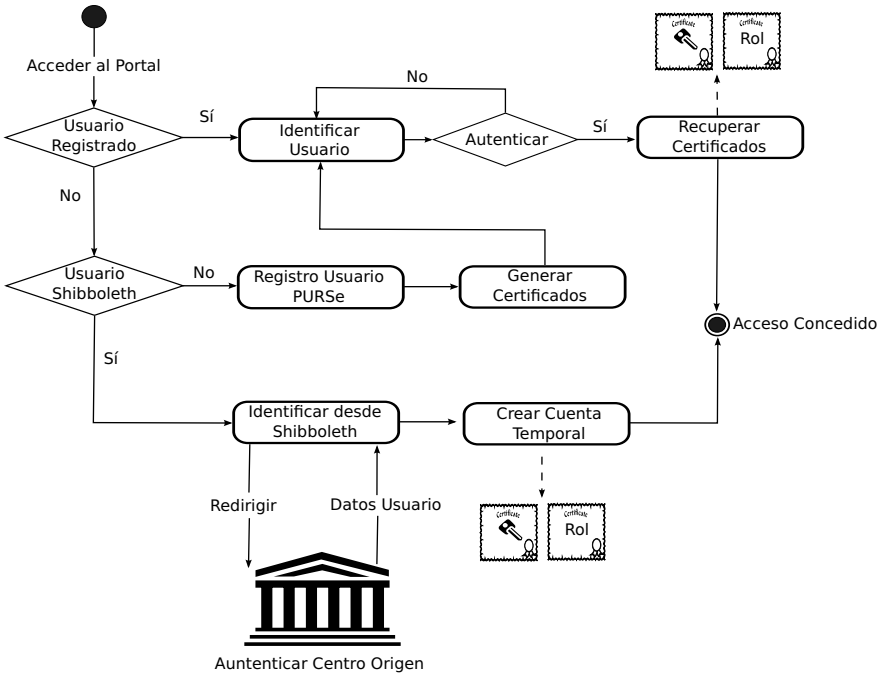

Figura 3.4: Esquema simplificado del registro y acceso de usuarios al sistema.

administrador del portal recibe un aviso para que valide los datos del nuevo usuario y, si es oportuno, se genera el DC a través de una CA propia del sistema. Los certificados y las claves privadas son almacenados en MyProxy e identificados con el nombre y contraseña proporcionados por el usuario al registrarse.

Para integrar el RBAC, fue necesario implementar una mejora sobre PURSe para que pudiese dar soporte a los roles de usuario. Gracias a esta mejora, el administrador puede asignar roles a los usuarios que gestionan sus accesos tanto al portal como a los recursos del sistema. La mejora realizada permite crear automáticamente Certificados de Atributos (ACs) X.509, con información sobre los roles de los usuarios, y almacenarlos en un árbol de directorios *Lightweight Directory Access Protocol* (LDAP) para su posterior uso en la gestión del acceso.

Detalles de implementación La implantación de un sistema RBAC en RETELAB se realizó intentando que tuviese el menor impacto posible sobre el resto de los subsistemas

integrados con el objetivo de que fuese reutilizable y sencillo de instalar. En el proceso se intentó aprovechar al máximo el código libre desarrollado por los proyectos integrados (PURSe, PERMIS, GridSphere, etc.) e implementar nuevas clases que sirviesen de nexo entre las diferentes tecnologías.

Para integrar los roles con el sistema de registro PURSe, fue necesario implementar varias clases Java (véase la Figura 3.5) que se encapsularon como un ARchivo Java (JAR), generar un archivo de propiedades con los datos de acceso al LDAP y añadir un nuevo módulo de registro en el fichero web.xml de PURSe. El diagrama mostrado en la Figura 3.5 contiene las clases implicadas en la asignación de roles. El módulo de registro es el nexo con el paquete PURSe y el encargado de «escuchar» los eventos y realizar las acciones consecuentes. PURSe proporciona un módulo de registro para integrarse con GridSphere que da de alta a los usuarios en el portal pero, además, permite añadir nuevos módulos siempre y cuando implementen la interfaz *RegistrationModule*. El módulo *RetelabRegistrationModule* fue desarrollado específicamente para dar soporte a los ACs. Cuando un nuevo usuario es dado de alta a través de PURSe, el evento es capturado y gestionado por la clase *RetelabRegistrationModule*. Los eventos de PURSe se reciben como objetos que contienen tanto la información de la acción como los datos del usuario. Utilizando la información personal y la través de la clase *CertificateGenerator* se genera un AC que es firmado por la clase *RetelabSigningUtility*. Una vez se ha generado un AC, la clase *LdapConnector* es utilizada para almacenarlo en el directorio LDAP. Las clases dedicadas tanto a la generación como a la firma de los ACs fueron implementadas basándose en clases de PERMIS, que es el software empleado para gestionar el acceso de los usuarios registrados a los recursos a través de roles.

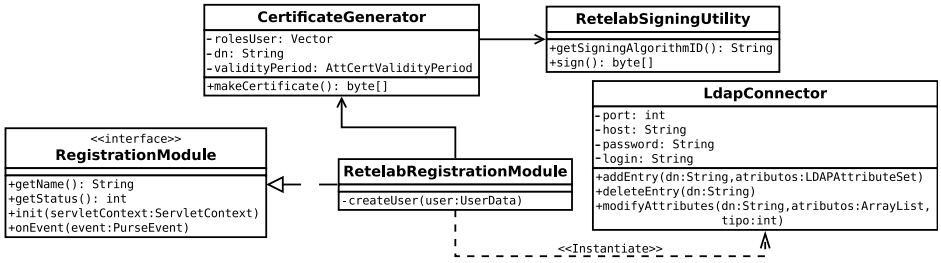

Figura 3.5: Diagrama de clases simplificado para la creación y asignación de roles.

La solución desarrollada utiliza DCs generados por PURSe y ACs creados a través de clases basadas en procedimientos empleados por PERMIS. Es interesante destacar un problema

surgido de la integración de los dos tipos de certificados: para la creación de los DCs se utilizó el proveedor de Extensión Criptográfica de Java (JCE) denominado Cryptix, mientras que para los ACs se utilizó el JCE IAIK. Para utilizar un proveedor JCE es necesario registrarlo y, concretamente, un error en la implementación de Cryptix impedía su uso en caso de que algún otro proveedor fuese registrado con posterioridad. Al utilizar varios JCEs se generaban errores frecuentemente, por lo que fue necesario insertar un procedimiento previo a su utilización para comprobar el orden de registro y modificarlo en caso de que fuese necesario (véase el Código A.1).

Una de las ventajas de usar GridSphere como base para el desarrollo del portal fue que disponía de una serie de portlets ya desarrollados para crear portales Web básicos. Del núcleo de portlets de GridSphere cabe destacar el portlet de administración que, entre otras cosas, permite crear y asignar roles a los usuarios del portal. GridSphere, en base a dichos roles, gestiona el uso y acceso de los usuarios a los portlets.

Dado que la gestión del sistema Grid también está basada en roles y con el objetivo de desarrollar un sistema de administración sencillo y usable, se implementó una mejora sobre el sistema de administración de GridSphere para dar soporte a los ACs. De este modo, un administrador puede gestionar los roles tanto del portal como del sistema Grid desde un punto único, ya que en el momento en que un rol es asignado el AC correspondiente es creado y almacenado automáticamente en el directorio LDAP.

Para dar soporte a los ACs fue necesario modificar la clase de GridSphere *UserManagerPortlet*. El fragmento de Código A.2 muestra los cambios realizados en el método que añade usuarios a GridSphere.

Usuario registrado

Los usuarios con cuenta en RETELAB pueden acceder al portal autenticándose en un portlet dedicado a tal fin. Una vez que el usuario es identificado, el sistema utiliza su nombre y contraseña para recuperar sus credenciales almacenadas. Las acciones que el usuario puede realizar sobre el sistema están guiadas por su rol y una política de autorización basada en un modelo RBAC. PERMIS [28] [27] es el sistema de autorización basado en roles utilizado para gestionar el acceso de los usuarios a los servicios Grid. Su integración con el GT4 se realiza gracias a que éste permite configurar, para cada servicio Grid, una secuencia de mecanismos de autorización denominados *Policy Decision Point* (PDP). PERMIS puede desplegarse como un PDP para añadir un control basado en roles al sistema. Para gestionar

la autorización de la ejecución de tareas en un recurso específico de acuerdo a un rol, es necesario modificar la política de autorización del método *createManagedJob* en el fichero *managed-job-factory-security-config.xml* y añadir el PDP de PERMIS como se muestra en el fragmento de Código A.3.

El control de acceso de un usuario a un servicio o recurso Grid se realiza desde 3 puntos, como puede verse en la Figura 3.6.

1. Se comprueba si el usuario posee un rol suficiente para acceder a un determinado portlet.
2. Se comprueba que el usuario tenga un DC activo en el portal.
3. Si el usuario invoca un servicio Grid, PERMIS comprueba su rol y, dependiendo de la política de seguridad del recurso solicitado, se permite o no su uso.

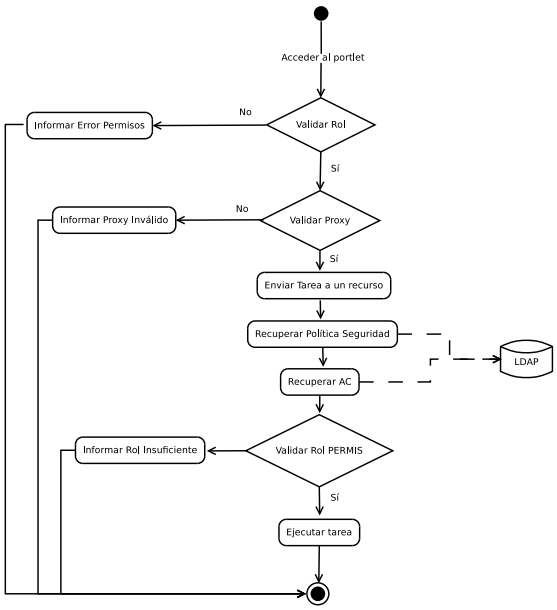

Figura 3.6: Esquema del control de acceso basado en roles de RETELAB.

Acceso confiable

Actualmente y debido a la gran proliferación de sistemas a través de Internet, cada persona gestiona multitud de cuentas y registros, lo que supone recordar una gran cantidad de nombres y contraseñas. En muchos casos, se acaba sacrificando la seguridad, anotando las contraseñas o utilizando siempre los mismos datos para facilitar la tarea.

Shibboleth [83] es un software que proporciona un sistema de autenticación *Single-Sign On* (SSO) a través de la Web. Esto significa que permite acceder a varios sistemas independientes a través de una única identificación. Shibboleth se basa en una red confiable de socios en los que cada uno tiene su propio sistema de autenticación. A grandes rasgos, su funcionamiento es el siguiente: cuando un usuario perteneciente a uno de los centros de la red quiere acceder a un sistema de otro centro, éste último lo redirige al sistema de autenticación del centro de origen. Una vez que el usuario es autenticado, el centro de origen manda los datos personales al centro de destino que permite o deniega el acceso en base a la información recibida.

La integración de Shibboleth con RETELAB tiene como punto de partida la solución diseñada por el grupo *Meta Access Management System* (MAMS) para su utilización junto a GridSphere. En RETELAB se ha partido de la versión de MAMS y se ha desarrollado una mejora para registrar y generar automáticamente los DCs y ACs de los usuarios que accedan a través de esta opción. Los datos para generar los certificados deberán ser enviados por el centro que lo autentica. La secuencia de pasos para acceder a RETELAB a través de Shibboleth es la siguiente:

1. El usuario accede a RETELAB a través del portlet de acceso de Shibboleth.
2. El sistema muestra al usuario una lista con los centros asociados para que seleccione a cuál pertenece.
3. El sistema redirige al usuario al sistema de autenticación correspondiente.
4. Una vez que el sistema externo autentica al usuario, éste es redirigido nuevamente al portal de RETELAB junto con sus datos personales (nombre, rol, etc.).
5. RETELAB utiliza la información recibida para generar las credenciales del usuario necesarias, incluyendo el AC, para utilizar el sistema.

Para la integración de los DCs y ACs en el procedimiento de SSO se actuó, principalmente, en la clase *GridSphereServlet*. Esta clase fue previamente modificada en el proyecto MAMS para crear usuarios Shibboleth en el portal GridSphere y, en el caso de RETELAB, se incorporaron los métodos necesarios para la gestión de los DCs y ACs. El código para la creación y almacenamiento del los DCs asociados a los usuarios de Shibboleth se muestra en el Código A.4, mientras que el procedimiento para la creación de los ACs se presenta en el Código A.5.

La inclusión de Shibboleth en el sistema de RETELAB permite que los socios del proyecto que deseen integrarse en una Red de confianza puedan autenticarse a través de sus sistemas internos sin necesidad de registrarse en el portal.

3.3. Sistema de almacenamiento distribuido

El sistema de almacenamiento de RETELAB [75] [80] fue diseñado para gestionar grandes colecciones de datos provenientes de la teledetección y de los propios proyectos oceanográficos. En RETELAB, el concepto de dato debe entenderse como un fichero binario que representa un producto obtenido a través de un sensor o generado a través de un proceso de análisis de datos previos. Este tipo de productos requiere un considerable espacio en disco, por lo que su almacenamiento masivo es una tarea complicada que no puede ser resuelta utilizando DBs tradicionales. La computación Grid puede afrontar este reto a través de un Data Grid, es decir, un sistema de almacenamiento distribuido entre los diferentes recursos del Grid. RETELAB no pretende ser sólo una herramienta para el almacenamiento y procesamiento, sino que busca ser un medio para compartir conocimiento entre la comunidad oceanográfica; por ello, debe proporcionar procedimientos a los usuarios que, de forma sencilla, permitan publicar y compartir los resultados de sus investigaciones. Estos procedimientos deberían estar basados en el uso de metadatos para describir los productos almacenados y así facilitar los procesos de localización y distribución de los mismos.

Los metadatos son datos que describen el contenido de otros datos. Funcionan de forma similar a un índice para localizar objetos. A cada objeto se le asigna una «ficha» que lo describe a través de varios campos y que es utilizada para localizarlo dentro de un almacén. Un ejemplo clarificador sería el de las fichas de los libros en una biblioteca. En RETELAB, cada dato almacenado tiene asociado un conjunto de pares campo-valor que lo describen y que ayudan a realizar búsquedas más concretas a través de las colecciones de datos. Los metadatos

pueden estar almacenados de forma interna, incrustados dentro del propio dato al que describe (cabecera), o en un almacenamiento externo.

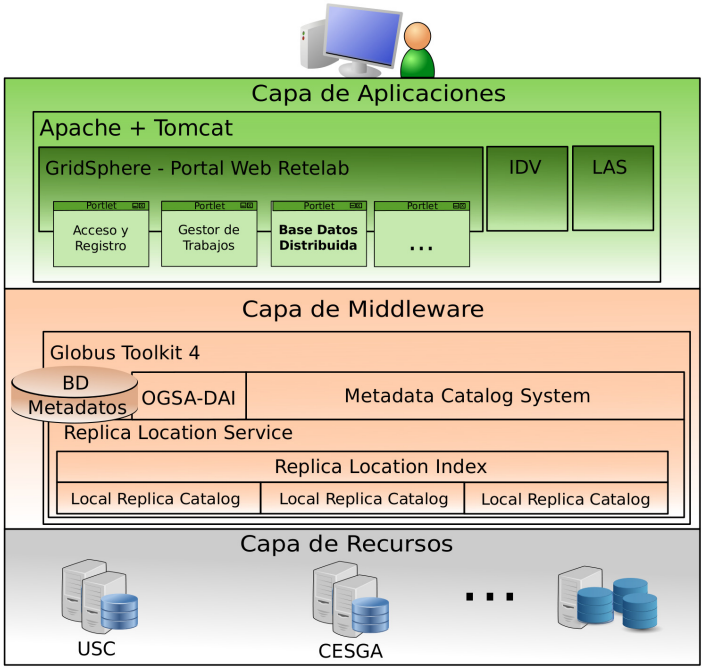

Figura 3.7: Arquitectura del Data Grid desplegado en RETELAB.

La arquitectura de la solución implementada, que puede verse en la Figura 3.7, está formada por tres capas: la capa inferior, que contiene a los recursos del Grid, almacena las colecciones de datos; la capa intermedia o *middleware* integra las diferentes tecnologías que forman el núcleo del sistema; y, por último, la capa superior proporciona la interfaz de usuario para acceder al sistema de almacenamiento.

Las tecnologías presentes en la capa de *middleware* son las encargadas de gestionar el almacenamiento de RETELAB. El núcleo del Data Grid lo forma el servicio de GT4 denominado *Replica Location Service* (RLS). Los elementos almacenados están identificados por un Nombre Lógico de Fichero (LFN) y cada uno de ellos tiene asociado 'N' Nombres Físicos de Ficheros (PFNs), es decir, las diferentes rutas donde está almacenado, ya que puede estar replicado. Por lo tanto, el servicio «conoce» la localización específica de cada dato dentro de RETELAB relacionando LFNs con PFNs. El RLS funciona gracias a dos subsistemas, el

Local Replica Catalog (LRC), y el *Replica Location Index* (RLI). Por un lado, cada recurso del Data Grid de RETELAB tiene desplegado un LRC con un listado que relaciona los LFNs con los PFNs almacenados en ese recurso y, por otro lado, RETELAB posee un RLI que, de forma centralizada, relaciona los LFNs con los diferentes LRCs donde están almacenados. Es tarea de cada LRC informar al RLI de las colecciones de datos que gestiona.

Para describir correctamente los datos es necesario disponer de un conjunto de metadatos que permita reflejar todas sus características y particularidades. RETELAB ha utilizado el estándar para metadatos geoespaciales ISO 19115 [7]. Este estándar proporciona un conjunto de metadatos orientado a describir datos geoespaciales aportando información sobre la identificación, la extensión, la calidad, el modelo espacial y temporal, la referencia espacial y la distribución de dichos datos.

El almacenamiento y gestión de los metadatos se hizo de forma externa al propio dato (la alternativa sería agregarlos en la propia cabecera) y centralizada empleando el *Metadata Catalog System* (MCS) [100]. El MCS es un servicio del GT4 que proporciona un catálogo de metadatos a través de una interfaz que cumple con el estándar de OGSA. Este catálogo permite describir datos a través de metainformación. OGSA-DAI [17], que proporciona una interfaz OGSA para el acceso a ficheros XML y DBs relacionales, puede integrarse con el MCS para proporcionarle la capacidad de almacenamiento de los metadatos, así como los mecanismos de autenticación propios de un Grid.

El sistema de almacenamiento fue integrado en el portal Grid para que el usuario pudiese utilizarlo a través de los diferentes portlets desplegados. El proceso de acceso y recuperación de un archivo almacenado se resume en la Figura 3.8 e incluye los siguientes pasos:

1. El usuario, utilizando metadatos, realiza una consulta a través del portlet correspondiente.

2. El sistema consulta el MCS para obtener los nombres lógicos cuyos metadatos se ajustan a la consulta de usuario.

3. El sistema consulta al RLI para saber qué LRCs gestionan los datos relacionados con los alias obtenidos.

4. El sistema consulta a los LRCs las direcciones físicas vinculadas a los LFN.

5. A través de la dirección física, el sistema puede mover los datos utilizando, por ejemplo, el servicio GridFTP [13]. El GridFTP, que es parte del GT4, se basa en el protocolo FTP para proporcionar transferencias de datos seguras, robustas, rápidas y eficientes.

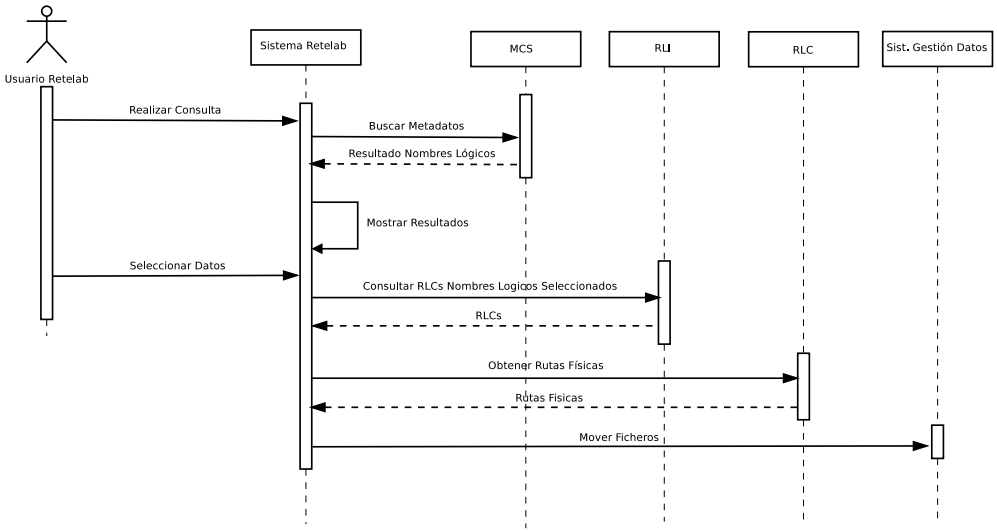

Figura 3.8: Secuencia resumida de acciones para buscar y acceder a los datos almacenados.

Los usuarios no sólo pueden buscar y utilizar las colecciones de datos del sistema, sino que pueden publicar los resultados de sus trabajos asignándoles una lista de metadatos que los describan. Además, cada usuario dispone de un espacio virtual propio donde puede descargar datos para editarlos y, cuando estén listos, subirlos y compartirlos con el resto de la comunidad. La integración del Data Grid con el sistema de envío de trabajos se detalla en la Sección 3.5.1.

Detalles de implementación Para facilitar la integración de la DB con los portlets, se desarrolló y empaquetó en un JAR un conjunto de clases que proporcionan las herramientas para la gestión de la DB virtual. Cada portlet que quisiese emplear la DB sólo necesitaba incorporar el JAR entre sus librerías. El paquete es descrito a través del diagrama de clases simplificado de la Figura 3.9. Las clases implementadas interactúan con las clases proporcionadas por el GT4.

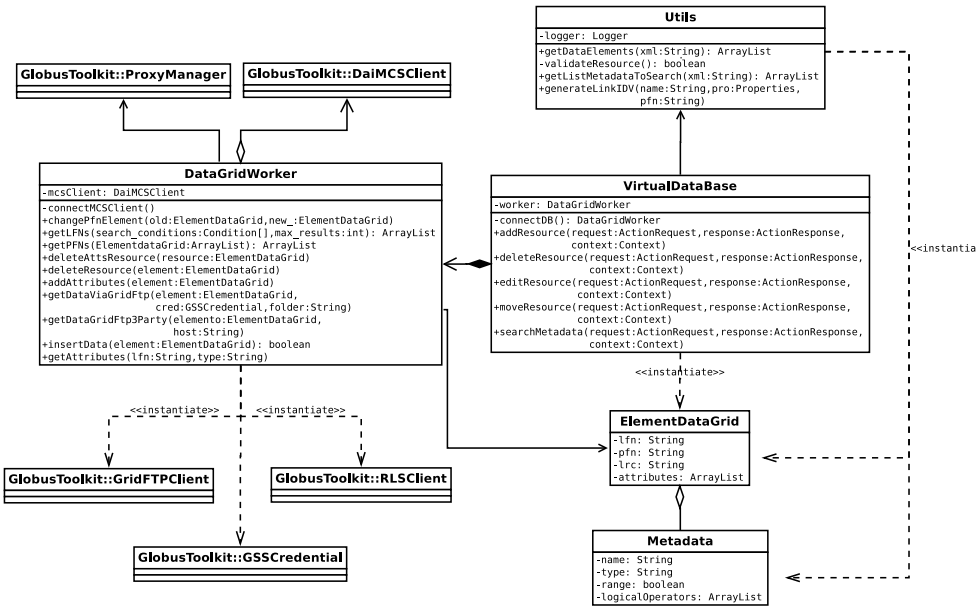

Figura 3.9: Diagrama de clases simplificado en el que se presentan las clases utilizadas para desarrollar las operaciones sobre la DB virtual.

El desarrollo de las clases para la gestión de la DB tuvo varias dificultades y, entre ellas, es interesante destacar dos [80]:

– Integración de OGSA-DAI y el MCS: GT proporciona librerías para gestionar el MCS; concretamente, la clase *DaiMCSClient* es una implementación de un cliente para conectarse al MCS sobre OGSA-DAI. Esta clase hace uso de otra llamada *GenericServiceFetcher* que se encarga de instanciar clases que implementan interfaces para comunicarse con los diferentes servicios de datos. *GenericServiceFetcher* se conecta con un servicio a través de una URL y recupera el WSDL que lo describe. El análisis del WSDL le permite determinar la distribución de OGSA-DAI empleada para implementarlo y así instanciar la interfaz adecuada. El código utilizado para recuperar el WSDL intenta realizar una conexión a una URL pero sin especificar la clase responsable para gestionar dicha conexión. Si la clase responsable no está definida de forma explícita, se asigna una en tiempo de ejecución que depende del protocolo usado para la conexión. Cuando *GenericServiceFetcher* forma parte de un software de escritorio funciona correctamente

pero en un entorno J2EE, debido a la forma de cargar las clases en memoria que usa el Tomcat, no encuentra la clase responsable adecuada. Fue necesario modificar y hacer explícito la clase que debía cargar en la conexión modificando la clase *GenericServiceFetcher* (véase el Código A.6).

– Gestión de *proxies*: cuando un usuario se autentica en el portal, el sistema obtiene automáticamente un *proxy* de su DC del repositorio MyProxy. El GT proporciona librerías para gestionar dicho *proxy* y definirlo como el DC por defecto para que sea empleado en las autenticaciones. El problema surge con el servicio RLS que necesita que el DC esté almacenado físicamente en un directorio local. Para solucionarlo se desarrolló un procedimiento de conexión/validación con la DB que fijaba el DC por defecto para que fuese utilizado por los servicios que lo requiriesen y, además, fuera almacenado local y temporalmente para que el RLS pudiese usarlo en la autenticación. El procedimiento empleado se detalla en el Código A.7.

3.3.1. Almacenamiento de datos

Se han desarrollado diferentes procedimientos para introducir datos en el sistema de almacenamiento de RETELAB que cubren las diferentes necesidades de sus usuarios.

Almacenamiento de datos locales Se ha habilitado un sistema de transferencia y almacenamiento de datos locales (véase la Figura 3.10) que permite a los usuarios compartir datos con la comunidad. Para transferir los datos, el usuario debe seleccionar el producto de su computadora que desea compartir, asignarle un LFN y describirlo a través de metadatos escogidos a través de una lista confeccionada a partir del estándar ISO 19115.

Almacenamiento de datos vía FTP En el campo de la observación de la Tierra, es habitual que los usuarios utilicen datos almacenados en servidores FTP externos para sus proyectos. Se ha implementado un portlet (véase la Figura 3.11) que permite el acceso a dichos servidores para copiar y transferir datos a RETELAB. El usuario debe introducir sus datos de acceso, navegar a través del árbol de directorios del servidor FTP, seleccionar los elementos que quiere añadir y describirlos a través de metadatos. Si los datos están comprimidos puede establecerse el tipo de compresión para que éstos sean desempaquetados automáticamente antes de almacenarse.

Figura 3.10: Portlet para el almacenamiento de datos locales en RETELAB.

Almacenamiento de datos generados en RETELAB Los datos generados tras la ejecución de una tarea en RETELAB pueden compartirse con el resto de la comunidad a través del sistema de almacenamiento (véase la Figura 3.12). El usuario puede visualizar los datos de salida y seleccionar los que desee añadir al Data Grid y, al igual que en el resto de los casos, podrá ir añadiendo los metadatos que considere necesarios para describir cada elemento.

Inicialización del Data Grid La introducción de nuevos elementos en el sistema de almacenamiento se hace, por defecto, de uno en uno a través de un portlet dedicado pero esto no es asumible en el momento de desplegar el sistema, ya que es necesario dar de alta grandes colecciones de datos. Se desarrolló un sistema de registro automático de datos a través de ficheros XML que indicaban rutas, alias y metadatos para cada elemento. Este procedimiento es de uso exclusivo del administrador y permite dar de alta automáticamente grandes volúmenes de datos.

Con el doble propósito de probar el proceso para la inicialización del Data Grid y el de proporcionar suficientes recursos para poder ejecutar los proyectos que validarían el laboratorio (véase la Sección 3.6), se dieron de alta varias colecciones de datos:

– Una colección de imágenes SAR correspondientes, por un lado, al Golfo de México y, por otro, a la costa noroeste de la Península Ibérica, así como los datos de viento

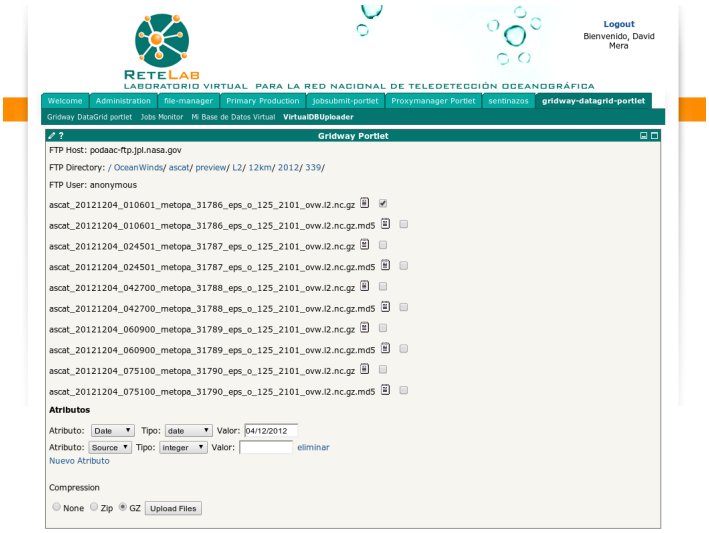

Figura 3.11: Portlet para la integración de datos externos en el sistema.

asociados a éstas en formato *Network Common Data Form* (NetCDF). Estos productos fueron empleados para desarrollar y validar el proyecto de SENTINAZOS (véase la Sección 3.6.3).

– Una colección de datos en NetCDF empleada para desarrollar la aplicación para el cálculo de la PP (véase la Sección 3.6.3) que incluía:

  - Datos de temperatura superficial del mar provenientes de los sensores AVHRR y MODIS.
  - Datos sobre la concentración de clorofila provenientes de los sensores MODIS y SeaWIFS.
  - Datos sobre la radiación solar disponible para el cálculo de la fotosíntesis estimados por la NASA.

Cada producto almacenado fue etiquetado con un conjunto de metadatos para describirlo y así facilitar su posterior identificación y búsqueda.

El formato NetCDF [93] ha sido ampliamente utilizado en RETELAB debido a su popularidad dentro de la comunidad científica. Es utilizado regularmente en aplicaciones relacio-

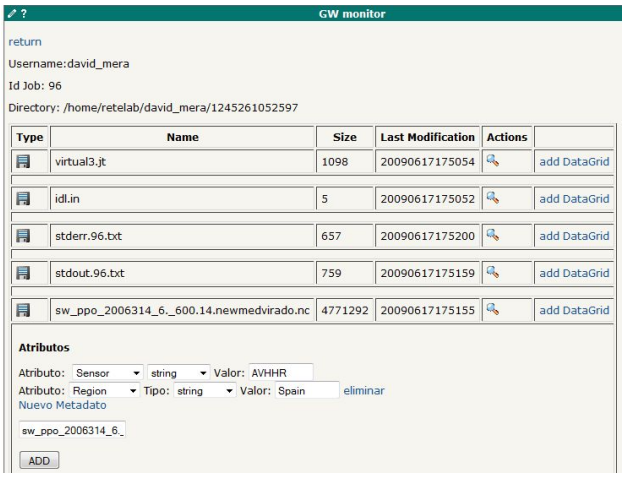

Figura 3.12: Resultados de la ejecución de un trabajo en el Grid. El usuario puede visualizarlos y revisarlos y/o añadirlos al Data Grid previa descripción usando metadatos.

nadas con la climatología, meteorología o la oceanografía para el intercambio de datos, es decir, como formato de entrada/salida.

3.4. Visualización de los datos

Se han integrado dos tipos de visores en RETELAB para la visualización de los datos, bien para validar las búsquedas, bien para el análisis de los resultados tras ser procesados.

3.4.1. Live Access Server

Un *Live Access Server* (LAS) [101] es un servidor con la capacidad de conectarse a servidores externos OPeNDAP [32] para el acceso e intercambio de datos. Está diseñado para proporcionar acceso a datos científicos georreferenciados y trabajar con información proveniente de distintos servidores como si fuese una única DB.

Un LAS permite generar gráficos de forma dinámica, solicitar fragmentos de datos en una variedad de formatos, comparar variables almacenadas en DBs diferentes y generar estadísticas básicas. La herramienta de visualización por defecto es Ferret, software de código abierto especialmente dirigido a trabajar con datos georreferenciados y que permite crear datos cien-

tíficos y proporciona una poderosa herramienta de análisis. Una de las grandes ventajas de un LAS es que cualquier usuario puede acceder a través de un simple navegador.

Se desplegó un LAS en RETELAB y se integró dentro del portal a través de un portlet con un Iframe, elemento HTML que permite empotrar un documento HTML dentro de otro; en este caso, se utilizó para incrustar la interfaz Web del LAS como puede verse en la Figura 3.13.

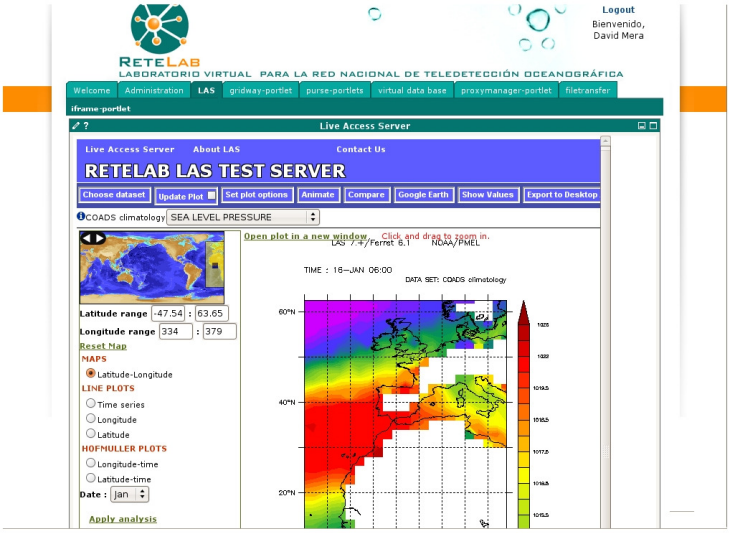

Figura 3.13: Integración del Live Access Server en RETELAB.

3.4.2. Integrated Data Viewer

El *Integrated Data Viewer* (IDV) [84], desarrollado por Unidata, es una herramienta software implementada en Java que permite visualizar y analizar datos geocientíficos. Para integrar IDV en el sistema se desarrolló un procedimiento que, por cada elemento recuperado tras una consulta a la DB o por cada resultado producido por la ejecución de un trabajo, genera dinámicamente un enlace a un fichero *Java Networking Launching Protocol* (JNLP). La descarga, por medio del enlace, de este fichero inicia, previa confirmación del usuario, la instalación temporal del software IDV. La instalación se realiza de forma automática y transparente gracias a JavaWS y, una vez finalizada, IDV es cargado con los datos asociados al enlace. El único requisito para su utilización es disponer del Entorno de Ejecución de Java (JRE) que incluye al propio JavaWS.

La Figura 3.14 muestra una composición de cómo un usuario podría acceder a un fichero NetCDF almacenado en su directorio virtual y visualizarlo a través del IDV de forma automática.

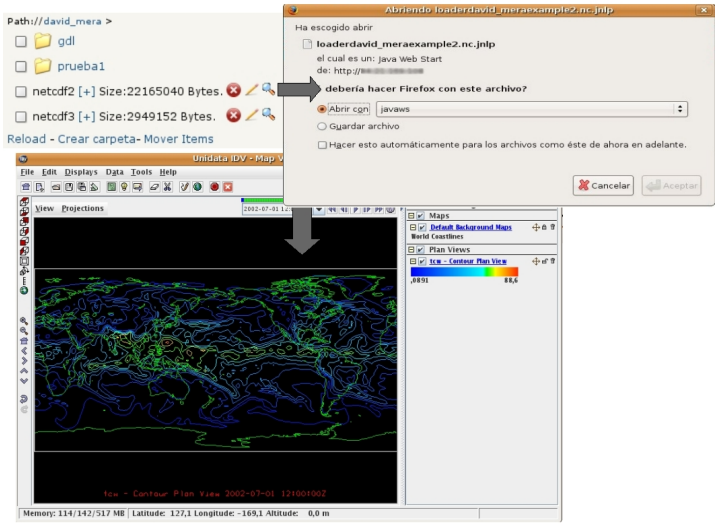

Figura 3.14: Integración del IDV en RETELAB.

3.5. Gestión y monitorización de trabajos

A pesar de que el CESGA fue el principal desarrollador del módulo de envío y monitorización de trabajos, su descripción en el documento es necesaria debido a que es una parte fundamental de RETELAB y se ha colaborado activamente en su integración con el resto de los módulos del sistema.

Existen diferentes alternativas que permiten a los usuarios enviar trabajos a un sistema Grid pero requieren la interacción del usuario para, por ejemplo, decidir a qué recurso del Grid se envían las tareas, lo que implica conocer su arquitectura. La interacción con el usuario resta transparencia y complica el proceso.

Como se ha mencionado en varias ocasiones, RETELAB ha sido diseñado teniendo como premisa la facilidad de uso, por lo que el sistema de envío y monitorización de trabajos fue desarrollado para que permitiese a los usuarios configurar, ejecutar, monitorizar y recuperar

los resultados sin necesidad de conocer detalles de implementación del sistema Grid. Este nivel de abstracción se consiguió, principalmente, gracias a la integración de un metaplanificador que optimiza el uso de los recursos compartidos y realiza tareas para el descubrimiento de recursos, planificación de tareas, ejecución, monitorización y recuperación de trabajos.

RETELAB incorporó un portlet para el envío y monitorización de trabajos que interactúa con el metaplanificador GridWay [64] [65] a través del API de *Distributed Resource Management Application* (DRMAA) [112]. La elección de Gridway como metaplanificador se basó en varias de sus características: permite interoperar con diferentes *middleware* Grid, tiene soporte para DRMAA y permite enviar trabajos empleando el estándar *Job Submission Description Language* (JSDL) [16]. Un esquema resumido de la arquitectura utilizada para el envío y monitorización de trabajos puede verse en la Figura 3.15.

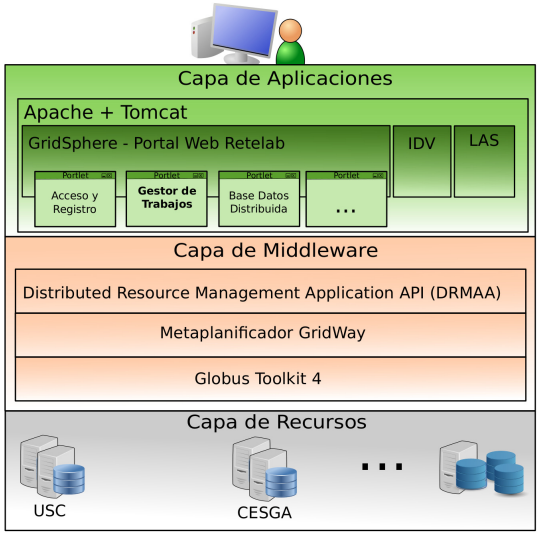

Figura 3.15: Esquema de la arquitectura empleada para el envío y monitorización de trabajos en RETELAB.

Detalles de implementación La integración del metaplanificador con el portal Web fue bastante laboriosa, ya que Gridway fue originalmente concebido para ser empleado por usuarios Unix a través de CLI o de aplicaciones DRMAA. Tal y como estaba originalmente diseñado, GridWay utilizaba el mismo proceso de Unix y el mismo *proxy* para todos los trabajos que se lanzaban desde la misma cuenta de usuario. La identificación y el acceso de usuarios

de RETELAB no se basa en cuentas de Unix, sino que son identificados por su *proxy* y su AC (véase la Sección 3.2). Todos los usuarios de RETELAB utilizan la misma cuenta de Unix a nivel de recurso de Grid, lo que lo hace incompatible con el diseño de GridWay. Fue necesario implementar, junto con el equipo de desarrollo del GridWay, una versión mejorada del metaplanificador [33] que, manteniendo la compatibilidad con el estándar DRMAA, permitiese identificar a los usuarios, no sólo por su cuenta Unix, sino también por el identificador presente en el DC X.509.

3.5.1. Integración con el sistema de almacenamiento distribuido

El portlet para el envío de trabajos está totalmente integrado con el Data Grid, lo que permite, por un lado, utilizar los datos almacenados en RETELAB como parámetros de entrada en los trabajos de usuario y, por otro, publicar y compartir los resultados de los trabajos con la comunidad.

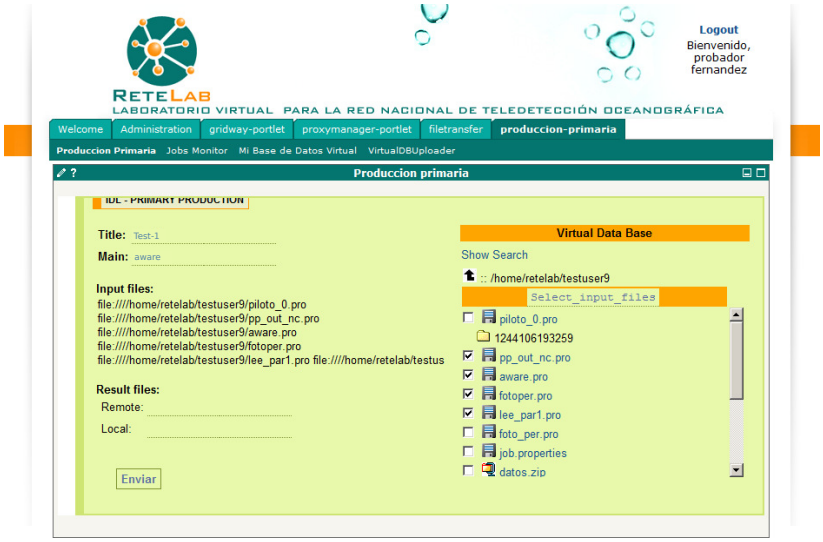

Figura 3.16: Integración del espacio de almacenamiento privado del usuario con el portlet de envío de trabajos.

Los parámetros de entrada para un trabajo pueden ser añadidos desde el directorio privado de un usuario, como puede verse en la Figura 3.16, o desde el Data Grid.

En el caso de que el usuario quiera utilizar los datos almacenados en el Data Grid, deberá hacer uso de la búsqueda dirigida por metadatos. Los resultados que obtenga podrán ser visua-

lizados a través del IDV y así confirmar su idoneidad antes de incorporarlos a su trabajo. La Figura 3.17 muestra la secuencia de búsqueda e incorporación de datos del Data Grid como parámetros de un trabajo.

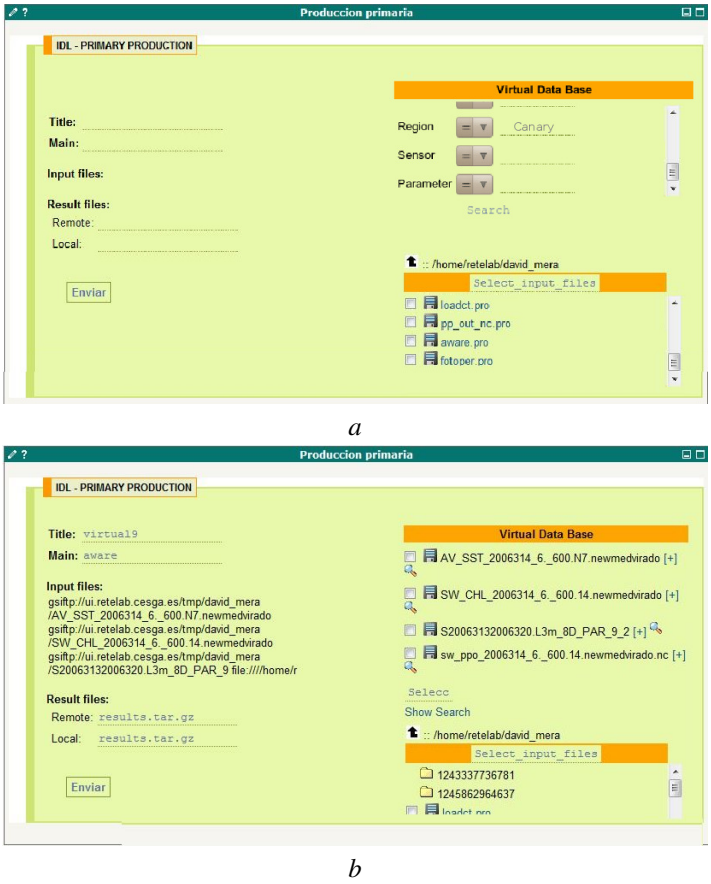

Figura 3.17: Integración del Data Grid con el portlet de envío de trabajos. a) Portlet de envío de trabajos con acceso a la búsqueda a través de la DB virtual. b)Resultados de la búsqueda y selección de los parámetros para el trabajo.

Una vez lanzado el trabajo al Grid, RETELAB pone a disposición del usuario un portlet de monitorización de tareas (véase la Figura 3.18) en el que puede verse el estado (espera, ejecución, error o finalizado) de todos sus trabajos. Una vez que el trabajo ha finalizado, el

monitor permite acceder a los resultados del mismo, visualizarlos e incluso integrarlos en el Data Grid (véase la Figura 3.12), en caso de que el usuario quiera compartirlos.

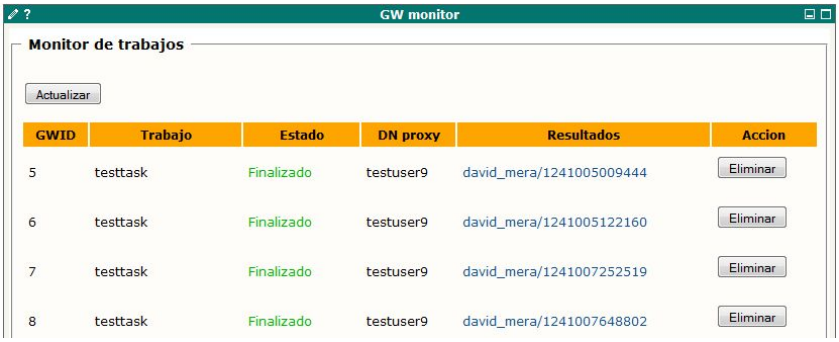

Figura 3.18: Portlet para la monitorización de los trabajos de usuario.

3.6. Validación del laboratorio virtual

Una vez finalizado el desarrollo del laboratorio virtual, fue probado desplegando varias aplicaciones desarrolladas en entornos de investigación en el área de la oceanografía. La integración de estas aplicaciones sirvió para validar el sistema y para proporcionar software especializado a la comunidad. Todas las aplicaciones desplegadas están completamente integradas en el portal, por lo que hacen uso del sistema de autenticación, del Data Grid y de la gestión y monitorización de trabajos.

De entre las aplicaciones que a continuación se van a describir, el *Regional Ocean Model System* (ROMS) y el modelo de producción primaria son aplicaciones desarrolladas por terceros que fueron adaptadas para ser ejecutadas en RETELAB, mientras que el proyecto SENTINAZOS es un desarrollo propio concebido para solucionar un problema concreto que, por su entidad, alcance y necesidad de investigación adicional del estado del arte, es descrito en detalle en la segunda parte del presente documento.

3.6.1. ROMS

ROMS [98] es un modelo de circulación oceánica de nueva generación especialmente diseñado para simular de manera precisa sistemas oceánicos regionales. La Figura 3.19 muestra

un ejemplo de una simulación de la temperatura superficial oceánica en la costa noroeste de la península Ibérica.

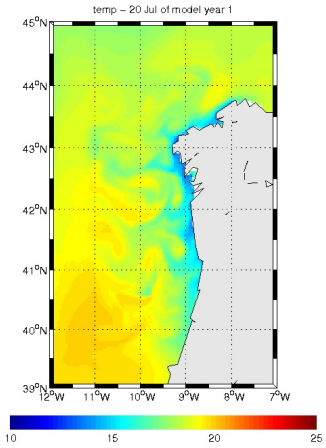

Figura 3.19: Representación de la temperatura superficial en un instante de una simulación empleando ROMS.

ROMS fue desarrollado en Fortran 90/95 e implementa, además del método en serie, dos métodos de paralelización [114], *Message Passing Interface* (MPI) y *Open Multiprocessing* (OpenMP), mutuamente excluyentes. Una característica del modelo es que todas sus entradas y salidas se basan en el formato NetCDF, lo que facilita compartir los datos entre los usuarios y otras aplicaciones.

El portlet que integra ROMS en el laboratorio fue desarrollado junto con el CESGA y permite definir y ejecutar instancias del modelo. Los usuarios, a través de la interfaz Web, pueden configurar los parámetros de las simulaciones, seleccionar los datos de entrada a través del Data Grid, monitorizar el estado de sus trabajos y, cuando estén finalizados, recuperar los resultados. En esta versión, es el usuario, en el momento de la configuración, quien debe seleccionar el modo de ejecución de la aplicación, ya sea en serie o en paralelo vía OpenMP o MPI.

3.6.2. Cálculo del modelo de producción primaria

RETELAB debía poder integrar aplicaciones realizadas por los propios centros asociados al proyecto, por lo que se adaptó, a través de un portlet, un modelo realizado por el ICCM para el cálculo de la Producción Primaria (PP) oceánica.

La PP puede definirse como la generación de componentes orgánicos a partir del CO_2 que realizan los organismos autótrofos a través de los procesos de fotosíntesis o quimiosíntesis. Es un mecanismo clave que permite a los océanos, a través del fitoplancton, ser los pulmones del planeta, ya que gracias a la PP se genera cerca del 50 % de la producción global de oxígeno y se captura alrededor de la mitad del CO_2 mundial. El cálculo de este parámetro es fundamental para comprender el papel que juega el ciclo del carbono en el sistema climático mundial.

Los llamados modelos bioópticos son los métodos más utilizados para el cálculo a gran escala de la PP. Estos modelos se basan en información obtenida a través de satélites que incluye, entre otros datos, información sobre las biomasas de fitoplancton y de la luminosidad oceánica. En concreto, el modelo implementado para RETELAB es una adaptación del Vertically Generalised Production Model (VGPM) desarrollado por Berhefeld y Falkowsky [20].

El portlet implementado permite definir los parámetros de entrada al modelo y ejecutarlo en el sistema Grid. Los parámetros esperados para un correcto funcionamiento son: la temperatura superficial oceánica, la concentración de clorofila y los datos sobre la radiación disponible para fotosíntesis. La Figura 3.20 muestra una composición que ilustra la visualización de resultados del cálculo de PP una vez lanzada la tarea en el sistema Grid.

3.6.3. SENTINAZOS

Después de haber integrado y adaptado aplicaciones desarrolladas por terceros, se afrontó el reto de implementar una desde cero que aprovechase el sistema y, además, fuese útil para la comunidad en un problema de difícil solución como es la detección semiautomática de contaminantes marinos basados en hidrocarburos. La contaminación marina es un problema que habitualmente afecta a nuestras costas y que, a menudo, se relaciona con grandes catástrofes de petroleros. Sin embargo, en un mayor porcentaje, es originada por pequeños derrames que, bien sea de forma fortuita debido a pequeños accidentes o bien sea de forma premeditada a causa de tareas de mantenimiento, se producen diariamente. Los centros de vigilancia marítima invierten una gran cantidad de recursos en la vigilancia y persecución de estos actos pero cubrir toda la superficie costera es una tarea inabarcable. Los satélites aportan la cober-

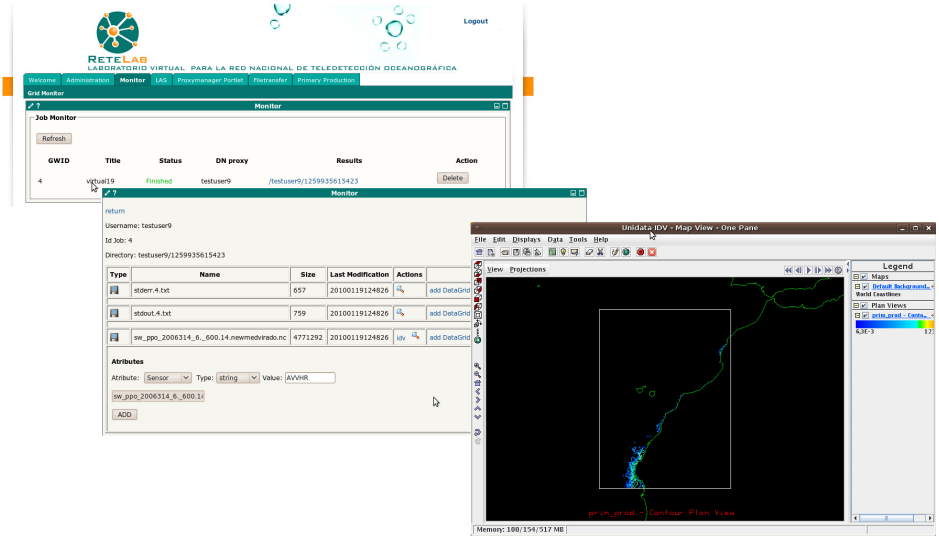

Figura 3.20: Composición de la monitorización de un trabajo y su integración con el Data Grid y el sistema de visualización de IDV.

tura necesaria para el estudio y monitorización de grandes áreas y, concretamente, estudios previos [25] [110] demostraron la viabilidad del uso de las imágenes SAR de satélites en la detección de este tipo de contaminantes.

El objetivo es desarrollar una aplicación que permita el análisis de imágenes de SAR para detectar hidrocarburos e integrarla en RETELAB, ya que esto proporcionaría ciertas ventajas como:

- Distribuir el almacenamiento de las imágenes SAR.
- Compartir las colecciones de imágenes que cada centro disponga con el resto de participantes del proyecto.
- Optimizar los algoritmos, ya que tienen una gran carga computacional que, en gran medida, está asociada a los bucles que las analizan y que son candidatos a ser paralelizados.
- Ejecutar diferentes instancias de la aplicación al mismo tiempo aprovechando los diferentes recursos del sistema y así analizar simultáneamente diferentes áreas de la costa.

Aunque la bibliografía muestra que existen algoritmos similares para la detección de vertidos, éstos suelen presentar uno o varios de los siguientes problemas:

- Requieren la presencia activa de un operador.
- El coste computacional es demasiado alto para su uso en tiempo real.
- Las implementaciones se basan en librerías comerciales con licencias privativas.
- No tienen en cuenta los efectos de los fenómenos meteorológicos, principalmente el viento, en la detección de los vertidos.

Debido a la complejidad de los algoritmos desarrollados, la segunda parte de la tesis está dedicada a describirlos en detalle y resaltar el carácter innovador de los mismos.

Parte II

Validación del Laboratorio Virtual: avances en la detección automática de vertidos de hidrocarburos

Capítulo 4

Estado del arte

Introducción

Aunque existen muchos tipos de contaminantes, los vertidos de hidrocarburos son, probablemente, los que más afectan al ecosistema marino. A pesar de lo que popularmente es aceptado, las grandes catástrofes provocadas por los petroleros suponen una pequeña parte de dicha contaminación. Un estudio de la Agencia Espacial Europea (ESA) [39] reveló que tan sólo un 7 % de los vertidos pueden ser atribuidos a accidentes, mientras que un 45 % corresponden a derrames que habitualmente se realizan durante operaciones de mantenimiento (por ejemplo la limpieza de las sentinas) y que denominaremos sentinazos, un 13 % proviene de fuentes naturales y el 35 % restante corresponde a vertidos realizados directamente en los ríos.

Los vertidos afectan a la economía (turismo, pesca, etc.) y al ecosistema de las regiones damnificadas, por lo que son necesarios sistemas y herramientas para su localización. Una rápida identificación permitiría minimizar su impacto e incluso identificar a sus causantes. Los sensores que operan en el rango de las microondas, comúnmente llamados radares, han demostrado ser de gran utilidad en la detección de vertidos de hidrocarburos.

Radar El radar es un sistema de detección activo [31] cuyo funcionamiento básicamente consiste en el envío de un pulso de energía en el rango de las microondas hacia la superficie terrestre. Este haz es reflejado y retrodispersado por la superficie y parte de la energía es recogida nuevamente por la antena del radar. La intensidad de la señal que vuelve a la antena es medida y grabada para posteriormente ser utilizada en la construcción de una imagen de la

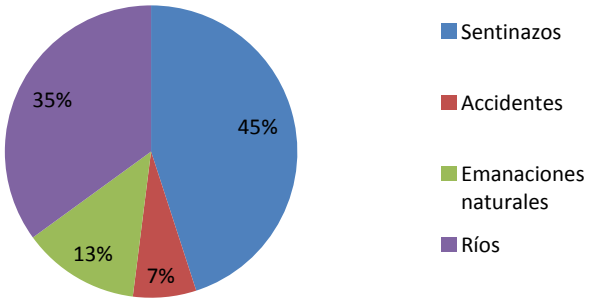

Figura 4.1: Orígenes de los vertidos de hidrocarburos. Imagen adaptada de [39]

zona estudiada. Cada píxel de la imagen está asociado a un área sobre el terreno y almacena un valor que representa la cantidad de energía que regresa al sensor debido a la retrodispersión.

Tradicionalmente los buques y los aviones han sido los medios de vigilancia más utilizados en la lucha contra la contaminación oceánica. Por un lado, los buques equipados con sistemas de radar son útiles para detectar contaminantes en sus proximidades y obtener muestras de los vertidos pero su restringida cobertura limita su eficacia; por otro lado, los aviones de vigilancia que utilizan sistemas de Radar Lateral Aerotransportado (SLAR) permiten vigilar áreas más extensas pero su coste hace que sea poco viable emplearlos para monitorizar superficies de gran tamaño de forma intensiva y simultánea. En resumidas cuentas, tanto los buques como los aviones son adecuados como medios de primera intervención pero sería deseable un sistema más económico con mayor cobertura.

El uso de sistemas de radar instalados a bordo de satélites permite abarcar áreas mucho más extensas y a un coste razonable pero su resolución espacial no es adecuada para la búsqueda de pequeños derrames de hidrocarburos. La resolución es la habilidad para distinguir dos objetos separados por la mínima distancia posible. Si la distancia entre los objetos es suficiente, cada uno de ellos estará localizado en una celda de la imagen diferente y serán distinguibles; en caso contrario, el radar devolverá una combinación de la energía reflejada por los dos objetos como si fuese uno sólo [74]. Tanto el radar aerotransportado como el espacial observan la superficie lateralmente (véase la Figura 4.2) y su resolución espacial es diferente

en la dirección paralela a la trayectoria del satélite, también denominada resolución en acimut, y en la perpendicular, es decir, en la dirección al objetivo.

En la perpendicular a la trayectoria del satélite, el sistema puede discriminar dos objetos si la distancia entre ellos es mayor que la mitad de la anchura del pulso (*pulse width*) enviado. La resolución perpendicular se calcula a través de la siguiente fórmula:

$$R_p = \frac{c \cdot \tau}{2} = \frac{c}{2 \cdot \beta} \tag{4.1}$$

, siendo c la velocidad de la luz y τ la anchura del pulso. También puede expresarse a través el ancho de banda del pulso (*pulse bandwidth*) representado por β.

Normalmente, los radares a bordo de satélites trabajan en anchos de banda que van desde los 10 a los 40 MHz. Este rango generaría resoluciones que van desde los 15 a los 3,7 metros, que son suficientemente buenas para la detección de vertidos. El problema principal está asociado a la resolución en acimut, es decir, en la trayectoria del satélite, que se calcula como:

$$R_{ac} = \frac{\lambda \cdot d}{L_A} \tag{4.2}$$

, siendo λ la longitud de onda, d la distancia del sensor al objetivo y L_A la longitud de la antena.

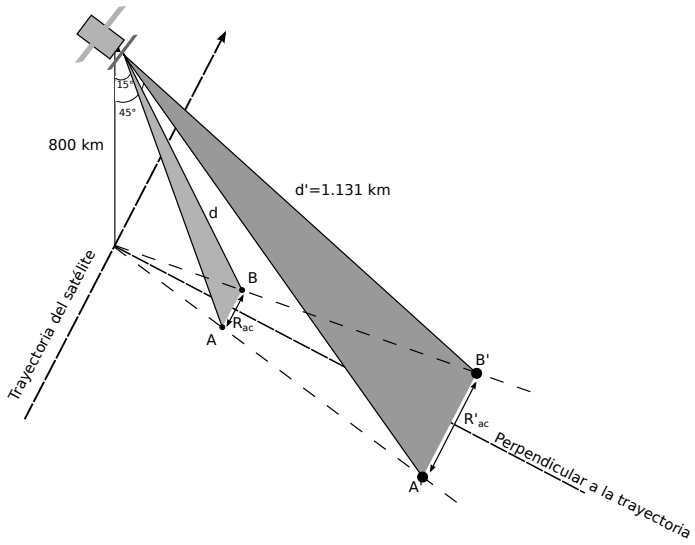

Figura 4.2: Resolución espacial de un radar a bordo de un satélite.

Dado que la longitud de onda está situada dentro de un rango determinado de microondas y que no puede haber grandes variaciones de la altura a la que orbita el satélite, la resolución en acimut queda directamente determinada por la longitud de la antena. Obtener una óptima resolución utilizando sistemas de radar sobre plataformas espaciales implicaría el uso de antenas de enormes proporciones, lo que sería impracticable. Empleando los datos de ejemplo de la Figura 4.2, un radar que operase en la Banda C con una longitud de onda de 5 cm y que viajase a bordo de un satélite que orbitase a 800 km de altura, necesitaría una antena de casi 2 km para conseguir una resolución de 30 m cuando el Ángulo de Incidencia (IA) fuese de 45°.

$$L_A = \frac{0,05 \cdot 1131 \cdot 10^3}{30} = 1.885m \tag{4.3}$$

Para solucionar este hándicap, surge un tipo especial de radar llamado SAR que permite obtener resoluciones de pocos metros con antenas de pequeñas dimensiones. Por ejemplo, la resolución del Envisat alcanzaba los 30 m empleando una antena rectangular de 1,3 m x 10 m.

SAR Los principios básicos del funcionamiento del SAR solucionan la limitación de resolución del radar. El SAR, montado sobre una plataforma móvil, analiza y recoge diferentes medidas de un mismo punto durante su trayecto (Figura 4.3). Esto permite simular una antena de las mismas proporciones que la distancia entre la primera y la última medición llegando a conseguir resoluciones de tan sólo unos pocos metros.

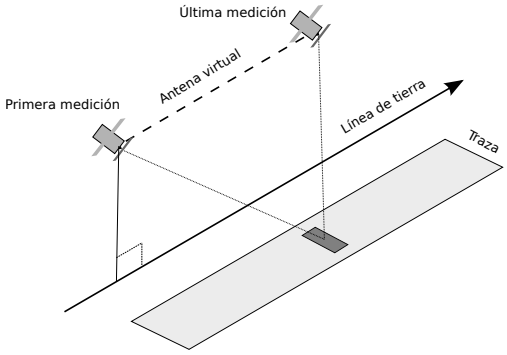

Figura 4.3: Antena virtual generada por SAR.

Las imágenes SAR son complejas y difíciles de interpretar y esto es debido a múltiples factores (la superficie es estudiada lateralmente, el ángulo de emisión e incidencia del haz de

energía varía, la señal de retorno se ve afectada por la distancia entre la antena y la superficie estudiada, etc.). La Figura 4.4 muestra un ejemplo de una imagen SAR en la que se aprecia cómo la intensidad no es homogénea. Las zonas más cercanas al satélite y con menor IA dan lugar a niveles de intensidad más altos (representados de forma más clara), mientras que las zonas más alejadas y con ángulos mayores son representadas de forma más oscura debido a los bajos niveles de intensidad.

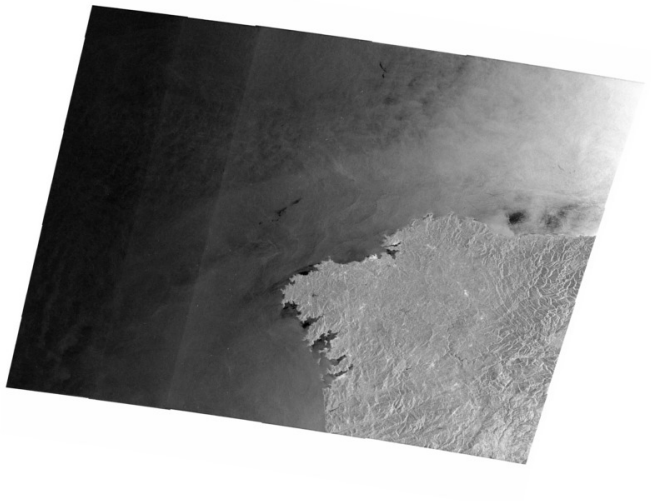

Figura 4.4: Imagen SAR de la costa gallega obtenida por el satélite Envisat (04/05/2007).

Además de la amplia cobertura y buena resolución que puede conseguirse con el SAR, es necesario destacar que, al ser un sensor activo, su funcionamiento no depende de la radiación solar y puede utilizarse durante el día y la noche. Además, el sensor es independiente de la cobertura nubosa, ya que su haz de energía en el rango de las microondas le permite atravesarla sin sufrir alteraciones.

Un estudio en mayor profundidad de SAR y de sus aplicaciones puede encontrarse en [66].

Imágenes SAR de la superficie oceánica

La superficie del mar, en condiciones moderadas de viento, presenta cierta rugosidad generada por las ondas capilares. Esta rugosidad provoca una retrodispersión uniforme del haz de energía enviado por el SAR. Los vertidos de hidrocarburos, así como determinados fe-

nómenos naturales [63] como las algas, el hielo o los vientos de baja intensidad, suprimen dichas ondas capilares provocando un aplanamiento de la zona (véase la Figura 4.5b). Estas áreas tienen un comportamiento especular frente al haz de energía que reduce considerablemente la cantidad de radiación retornada al sensor. Los píxeles que representan a este tipo de zonas almacenan valores de retrodispersión muy bajos, lo que provoca que se visualicen como áreas oscuras. El efecto provocado por los distintos fenómenos naturales y que es similar al de los vertidos es conocido como *look-alike*. Si el viento que afecta a la zona del vertido es suficientemente fuerte, el efecto de «aplanamiento» de los hidrocarburos es atenuado haciendo muy difícil o imposible distinguirlo del mar limpio (véase la Figura 4.5c). Por el contrario, si el área analizada está afectada por vientos de baja intensidad (véase la Figura 4.5a), las ondas capilares desaparecen y el contraste entre el vertido y el mar limpio no es suficiente para distinguirlos, por lo que aparecen *look-alikes*.

Diversos estudios [40] [92] [43] han fijado la velocidad del viento mínima y necesaria para detectar contaminantes entre 2 y 3 m/s. El límite superior se encuentra más difuso dado que depende del tipo de hidrocarburo y del tiempo que ha pasado desde su vertido pero podría establecerse que a partir de 10 m/s su detección es complicada y que una velocidad de entre 12 y 15 m/s podría ser considerada el umbral máximo para poder identificar un sentinazo. Las condiciones óptimas de viento se encontrarían en el rango de 3,5 a 7,0 m/s [54].

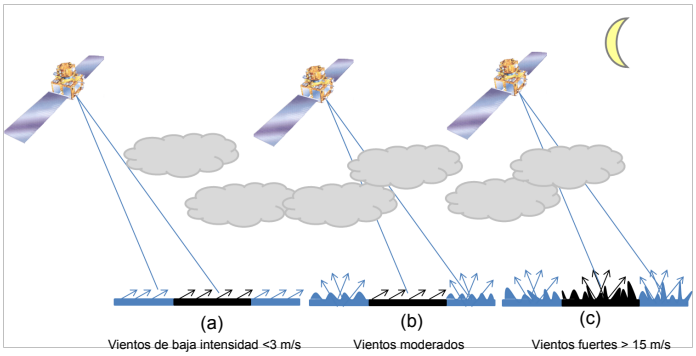

Figura 4.5: Efecto del viento en la retrodispersión de un pulso SAR.

Metodologías para la detección de vertidos de hidrocarburos en imágenes SAR

Inspección manual La inspección manual de las imágenes SAR para la detección de sentinazos es la técnica más popular y más sencilla de aplicar. Está basada en dos grandes fases:

- Localización de fenómenos sospechosos.
- Análisis y asignación de un grado de probabilidad (alto, medio o bajo) de que sean sentinazos.

Aunque la mayoría de los fenómenos detectados pueden ser clasificados correctamente por un experto basándose sólo en la imagen, existen algunos más confusos en los que el operador necesita utilizar información externa adicional para su correcta clasificación como puede ser: datos de vientos, periodo del año, forma, características del área de observación, etc.

Inspección automática La literatura muestra un gran esfuerzo en el desarrollo de sistemas automáticos y semiautomáticos para la detección de vertidos analizando imágenes SAR. Puede encontrarse un extenso repaso a los antecedentes bibliográficos tanto en el análisis de Brekke y Solberg [25] como en el de Topouzelis [110]. Dichos trabajos muestran que la mayoría de los sistemas, después de realizar algún tipo de preprocesado sobre las imágenes (calibrado, proyección, filtrado de ruido, enmascaramiento de la tierra, etc.), aplican un algoritmo compuesto de 3 grandes fases:

- Segmentación: el objetivo es localizar, aislar y extraer todas las áreas susceptibles de ser vertidos de hidrocarburos (incluyendo los falsos positivos).
- Extracción de características: las regiones previamente detectadas son analizadas y caracterizadas.
- Clasificación: analizando las características extraídas y a través de un clasificador, se asigna a cada candidato una etiqueta o una probabilidad de ser sentinazo o *look-alike*.

La extracción de características y la clasificación pueden estar apoyadas por información contextual como, por ejemplo, proximidad a fuentes de vertidos (plataformas petrolíferas, barcos, etc.) o datos meteorológicos. La Figura 4.6 resume las fases principales de un sistema automático.

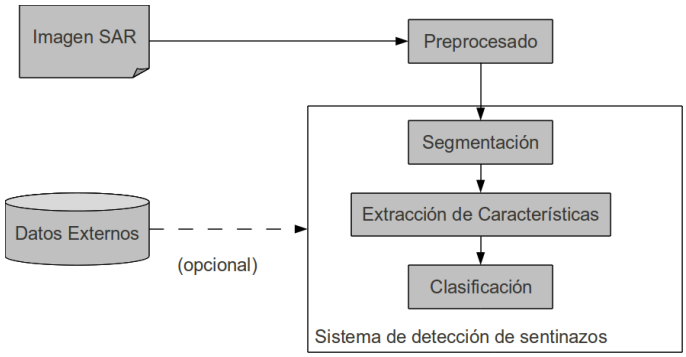

Figura 4.6: Sistema automático para la detección de sentinazos.

Comparativa entre la inspección automática y la inspección manual Los sistemas manuales presentan ciertas desventajas frente a los automáticos: su eficiencia depende de la experiencia del experto que analiza las imágenes, el ojo humano tiene dificultades para clasificar áreas difusas donde las fronteras entre la mancha y el fondo no están claras, el operador tiende a sobreseleccionar píxeles al delinear los bordes de las manchas, el tiempo de procesamiento es claramente superior al de un sistema automático, etc.

La principal desventaja de los sistemas automáticos respecto a los manuales tiene que ver con la confianza en los mismos. Los medios de vigilancia *in-situ* como barcos y aviones son costosos y limitados y, aunque los sistemas automáticos pueden clasificar correctamente la mayoría de los sentinazos, existe un pequeño número de ellos que son clasificados incorrectamente. Los sistemas de detección automáticos suelen utilizarse como una herramienta de apoyo a las decisiones donde un operario tiene la última palabra a la hora de movilizar los medios de vigilancia.

Técnicas de detección automática de sentinazos

Los siguientes subapartados profundizan en las técnicas utilizadas para desarrollar las principales fases de los algoritmos de detección.

Segmentación Analizando la intensidad de la señal recuperada por el sensor es posible distinguir ciertos fenómenos oceánicos. Como se ha visto anteriormente, los sentinazos, así como otros fenómenos naturales, se representan en las imágenes SAR con valores de baja in-

tensidad. El objetivo de esta fase es detectar todos los posibles sentinazos, siendo su resultado una imagen binaria en la que se destacan todos los candidatos. El umbral de intensidad para que un píxel sea segmentado no es constante y depende de factores asociados al área estudiada como el IA del haz del satélite o las condiciones meteorológicas. La segmentación es una fase fundamental dentro del algoritmo dado que, si un sentinazo no es aislado en esta fase, no podrá ser clasificado como tal en fases posteriores.

La diferencia de contraste entre los candidatos y el fondo de la imagen sugiere que las técnicas basadas en umbrales pueden ser adecuadas para la segmentación, por lo que, tanto las técnicas de umbralización simple [41] como las de umbralización adaptativa [103] [104] [37], han sido ampliamente utilizadas. Además de las basadas en umbrales, han sido empleadas otras técnicas para la segmentación como son: las redes neuronales [108] [54], las técnicas basadas en texturas [21] [72] o las basadas en detección de bordes [29] [81].

Debido a la complejidad de la segmentación y a las dimensiones de las imágenes SAR, ésta es la fase computacionalmente más costosa. La velocidad de ejecución del algoritmo es crucial para dar una respuesta en tiempo cuasi real. Para acelerar la segmentación algunos sistemas reducen la resolución radiométrica de la imagen original de 32 bits codificándola en 8 bits [90] [54] o bien analizan sólo pequeñas Regiones de Interés (ROIs) de la imagen principal que previamente han sido seleccionadas por un experto [37]; otros, en cambio, usan factores de escape en sus algoritmos para no procesar todos los píxeles de la imagen [104].

Extracción de características Una vez detectados los candidatos, éstos son analizados y caracterizados. El vector de características debería recoger la información necesaria para una posterior clasificación. A pesar de que existen algunos trabajos que intentan unificar el número y tipo de características [109] [111], en general no existen estudios sistemáticos sobre la extracción de características y su influencia en la clasificación y, por ello, su elección suele producirse de forma heurística. No obstante, la mayoría de las características utilizadas se engloban en las siguientes clases:

- Características geométricas: desde el punto de vista de la composición química del hidrocarburo, la forma no guarda una relación directa con el vertido; no obstante, desde el punto de vista de su origen, estudios previos [92] muestran que la forma de los sentinazos es característica y pertenece a uno de los tipos mostrados en la Figura 4.7.

- Características físicas: asociadas al nivel de intensidad de los píxeles que forman el candidato y a los de la superficie circundante.

– Texturas: en contraposición a las características obtenidas de cada píxel de forma aislada, las texturas aportan información sobre la correlación espacial de dichos píxeles en un área.

– Información contextual: la información exclusivamente basada en la mancha puede no ser suficiente para una correcta clasificación de las áreas más confusas y es en este punto donde la información contextual, como la distancia a una posible fuente del vertido [104] o los datos meteorológicos [40], puede ser de utilidad.

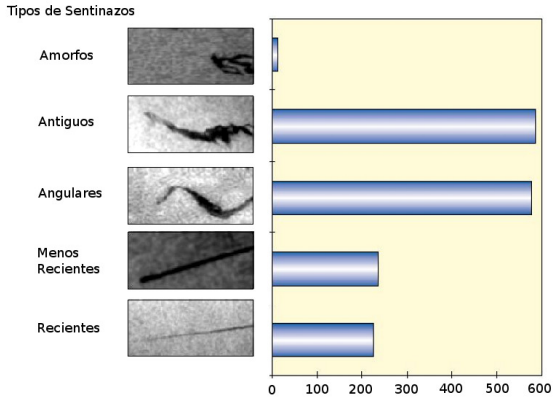

Figura 4.7: Análisis de la forma de 1.638 sentinazos detectados en el Mediterráneo durante el año 1999. Figura adaptada de [92].

Clasificación El objetivo de esta fase es distinguir cuáles de los candidatos segmentados son sentinazos y cuáles son *look-alikes* (falsos positivos), utilizando para ello un clasificador cuya entrada es el vector de características obtenido en la fase anterior. La comparación, en términos de eficiencia, entre los distintos clasificadores existentes en la bibliografía, resulta muy complicada dado que cada sistema utiliza su propio conjunto de pruebas y las características extraídas de los candidatos son diferentes. Es conveniente resaltar que muchos de los conjuntos de prueba utilizados no contienen sentinazos confirmados *in-situ*, sino que han sido detectados y clasificados por un operador de forma manual. En otros muchos casos simplemente no se especifica la forma en que fueron identificados.

En la bibliografía podemos encontrar múltiples aproximaciones al problema de la clasificación. Las Red Neuronal Artificial (ANN) [71] y sus diferentes implementaciones han sido muy utilizadas:

– Frate et ál. [37] utilizaron una DB con 600 imágenes ERS de baja resolución (100 m x 100 m) codificadas en 8 bits, de la que se extrajeron 139 candidatos (71 sentinazos y 68 *look-alikes*). Utilizando un clasificador basado en una ANN y a través de una validación cruzada (dejando uno fuera), se clasificó correctamente al 82 % de los sentinazos y al 90 % de los *look-alikes*.

– Topouzelis et ál. [108] parten de una DB de 24 imágenes de alta resolución (25 m x 25 m) codificadas en 8 bits. De la DB se seleccionaron 159 fragmentos (90 *look-alikes* y 69 sentinazos). La ANN utilizada clasificó correctamente el 91 % de los sentinazos y el 87 % de los *look-alikes*.

– Oscar García-Pineda et ál. [54] seleccionaron de una DB con 700 imágenes Radarsat codificadas en 8 bits, dos conjuntos de entrenamiento con 14 y 10 imágenes respectivamente y dos conjuntos de validación con 5 imágenes cada uno. Un primer clasificador, entrenado para distinguir los píxeles con hidrocarburos de los píxeles con mar limpio, fue aplicado sobre el conjunto de 14 imágenes obteniéndose una eficacia del 98,22 %. El segundo clasificador, entrenado sobre el conjunto de 10 imágenes para clasificar entre 3 tipos de píxeles (sentinazos, mar limpio y bajo viento), obtuvo una eficacia del 97,74 %.

Otro gran grupo de clasificadores son los basados en modelos probabilísticos:

– Solberg et ál. [103] combinaron un modelo estadístico con un sistema basado en reglas e incorporaron información contextual (datos de vientos y distancia a posibles orígenes del vertido). Aplicando el algoritmo sobre 84 imágenes SAR (inspeccionadas por un operador) y usando un método de validación cruzada, su eficacia alcanzó el 94 % en la detección de sentinazos y el 99 % en la de *look-alikes*. Este modelo ha sido mejorado en varias ocasiones:

  • Solberg y Brekke [104] trabajaron con un conjunto de entrenamiento consistente en 56 imágenes del Envisat y 71 imágenes del Radarsat y un conjunto de prueba con 27 imágenes del Envisat. El conjunto de prueba contenía 37 sentinazos confirmados y 12.110 *look-alikes* (obtenidos de la segmentación de las imágenes).

El 99,4 % de los *look-alikes* fueron clasificados correctamente, mientras que la eficacia en la detección de los sentinazos alcanzó el 78,4 %.

- Brekke y Solberg [26] se basaron en el modelo anterior para desarrollar un sistema de detección de sentinazos en dos pasos. En el primer paso emplearon un clasificador estadístico normalizado con el objetivo de detectar todos los sentinazos y, en un segundo paso, aplicaron un estimador de confianza que asignaba una probabilidad a cada sentinazo. Su DB se componía de 104 imágenes Envisat *Wide Swath Mode* (WSM) divididas en tres subconjuntos: entrenamiento (56 imágenes), validación (20 imágenes) y test (27 imágenes). La eficacia del clasificador fue del 92,7 % sobre los sentinazos y del 89,7 % con los *look-alikes*.

– Fiscella et ál. [41] realizaron una comparativa entre un clasificador de probabilidad compuesta y uno basado en la distancia de Mahalanobis. El conjunto de datos de entrenamiento estaba compuesto por 80 sentinazos y 43 *look-alikes*, mientras que el de test, que estaba analizado por un operador, tenía 11 sentinazos, 6 *look-alikes* y 4 indefinidos. El 81,8 % de los sentinazos y el 100 % de los *look-alikes* fueron correctamente clasificados usando la distancia de Mahalanobis, mientras que el clasificador de probabilidad compuesto identificó el 90,9 % de los sentinazos y el 67 % de los *look-alikes*.

La eficacia de los diferentes métodos de clasificación es totalmente dependiente de las fases previas y por tanto, es difícil compararlos. Es necesario segmentar el candidato para poder clasificarlo, siendo clave para su correcta clasificación las características seleccionadas y su poder discriminatorio. Es importante también destacar que las eficacias de algunos clasificadores viene dada a nivel de píxel [54] y las de otros a nivel de candidato [103]. En general, la eficacia a nivel de píxel suele ser más elevada dado que se evalúa una muestra mayor y los errores de clasificación, si son pocos, quedan más diluidos.

Capítulo 5

Sistema automático de detección de vertidos en imágenes de SAR

5.1. Descripción del proyecto

Galicia es una región en el noroeste de España con más de 1.200 km de costa y aproximadamente 720 playas donde el mar ha jugado siempre un papel muy importante en la vida de sus habitantes. La pesca y la agricultura han sido pilares fundamentales de su economía y, actualmente, el sector turístico juega también un papel esencial. Sus costas, playas y las bondades de su gastronomía, particularmente la relacionada con el producto de mar (pescados y mariscos), son en buena parte culpables del incremento turístico. Además, se encuentran en esta región importantes astilleros y puertos tanto civiles como militares.

La situación geográfica de Galicia convierte a sus costas en un paso obligado para buena parte del tráfico marítimo europeo. El Esquema de Separación de Tráfico de Fisterra (ESTF), mostrado en la Figura 5.1, es utilizado para dirigir y controlar este tránsito. Las vías de circulación están, empezando por la más próxima a la costa, a 21,7 millas náuticas (40 km), 28,3 millas náuticas (52,4 km), 35,5 millas náuticas (65,7 km) y 39,5 millas náuticas (73 km) de Cabo Fisterra. Las vías C y D del ESTF son usadas para navegar de norte a sur, mientras que la A y B se utilizan en sentido contrario. Las vías B y D están reservadas para embarcaciones con mercancías peligrosas.

Cada año miles de buques de mercancías navegan a lo largo del ESTF y, debido a la normativa internacional, todos los barcos tienen que informar de su posición al servicio de

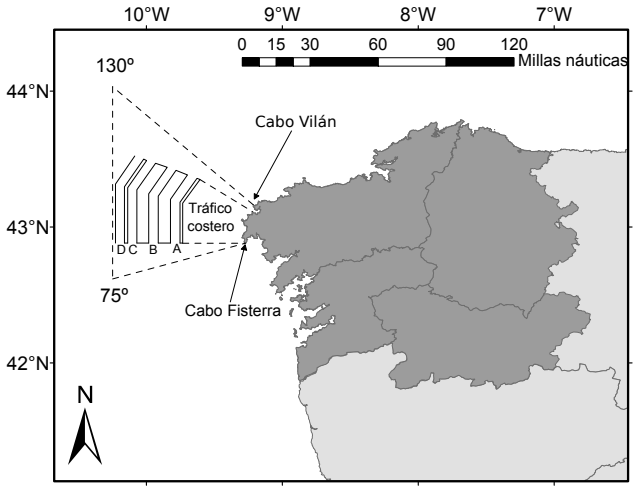

Figura 5.1: Esquema de Separación de Tráfico de Fisterra.

Tráfico Marítimo de Fisterra, al cruzar las líneas de demora 130° a Cabo Vilán y 75° a Cabo Fisterra. La Tabla 5.1 muestra el número de informes recibidos entre los años 1999 y 2009.

Año	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Buques	41.829	44.561	44.331	43.209	43.469	42.538	43.212	41.942	42.136	42.354	40.320

Tabla 5.1: Número de embarcaciones identificadas en el ESTF.

Este intenso tráfico tiene graves consecuencias medioambientales en forma de accidentes y vertidos como, por ejemplo, el del petrolero Urquiola en 1976 o el del Mar Egeo en 1992. Sin embargo, la catástrofe más importante y con mayor repercusión ha sido la del buque petrolero Prestige que, el 13 de Noviembre del año 2002, sufrió un accidente frente a la costa gallega para acabar hundiéndose al cabo de seis días. En la Figura 5.2 se observa la extensión del vertido 5 días después de la catástrofe. La amplia cobertura científica del accidente permitió obtener multitud de datos y de información, lo que unido a un incremento de los recursos económicos tuvo como consecuencia el desarrollo de varios proyectos para la detección y seguimiento de grandes vertidos [90] [18] [34].

A pesar de que los medios de vigilancia y prevención se incrementaron después de la catástrofe, se descubren regularmente pequeños vertidos (sentinazos) provenientes de las operaciones de mantenimiento y limpieza de sentinas. Es conveniente recordar que los pequeños

vertidos, a pesar de no ser tan mediáticos, son la fuente de contaminación por hidrocarburos más importante (véase la Figura 4.1).

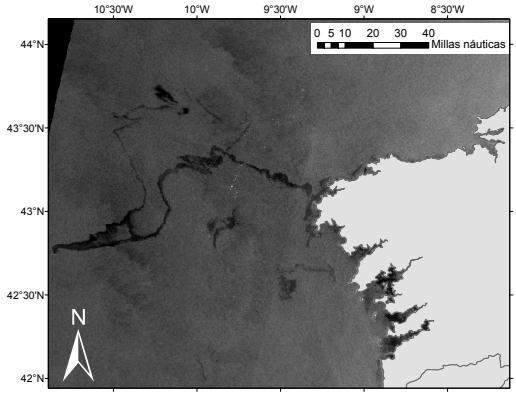

Figura 5.2: Imagen SAR del vertido provocado por el accidente del buque petrolero Prestige (17/11/2002).

La Agencia española de Salvamento y Seguridad Marítima (SASEMAR) es la autoridad pública encargada de monitorizar las costas gallegas en general y el ESTF en particular. SASEMAR, como se observa en la Figura 5.3, cuenta con varias bases de vigilancia en Galicia y con recursos, tanto marinos como aéreos, para efectuar la monitorización y la vigilancia. El uso de buques y aviones es costoso y limitado, por lo que el acuerdo suscrito con la Agencia Europea de Seguridad Marítima (EMSA) para recibir hasta un máximo 12 avisos al mes provenientes de la inspección de imágenes SAR sobre posibles vertidos es especialmente útil. Estos avisos son evaluados por el personal de SASEMAR y, cuando se considera apropiado, envían sus propios recursos a la zona.

En general, una nación costera como es España y en particular Galicia con su intenso tráfico marítimo y su trágico historial de catástrofes medioambientales, debería poseer su propio sistema de análisis de imágenes SAR sin depender de sistemas de terceros. Con este ánimo, se desarrolló un sistema semiautomático de detección de vertidos de hidrocarburos a partir de imágenes SAR que, aunque resulta fácilmente adaptable a otras regiones, fue optimizado para su uso en las costas gallegas [79]. Se prestó especial atención a su implementación con el objetivo de dar una respuesta en tiempo cuasi real sin sacrificar la eficiencia. Las diferentes herramientas, software o librerías utilizadas para el desarrollo del prototipo están soportadas por licencias públicas, lo que permite la libre distribución del producto final. Los siguientes

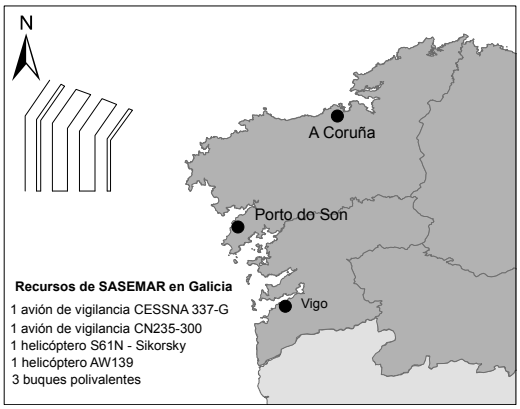

Figura 5.3: Bases de la SASEMAR en Galicia y recursos asignados a la región.

apartados detallan el desarrollo del algoritmo utilizado y su incorporación a un prototipo. La Sección 5.2 presenta la DB de imágenes usada para el desarrollo y las fuentes de datos utilizadas. Las particularidades del algoritmo se describen en las Secciones 5.3, 5.4, 5.5 y 5.6. Por último, la Sección 5.7 se centra en la descripción del prototipo implementado y en su integración en RETELAB.

5.2. Fuentes de datos

5.2.1. Satélite Envisat

En Marzo del 2002 la ESA puso en órbita el satélite de observación terrestre Envisat [1]. Su objetivo principal era dotar a Europa de una mayor capacidad para la observación de la Tierra desde el espacio y dar continuidad a las misiones predecesoras de los satélites ERS-1 y ERS-2. La instrumentación del Envisat estaba diseñada para realizar observaciones y mediciones del océano, la tierra, el hielo y la atmósfera. Estuvo operativo hasta el 8 de Abril del 2012, día en que se perdió el contacto. El 9 de Mayo del 2012, la ESA dio por finalizada oficialmente la misión. Algunos de los parámetros orbitales del satélite pueden consultarse en la Tabla 5.2.

La continuidad de las misiones europeas para la observación de la Tierra está garantizada con la serie de satélites Sentinel que está siendo desarrollada por la ESA dentro del programa

de Monitorización Global para el Medioambiente y la Seguridad (GMES). Se espera que el primer satélite Sentinel sea lanzado durante el año 2013.

Órbitas por día	14,3
Periodo repetición de ciclo	35 días
Órbitas por ciclo	501
Período orbital	100,59 min.
Velocidad de la órbita	7,45 km/s
Inclinación orbital	$98,55^{\circ}$
Altitud media	799,8 km

Tabla 5.2: Parámetros orbitales del satélite Envisat.

A bordo del Envisat viajaban 10 instrumentos dedicados a la observación de la Tierra:

– El instrumento de Orbitografía y Radioposicionamiento Doppler Integrado por Satélite (DORIS) ofrecía datos sobre la órbita del satélite con gran precisión. Esta información era utilizada para determinar la posición exacta del satélite.

– El Espectrómetro de Imágenes de Resolución Media (MERIS) medía la reflectancia de la Tierra en el rango del espectro solar, analizando el color de los océanos y la cobertura del terreno. Los mapas de alta resolución generados con sus datos son utilizados para evaluar los efectos del cambio climático.

– El Radiómetro Avanzado de Exploración Longitudinal (AATSR) permitía medir la temperatura de la superficie de la Tierra y de los océanos a partir de la radiación térmica infrarroja que emite nuestro planeta.

– El Altímetro de Radar (RA-2) era capaz de registrar la topografía de la superficie terrestre con una precisión de unos pocos centímetros, lo que hacía posible analizar, por ejemplo, la evolución temporal del nivel del mar.

– El objetivo principal del Radiómetro de Microondas (MWR) era el de realizar mediciones del vapor de agua de la atmósfera y del contenido de agua líquida de las nubes.

– El instrumento para Monitorización Global del Ozono mediante la Ocultación de Estrellas (GOMOS) proporcionaba una cartografía global del ozono en altitud y un seguimiento de tendencia de gran precisión.

– El Interferómetro de Michelson para Ecografía Atmosférica Pasiva (MIPAS) era un espectrómetro independiente de la luz solar que servía para realizar mediciones de alta resolución del espectro de emisiones gaseosas en la atmósfera terrestre.

– La misión principal del Espectrómetro de Absorción de Exploración e Imágenes para Cartografía Atmosférica (SCIAMACHY) era registrar el espectro de rayos solares que pasan a través de la atmósfera. Se utilizaba para encontrar huellas de absorción espectral producidas por determinados gases y, entre otras utilidades, permitía realizar mapas globales de polución atmosférica.

– El Retrorreflector Láser (LRR) era un sensor pasivo utilizado como reflector de pulsos láser de alta potencia enviados desde estaciones en tierra y que proporcionaba la distancia entre el satélite y la estación.

– El Radar de Apertura Sintética Avanzado (ASAR) era un novedoso instrumento SAR del que se obtuvieron los datos utilizados para el desarrollo de SENTINAZOS. La siguiente Sección profundiza su descripción.

La mayoría de los instrumentos presentados, excepto el MWR y el RA-2, tenían cobertura global y necesitaban entre 1 y 3 días para completarla.

Radar de Apertura Sintética Avanzado

El ASAR es un sensor activo que opera en el rango de las microondas, concretamente en la banda C con una frecuencia de 5,331 GHz. En la figura 5.4 se muestra el espectro electromagnético, ofreciendo mayor detalle en el rango de las microondas.

Las características particulares del sensor, además de su independencia de la luz solar y de la cobertura nubosa, lo hacen especialmente eficiente en el estudio de temas tan diversos como: frentes oceánicos, oleaje marino, detección de barcos y contaminantes, movimiento y extensión de hielos, agricultura, estudios urbanísticos, desastres naturales, etc. Dependiendo del área de estudio, se trabajará en uno de los 5 modos de operación soportados por el sensor y que son descritos en la Tabla 5.3.

Envisat *Wide Swath Mode*

El Envisat *Wide Swath Mode* (WSM) es el modo de operación que mejor se adapta a la búsqueda de vertidos de hidrocarburos. Por un lado, gracias a la técnica ScanSAR, ofrece una

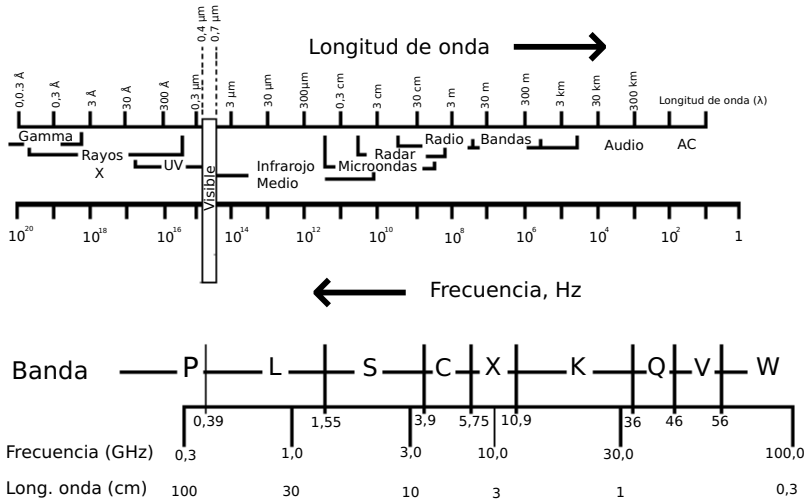

Figura 5.4: Espectro electromagnético. Figura adaptada de la web de la ESA [1].

Modo	Trazas	Polarización	Resolución (m)	Cobertura aproximada (km)	IA (grados sexagesimales)	Ruido de fondo (dB)
Image Mode (IM)	IS1, IS2, IS3, IS4, IS5, IS6 o IS7	VV o HH	30	56 (traza 7) - 100 (traza 1)	$15° - 45°$	-19 a -22
Alternating Polarisation (AP)	IS1, IS2, IS3, IS4, IS5, IS6 o IS7	HH/VV, HH/HV, VV/VH	30	56 (traza 7) - 100 (traza 1)	$15° - 45°$	-19 a -22
Wide Swath (WS)	SS1 + SS2 + SS3 + SS4 + SS5	VV o HH	150	400	$17° - 42°$	-21 a -26
Global Monitoring (GM)	SS1 + SS2 + SS3 + SS4 + SS5	VV o HH	1.000	400	$17° - 42°$	-32 a -35
Wave (WV)	IS1, IS2, IS3, IS4, IS5, IS6 o IS7	VV o HH	30	5	$15° - 45°$	-20 a -22

Tabla 5.3: Modos de operación del sensor ASAR a bordo del Envisat.

amplia cobertura de 405 km y una resolución media de 150 m. Por otro lado, la relación entre el ruido de fondo y la retrodispersión es adecuada para la búsqueda de sentinazos en casi todas las condiciones cuando se utiliza la polarización vertical-vertical.

Polarización Los radares pueden transmitir el haz energético con polarización horizontal (H) o con polarización vertical (V) y recibir la señal retornada nuevamente con polarización horizontal, vertical o ambas. Esto es interesante debido a que los diferentes materiales

de los que está compuesta la cubierta terrestre reflejan las señales con intensidades diferentes dependiendo de la polarización e incluso algunos materiales convierten una polarización en otra. Dependiendo del objetivo de la investigación, es más adecuado un tipo u otro de polarización. Los productos de Envisat WSM pueden trabajar con dos tipos de polarización, la vertical-vertical (VV) y la horizontal-horizontal (HH).

Ruido de fondo Los sistemas para la adquisición de datos y procesamiento de señales están afectados por diferentes tipos de ruido y de señales no deseadas. La suma de todo esto es lo que se define como ruido de fondo o *noise floor*. La intensidad de este ruido sitúa el umbral o medida mínima que puede ser tomada e identificada con certeza, es decir, no se puede identificar nada cuya intensidad de señal sea inferior al ruido de fondo, ya que queda enmascarado por dicho ruido.

La retrodispersión del océano varía dependiendo del IA, la polarización y la velocidad y dirección del viento. Los productos WSM de Envisat con polarización VV poseen un ruido de fondo, entre -21 dB y -26 dB, que se encuentra por debajo de la media de retrodispersión del océano en casi todas las condiciones. Esta característica permite buscar la presencia de hidrocarburos, ya que éstos se asocian a valores de retrodispersión inferiores a los generados por las áreas del mar sin presencia de contaminantes. La Figura 5.5 muestra la retrodispersión del océano obtenida de una imagen SAR que no tenía presencia de hidrocarburos. Se puede observar que la retrodispersión varía según el IA y la velocidad del viento.

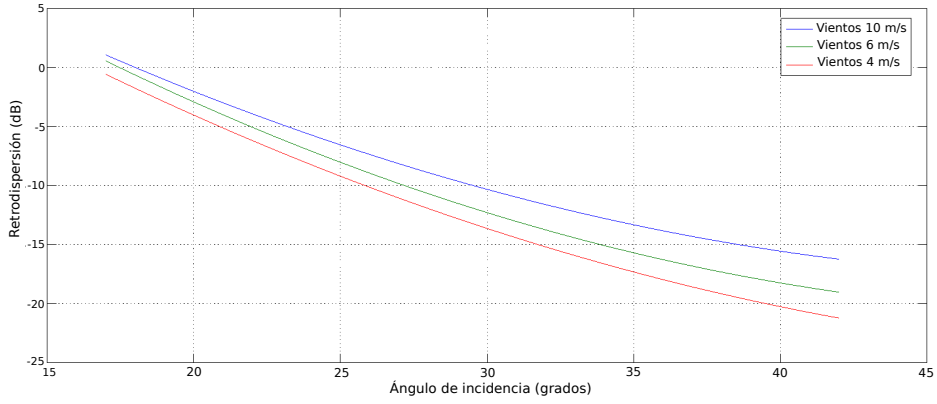

Figura 5.5: Retrodispersión media obtenida de una imagen ASAR WSM para vientos de 4 m/s, 6 m/s y 10 m/s.

5.2.2. Colección de datos

Para el desarrollo del sistema se ha contado con una colección de 47 imágenes WSM con polarización VV del Envisat obtenidas a través de un convenio con la ESA. Las imágenes, tomadas entre los años 2007 y 2011, corresponden al noroeste de la Península Ibérica, estando la mayoría centradas en la costa gallega, como puede verse en el mapa de densidad mostrado por la Figura 5.6.

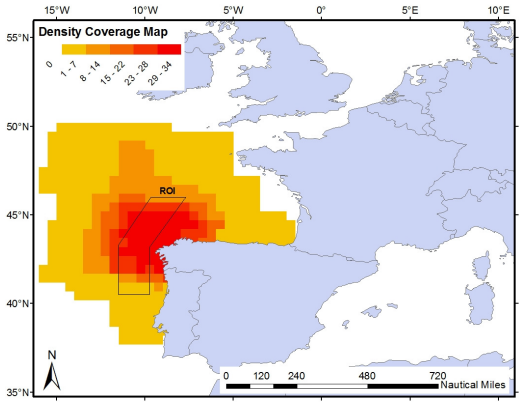

Figura 5.6: Representación de la densidad de cobertura de la Base de Datos.

La región de estudio es una ampliación del ESTF construida tal y como se muestra en la Figura 5.7.

Las imágenes de los años 2007 y 2008 contienen sentinazos detectados por operadores de la EMSA a los que se les ha asignado una probabilidad (alta, baja o media) de ser hidrocarburos. El resto de la colección contiene sentinazos y *look-alikes* confirmados a través de misiones aéreas de SASEMAR. El número de imágenes de la colección y cuántas de ellas contienen sentinazos (confirmados o no) se puede ver en la Figura 5.8a agrupadas por año. La clasificación de los sentinazos de la colección es mostrada en la Figura 5.8b.

La Figura 5.9 muestra la posición geográfica de los vertidos presentes en la DB. Éstos se caracterizan por estar agrupados en torno al ESTF, lo cual es debido a que es paso obligado para los buques y, en consecuencia, sufre en mayor medida el efecto de los sentinazos.

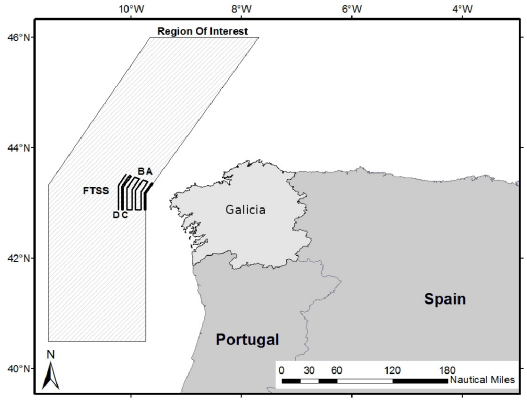

Figura 5.7: Área de estudio y ESTF.

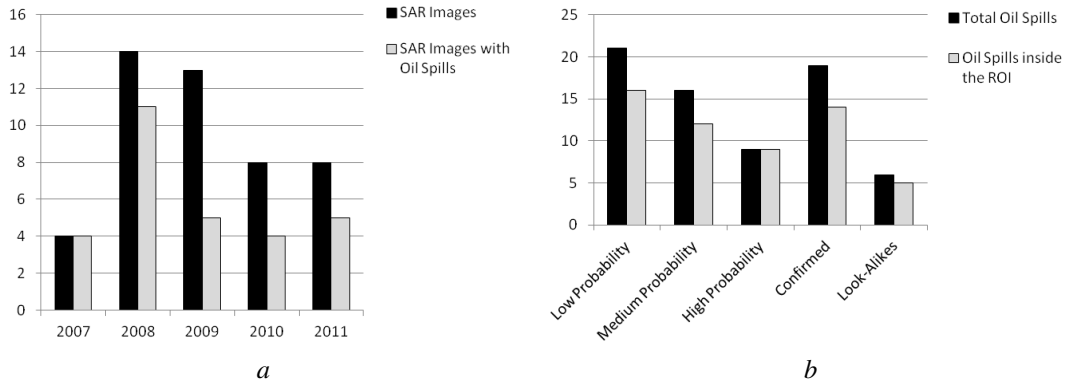

Figura 5.8: a) Imágenes SAR e imágenes con sentinazos agrupadas por año. b) Clasificación de las manchas de la Base de Datos.

5.3. Preprocesado

5.3.1. Calibrado y proyección cartográfica

La DB utilizada para el desarrollo del proyecto almacena productos Envisat WSM de nivel 1. Para que estos productos se adapten a las necesidades del proyecto, es necesario aplicarles varias operaciones de preprocesado: por un lado, las imágenes son calibradas [95] con el objetivo de conseguir valores que realmente representen la retrodispersión de la superficie

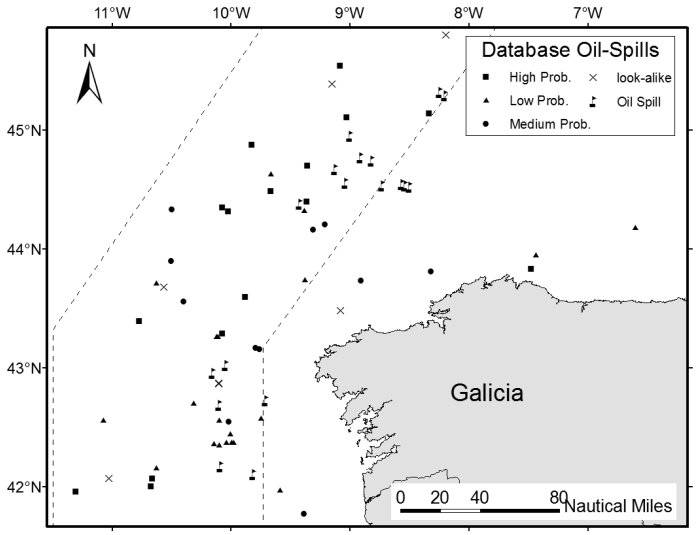

Figura 5.9: Localización de los sentinazos contenidos en las imágenes de la colección.

estudiada y que puedan ser comparados entre ellos. La calibración aplicada consiste en una serie de correcciones radiométricas que permiten comparar imágenes sin importar si fueron obtenidas por el mismo sensor, en qué momento fueron tomadas, el modo usado o si fueron procesadas por diferentes procesadores. Por otro lado, las imágenes calibradas son proyectadas a un mismo sistema de referencia de coordenadas.

Una vez realizadas estas actividades de preprocesado, las imágenes pueden utilizarse como datos de entrada al sistema; no obstante, se aplicarán otras dos funciones, un enmascaramiento de la tierra y un filtrado de ruido, antes de proceder a la segmentación con el objetivo de optimizar la información contenida en las mismas.

5.3.2. Máscara de tierra

El objetivo del sistema es la detección de vertidos de hidrocarburos en la superficie oceánica. Las imágenes utilizadas tienen una cobertura de 400 km x 400 km y, al estar centradas en el ESTF, contienen habitualmente regiones de tierra que corresponden a la costa noroeste de la Península Ibérica. Para no procesar dichas regiones, éstas son enmascaradas asignando valores nulos a los píxeles que las forman.

Desde el punto de vista espacial, la costa gallega es muy compleja debido a la presencia de las rías y otros accidentes geográficos. La costa y sobre todo las rías funcionan como parapeto protegiendo de los vientos a la superficie oceánica y, como se ha visto en el Capítulo 4, las áreas marinas dominadas por vientos de baja intensidad se visualizan en las imágenes SAR de forma similar a las afectadas por vertidos (falsos positivos). Para evitar estos *look-alikes* se ha ampliado la máscara de tierra en 3 millas náuticas tal y como puede apreciarse en la figura 5.10.

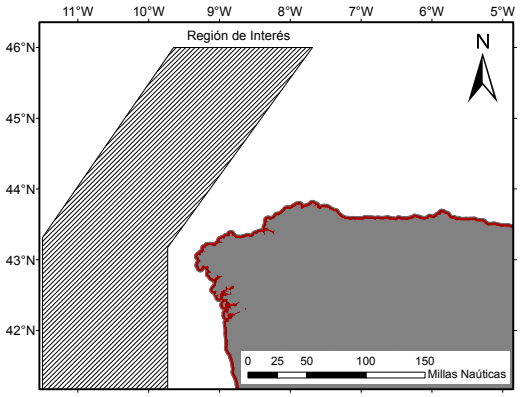

Figura 5.10: Máscara de tierra utilizada. La franja roja representa una ampliación de 3 millas náuticas a partir de la línea de costa.

5.3.3. Filtro de ruido

Previamente al procesado de las imágenes, es necesario aplicar un filtro de mediana con una ventana de 3 x 3 píxeles con el objetivo de eliminar/suavizar el ruido de las imágenes. La Figura 5.11 muestra un ejemplo de aplicación del filtro de mediana y cómo el valor de un píxel se ve influenciado por los de su entorno.

5.4. Segmentación

El objetivo de esta fase es detectar, sobre la imagen SAR, todos los píxeles susceptibles de pertenecer a un vertido de hidrocarburo y generar una imagen binaria donde los candidatos destaquen de la superficie marina limpia.

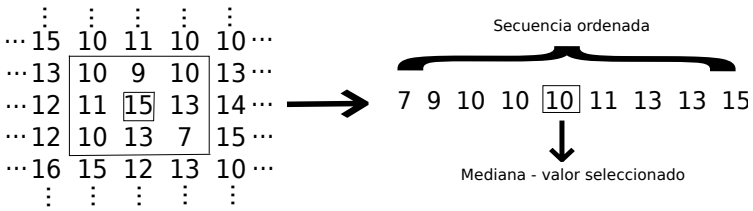

Figura 5.11: Ejemplo de uso del filtro de mediana para reducir el ruido de la imagen. Imagen adaptada de [35].

Dado que los bajos niveles de intensidad, generados por las áreas con baja retrodispersión, están asociados a hidrocarburos (además de a otros fenómenos naturales), es lógico pensar que una segmentación basada en umbrales pueda ser el método más adecuado. La complejidad de las imágenes SAR (por ejemplo, la fuerte dependencia del nivel medio de intensidad con el IA) provoca que los umbrales simples no sean adecuados. Una función de umbral adaptativo que tenga en cuenta el IA del píxel analizado, además de otros factores, parece la mejor opción.

5.4.1. Establecimiento del umbral adaptativo

Con el objetivo de establecer una función de umbral adaptativo, se muestrearon 1.652 píxeles pertenecientes a zonas afectadas por vertidos de hidrocarburos. De cada píxel se registró su valor de intensidad y su IA asociado. La Figura 5.12 presenta gráficamente los píxeles muestreados agrupados por su IA.

En este histograma se observa que los IAs de los píxeles seleccionados no están distribuidos uniformemente y la razón es que la mayoría de las imágenes de la DB están centradas en el ESTF, por lo que la relación entre las áreas de la imagen y los IAs se mantiene relativamente constante. El rango de IAs con mayor número de píxeles muestreados suele coincidir con el área cubierta por el ESTF, que es la zona que contiene la mayor cantidad de vertidos. Los píxeles de la muestra fueron agrupados de acuerdo con su IA en intervalos de 1° entre los 17° y los 42°. Para cada intervalo se calculó la media y su desviación típica y se descartaron los píxeles con valores de intensidad superiores a la suma de los dos valores calculados. Para cada intervalo se seleccionaron los valores de intensidad máximos (véase Ecuación 5.1) y, con estos puntos como límite superior, se realizó un estudio de regresión para obtener una función que los aproximase.

$$x = max(B); B = \{y \in B \ / \ y \leq \nu(A) + \sigma(B)\} \tag{5.1}$$

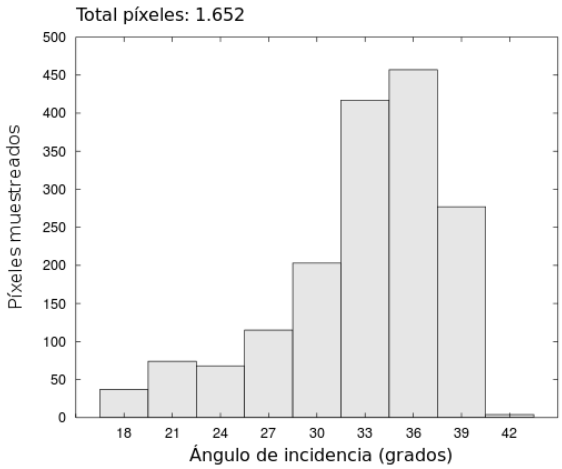

Figura 5.12: Píxeles muestreados agrupados según su IA.

El resultado del análisis fue que tanto una función de cuarto grado (5.2) como una exponencial negativa (5.3) se ajustaban de forma significativa a los datos de la muestra:

$$y = -6{,}3401 \cdot 10^{-6}x^4 + 0{,}00073027x^3 - 0{,}029919x^2 + 0{,}50015x - 2{,}6381 \tag{5.2}$$

$$y = 7{,}4199e^{-0{,}18212x} \tag{5.3}$$

Estas funciones debían representar los valores máximos de intensidad que podía almacenar un píxel, en función de su IA asociado, en caso de que representase un vertido, es decir, todo píxel con un valor de intensidad igual o menor se consideraría vertido. Se optó por una aproximación agresiva, ya que es preferible, en esta fase de segmentación, obtener algunos falsos positivos a descartar vertidos.

Las pruebas efectuadas aplicando cualquiera de las dos funciones mostraron demasiados falsos positivos. La obtención de *look-alikes* en la fase de segmentación no es preocupante e incluso es esperada, dado que es la fase de clasificación la que tradicionalmente asume la responsabilidad de seleccionar cuáles son los sentinazos y cuáles no pero sería deseable obtener el mínimo posible y así disminuir la responsabilidad del clasificador.

Las funciones obtenidas estaban basadas en la intensidad y en el IA pero determinados estudios previos [40] ponen de manifiesto la importancia de los factores meteorológicos y,

particularmente, los vientos en la detección de sentinazos. Dado que el nivel de retrodispersión tiene relación directa con la rugosidad de la superficie (véase la Figura 4.5) y ésta se debe a la presencia de viento, la función adaptativa debería tener en cuenta los datos del viento de la zona estudiada. La Figura 5.5 muestra la influencia de la velocidad del viento en la intensidad. Utilizar datos de viento en la fase de segmentación permitirá realizar una segmentación más precisa facilitando el trabajo posterior del clasificador.

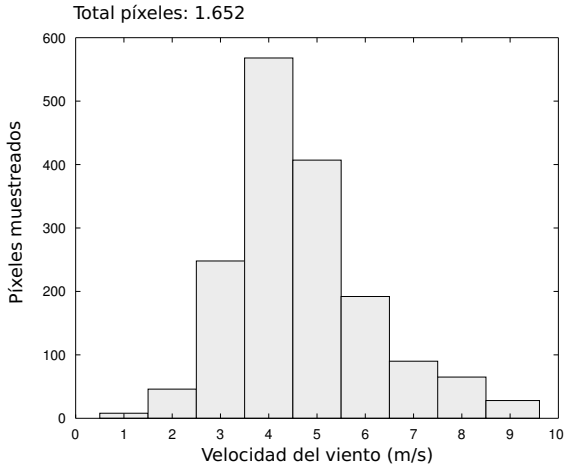

Figura 5.13: Elementos de la muestra agrupados por la velocidad del viento.

La forma más directa de acceder a los datos de viento correspondientes a la zona cubierta por una imagen SAR es a través de los datos contenidos en la propia imagen. CMOD5 [60] es un modelo geofísico desarrollado para datos obtenidos en la banda C que permite relacionar la retrodispersión generada por la rugosidad de la superficie oceánica con la velocidad y dirección del viento. Utilizando este modelo se obtuvieron los datos de viento correspondientes a las imágenes de la DB. Los resultados generados, con una resolución menor que la de la imagen SAR, fueron interpolados usando el algoritmo Kriging [88] para obtener la misma resolución que la imagen original.

La muestra de 1.652 píxeles fue enriquecida añadiendo los datos de velocidad de viento calculados con el modelo CMOD5. Los píxeles de la muestra fueron agrupados según la velocidad del viento (véase Figura 5.13). La mayoría de estos píxeles tienen asociadas velocidades de viento entre los 3 y 7 m/s, por lo que cubren el rango óptimo para la detección de vertidos [54]. Los píxeles con vientos mayores a 7 m/s se agruparon en un sólo intervalo, ya que la

muestra no tenía suficientes datos como para realizar más divisiones. Cada grupo creado fue nuevamente dividido en subconjuntos en base al IA. En cada subconjunto (con al menos 10 valores) se calculó la media y la desviación típica de los valores de intensidad. Finalmente se escogió el valor de intensidad más alto para cada subconjunto siempre y cuando fuese igual o menor a la suma de la media y el doble de la desviación típica de dicho subconjunto. Nótese que en esta ocasión, al ajustar mejor los valores gracias a los vientos, se optó por relajar el umbral máximo empleando dos veces la desviación típica y no sólo una como en el caso de la umbralización sin vientos (véase Ecuación 5.1). El esquema del proceso seguido se muestra en la Figura 5.14.

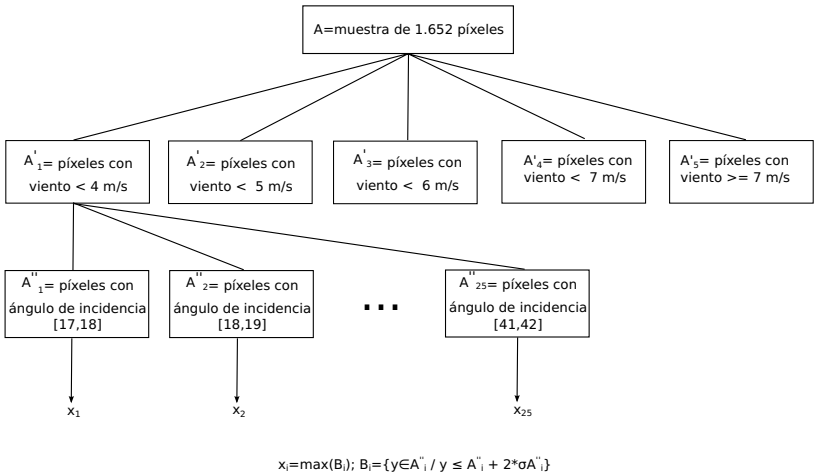

Figura 5.14: Esquema de acciones realizadas para establecer el umbral adaptativo.

El resultado de este proceso es un conjunto de intensidades para cada rango de velocidad estudiado. Realizando un análisis de regresión sobre los datos se comprobó que, al igual que en la primera aproximación, una función de cuarto grado y una exponencial negativa se ajustaban de forma significa a la muestra. Estudiando dichas funciones con más detalle y a través de pruebas de segmentación, se comprobó que los mejores resultados se obtenían usando la función de cuarto grado en los ángulos de incidencia menores y la exponencial negativa en los mayores. Se estableció una función definida por partes para el cálculo adaptativo del umbral en el que el IA es utilizado para establecer el punto de corte entre las partes. La Tabla 5.4 presenta los coeficientes de las funciones adaptados según la velocidad del viento, así como los puntos de corte establecidos por los IAs.

Velocidad del viento	Función de cuarto grado	Punto de corte	Función exponencial negativa
$vel. <= 4m/s$	$6,3662 * 10^{-6}x^4 - 0,00083671x^3 + 0,041262x^2 - 0,90719x + 7,5353$	$37,8°$	$5,9169e^{-0,17881x}$
$vel. <= 5m/s$	$3,0874 * 10^{-6}x^4 - 0,00043845x^3 + 0,023424x^2 - 0,55931x + 5,0588$	$36,25°$	$6,6689e^{-0,17944x}$
$vel. <= 6m/s$	$2,1022 * 10^{-6}x^4 - 0,00031131x^3 + 0,017414x^2 - 0,4365x + 4,1503$	$36,82°$	$5,7770e^{-0,17217x}$
$vel. <= 7m/s$	$2,7742 * 10^{-6}x^4 - 0,00039094x^3 + 0,02086x^2 - 0,50088x + 4,5883$	$37,52°$	$5,4477e^{-0,16858x}$
$vel. > 7m/s$	$1,2416 * 10^{-6}x^4 - 0,00019466x^3 + 0,011604x^2 - 0,31143x + 3,182$	$36,8°$	$6,2053e^{-0,17101x}$

Tabla 5.4: Coeficientes y puntos de corte para las funciones de umbral adaptativo.

Las funciones exponenciales negativas para vientos de 4 m/s, 6 m/s y mayores que 7 m/s se representan gráficamente en la Figura 5.15

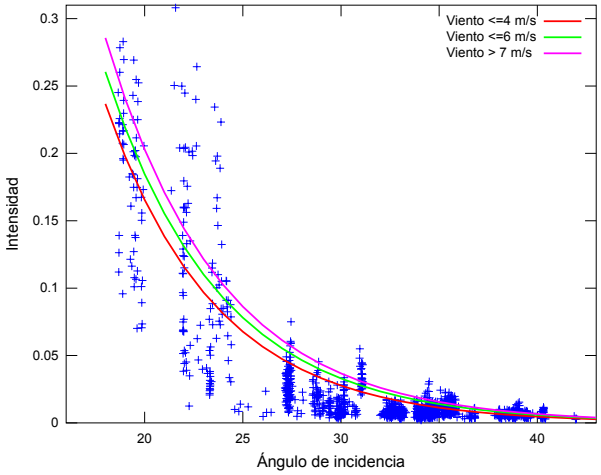

Figura 5.15: Representación de la aproximación de la intensidad en función del IA para distintos valores de la velocidad del viento.

5.4.2. Aplicación del umbral adaptativo

El proceso de segmentación se aplica sobre todos los píxeles de una imagen SAR excepto los que previamente han sido enmascarados como tierra y los que pertenecen a zonas fuera del área de alcance del sensor.

De cada píxel procesado se extrae su intensidad, el IA y la velocidad del viento. A partir de la velocidad del viento y de los coeficientes presentados en la Tabla 5.4, se obtienen las funciones adecuadas y el punto de corte (IA) entre la función polinómica y la exponencial negativa. Según el IA del píxel y tras compararlo con el punto de corte, se selecciona una

de las dos funciones y se utiliza el ángulo como parámetro. La función devolverá un umbral que identificará el máximo valor de intensidad que podría tener el píxel analizado para ser considerado parte de un sentinazo. Si la intensidad es menor o igual al umbral, el píxel será etiquetado como candidato y, en caso contrario, como superficie limpia.

5.4.3. Filtro de vientos

Dado que existen estudios previos que consideran que es necesaria una velocidad del viento de al menos 3 m/s para detectar un sentinazo, los píxeles segmentados que estén afectados por vientos de intensidad menor que ese umbral son considerados *look-alikes* y son eliminados de la imagen segmentada.

5.4.4. Filtro de área

Los medios de vigilancia contra la polución como los buques y aviones tienen un coste de uso elevado y son limitados, por lo que sólo son utilizados cuando la situación es potencialmente dañina o el vertido tiene una dimensión considerable. Las estructuras formadas por los píxeles segmentados que tengan un área menor que 50 píxeles (equivalente a $0,3km^2$) son consideradas como falsos positivos o como pequeños sentinazos no dañinos que pueden ser eliminados de la imagen segmentada.

5.5. Extracción de características

El resultado de la fase de segmentación es una imagen binaria donde aparecen resaltados todos los sentinazos junto con un gran número de *look-alikes*. Es tarea de un clasificador separar los sentinazos de los falsos positivos y, para una correcta clasificación, es necesario obtener un vector de características para cada candidato que pueda ser analizado por el clasificador.

El vector de características utilizado para el desarrollo de este algoritmo está basado en tres tipos de características:

– Características de forma: la forma puede ser un factor determinante para clasificar un candidato. Estudios anteriores [92] han mostrado que, debido a su naturaleza, los sentinazos pueden agruparse en 5 grandes clases desde el punto de vista de su forma (véase la Figura 4.7).

– Características físicas: se han incorporado características asociadas al nivel de intensidad de los píxeles dado que su importancia y utilidad ha sido refrendada por estudios previos [37].

– Información contextual: Los datos de contexto en general y más concretamente los de vientos [40] son de gran utilidad a la hora de detectar sentinazos y deben ser incorporados al vector de características.

Tipo	Nombre	Descripción
Forma	APR	Área / Perímetro
	Elongación	Eje mayor / Eje menor
	MPR	Eje mayor / Perímetro
	Rectangularidad	Área / Área (Min. rect. engloba)
	Circularidad	Perímetro^2 / (4 π*Área)
	Thickness	Número de erosiones necesarias para dividir un objeto.
	Hu1	$(\eta_{20}+\eta_{02})$
	Hu2	$(\eta_{20}-\eta_{02})+4\eta_{11}^2$
	Hu3	$(\eta_{30}+3\eta_{12})^2+(3\eta_{21}-\eta_{03})^2$
	Hu4	$(\eta_{30}+\eta_{12})^2+(\eta_{21}+\eta_{03})^2$
	Hu5	$(\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2]+(3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2]$
	Hu6	$(\eta_{20}-\eta_{02})[(\eta_{30}-\eta_{12})^2-(\eta_{21}\eta_{03})^2]+4\eta_{21}(\eta_{30}+\eta_{12})(\eta_{21}+\eta_{03})$
	Hu7	$(3\eta_{21}-\eta_{30})(\eta_{00}+\eta_{12})[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2]+(3\eta_{12}-\eta_{03})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2]$
	Flusser y Suk - I_1	$(\eta_{20}\eta_{02}-\eta_{11}^2)/\eta_{00}^4$
	Flusser y Suk - I_2	$(-\eta_{30}^2\eta_{03}^2+6\eta_{30}\eta_{21}\eta_{12}\eta_{03}-4\eta_{30}\eta_{12}^3-4\eta_{21}^3\eta_{03}+3\eta_{21}^2\eta_{12}^2)/\eta_{00}^{10}$
	Flusser y Suk - I_3	$(\eta_{20}\eta_{21}\eta_{03}-\eta_{20}\eta_{12}^2-\eta_{11}\eta_{30}\eta_{03}+\eta_{11}\eta_{21}\eta_{12}+\eta_{02}\eta_{30}\eta_{12}-\eta_{02}\eta_{21}^2)/v_{00}^7$
	Flusser y Suk - I_4	$(-\eta_{20}^3\eta_{03}^2+6\eta_{20}^2\eta_{11}\eta_{12}\eta_{03}-3\eta_{20}^2\eta_{02}\eta_{12}^2-6\eta_{20}\eta_{11}^2\eta_{21}\eta_{03}-6\eta_{20}\eta_{11}^2\eta_{12}^2+12\eta_{20}\eta_{11}\eta_{02}\eta_{21}\eta_{12}-3\eta_{20}\eta_{02}^2\eta_{21}^2+2\eta_{11}^3\eta_{30}\eta_{03}+6\eta_{11}^3\eta_{21}\eta_{12}-6\eta_{11}^2\eta_{02}\eta_{30}\eta_{12}-6\eta_{11}^2\eta_{02}\eta_{21}^2+6\eta_{11}\eta_{02}^2\eta_{30}\eta_{21}-\eta_{02}^3\eta_{30}^2)/\eta_{00}^{11}$
Contexto	AI	IA respecto al centroide de la mancha.
	Viento	Velocidad del viento en el área estudiada.
Físicas	Intensidad	Intensidad media de los píxeles contenidos en la mancha candidato.
	ASR	Ratio de la intensidad del candidato respecto al área circundante.

Nota: η_{ij} representa a los momentos centrales normalizados.

Tabla 5.5: Vector de características.

El vector generado (véase Tabla 5.5) está compuesto por 21 componentes de los que 17 corresponden a características de forma. Para disminuir la dimensionalidad del vector se aplicó un Análisis de Componentes Principales (PCA) [67] sobre las características de forma. La Tabla 5.6 contiene los porcentajes de varianza explicados por los componentes principales obtenidos tras el PCA, mientras que la Figura 5.16 muestra gráficamente el porcentaje explicado por el subconjunto de componentes más representativo. Finalmente, se sustituyeron las

17 características de forma por los 5 primeros componentes cubriendo así más del 92 % de la varianza y reduciendo considerablemente la dimensionalidad del vector. Los coeficientes de los componentes principales seleccionados se muestran en la Tabla 5.5.

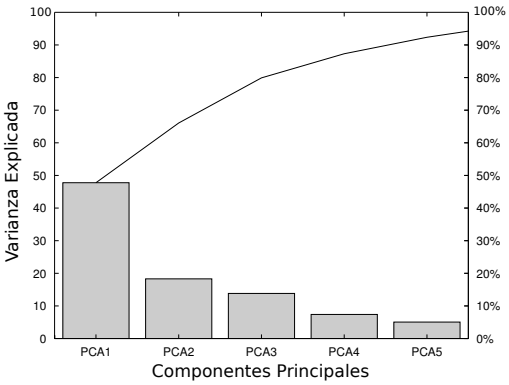

Figura 5.16: Componentes principales y porcentaje de varianza explicado.

Componente Principal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Varianza explicada (%)	47,7716	18,3024	13,8332	7,3918	5,0537	3,7671	1,9663	0,8758	0,4884	0,3178	0,0938	0,0693	0,0414	0,0142	0,0089	0,0041	0,0001

Tabla 5.6: Componentes principales y porcentaje de varianza explicado.

5.6. Clasificación

Una vez caracterizados los candidatos, es necesario clasificarlos y asignarles una clase (sentinazo o *look-alike*). El vector de características de cada candidato es la entrada para un clasificador cuya salida será una clase o etiqueta identificándolo. La eficiencia del clasificador está directamente relacionada con los datos del vector, es decir, un buen clasificador no podrá clasificar correctamente si las características no discriminan suficientemente bien.

La máxima seguida para el desarrollo de los clasificadores fue la de minimizar el número de falsos negativos incluso cuando la consecuencia era incrementar el número de falsos positivos. Para el sistema es asumible mostrar falsos positivos debido a que, o bien el sistema es semiautomático, es decir, un operador humano filtra los resultados antes de enviar los medios de vigilancia, o bien el sistema es completamente automático, en cuyo caso los sistemas de vigilancia se activarían tras el descubrimiento del *look-alike* con el consecuente gasto

	Var1 PCA	Var2 PCA	Var3 PCA	Var4 PCA	Var5 PCA
APR	-0,0321	0,0466	0,6160	0,1722	-0,1237
elongación	0,1770	0,3460	-0,1469	0,2321	-0,1141
MPR	0,.2596	0,0579	0,3499	0,0017	-0,0160
Rectangularidad	-0,1196	-0,0306	0,2473	0,0149	0,8811
Circularidad	0,3070	-0,.0775	0,0457	-0,0321	-0,2217
Thickness	-0,0013	0,0530	0,6099	0,1626	-0,2103
Hu_1	0,3299	0,0316	-0,0735	0,0802	-0,0472
Hu_2	0,3371	0,0255	-0,0683	0,1573	0,0767
Hu_3	0,3389	-0.0366	-0,0237	0,0670	0,1383
Hu_4	0,2769	0,3158	-0,0167	-0,0164	0,1338
Hu_5	0,2092	0,4218	-0,0049	-0,0343	0,1185
Hu_6	0,2055	0,4349	-0,0000	-0,0842	0,1202
Hu_7	-0,2791	0,3182	-0,0151	0,0583	-0,1003
FS_1	0,2795	-0,2692	0,0373	-0,2175	-0,0562
FS_2	0,0728	-0,2281	-0,1329	0,7528	0,0772
FS_3	-0,2540	0,2239	-0,1053	0,4511	-0,0382
FS_4	-0,2567	0,3457	-0,0020	-0,1540	-0,0864

Tabla 5.7: Coeficientes de los componentes principales seleccionados.

económico pero sin consecuencias medioambientales. Los falsos negativos, por el contrario, no podrían detectarse ni de forma automática ni por un operador ya que el sistema no los mostraría y esto conllevaría un mayor impacto en el ecosistema dado que no se activarían los protocolos de gestión de forma temprana.

Los siguientes apartados describen dos clasificadores desarrollados para identificar los candidatos generados en la segmentación. Para su desarrollo y entrenamiento se ha utilizado una DB de candidatos previamente etiquetados con 155 falsos positivos y 80 sentinazos que fueron distribuidos en 3 grandes grupos (véase la Figura 5.17): el 70 % de los elementos fue reservado para el entrenamiento, un 15 % fue utilizado para el conjunto de validación y el 15 % restante fue seleccionado para el conjunto de test.

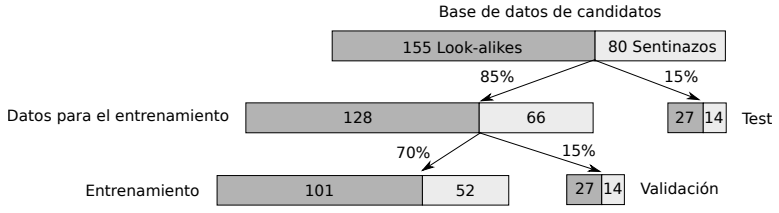

Figura 5.17: DB de candidatos etiquetados utilizados para desarrollar los clasificadores.

5.6.1. Redes Neuronales Artificiales

Introducción

Las ANN tratan de emular el comportamiento del cerebro humano y la extracción de conocimiento genérico a partir de un conjunto de datos [71]. Las ANNs, al igual que el cerebro, están compuestas de procesadores elementales llamados neuronas artificiales o simplemente neuronas que, conectadas entre sí o a entradas externas, cooperan para producir un estímulo de salida. Cuando un conjunto de neuronas recibe todas sus entradas de una misma fuente y todas sus salidas se dirigen a un mismo destino, forman una capa.

Una neurona artificial se caracteriza por los siguientes elementos (véase la Figura 5.18):

- Estado inicial: las neuronas de la red presentan un estado o valor inicial (a_{t-1}) previo a recibir los estímulos externos y que influirá en su respuesta a éstos.
- Conexiones neuronales: valores de entrada a la neurona j-ésima (x_i) con unos pesos asociados (w_{ij}).
- Función de propagación: determina la entrada de la neurona (Net_j). Generalmente es el sumatorio de cada uno de sus estímulos externos o entradas (x_i) multiplicado por su peso asociado (w_{ij}). Es habitual que se incluya un término constante (x_0) denominado sesgo. La inclusión de este elemento se deriva del comportamiento de las neuronas biológicas, que poseen umbrales internos de activación que distorsionan el impacto causado por los estímulos recibidos. El término suele tener el valor '1' y el valor y signo de su peso indica su influencia en la función.
- Función de activación o transferencia: combina el valor inicial de la neurona (a_{t-1}) con la entrada obtenida por la función de propagación (Net_j) para producir un nuevo estado o valor de la neurona.
- Función de salida: transforma el estado de la neurona generado a través de la función de transferencia en una señal de salida (y_j) que se trasmitirá a las siguientes neuronas. Habitualmente, la función de salida coincide con la función identidad, es decir, el valor de salida es, en la práctica, el valor retornado por la función de transferencia.

Una de las características más importantes de las ANNs es su capacidad de generalizar a partir de ejemplos, lo que permite dar una respuesta correcta ante patrones no presentados previamente. Las ANNs tienen la capacidad de aprender creando, modificando o destruyendo

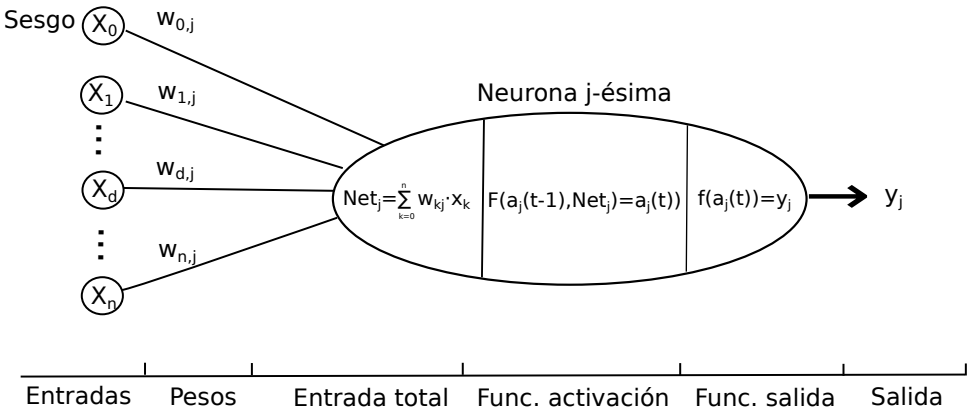

Figura 5.18: Elementos de una neurona artificial. Figura adaptada de [71].

sus conexiones (pesos) entre sus neuronas. Existen diferentes algoritmos de aprendizaje pero todos ellos pueden ser englobados en uno de los dos grandes tipos presentados a continuación:

– Aprendizaje supervisado: se caracteriza por la presencia de un agente que controla el proceso de entrenamiento estableciendo una respuesta correcta para una serie de valores de entrada determinados. Si la salida de la red no es la salida deseada, el supervisor modifica iterativamente los pesos de las neuronas de la red (w_{ij}) con el objetivo de que el resultado tienda al deseado. Éste es el tipo de aprendizaje utilizado para desarrollar la ANN para la detección de los sentinazos y, en concreto, se usó una técnica de aprendizaje por corrección de error denominada backpropagation. Las técnicas por corrección de error ajustan los pesos de las conexiones de la red en base a la diferencia entre la salida deseada y la obtenida. Una de las reglas más sencillas para la corrección es la siguiente:

$$\begin{aligned} w_{ij}^{t+1} &= w_{ij}^{t} + \Delta w_{ij} \\ \Delta w_{ij} &= \alpha(r_j - y_j)x_j \end{aligned} \tag{5.4}$$

Dada una conexión neuronal entre la neurona i-ésima y la j-ésima con un peso w_{ij} en el instante t, Δw_{ij} representaría la variación en el peso de la conexión en el instante $t+1$, es decir, la corrección del error. En el cálculo de Δw_{ij}, α es un factor de aprendizaje, r_j es la salida deseada ante un determinado patrón, mientras que y_j es la salida realmente obtenida y, finalmente, x_j es la entrada a la neurona j.

– Aprendizaje no supervisado: se presenta a la red una serie de patrones sin una respuesta asociada. La red busca particularidades, correlaciones o categorías presentes en los datos de entrada y los categoriza.

Las conexiones neuronales son uno de los elementos fundamentales en el aprendizaje y pueden producirse entre neuronas de una misma capa o entre neuronas de capas diferentes sin limitaciones. El modelo de ANNs más utilizado en la práctica es el Perceptrón Multicapa (MLP), que constituye un modelo particular en el que la propagación de los datos es siempre hacia adelante, es decir, ninguna salida neuronal constituye la entrada a una neurona de la misma capa o de una capa anterior. Las redes MLP (véase la Figura 5.19) están compuestas por una capa neuronas de entrada donde se recogen los datos del exterior, una o varias capas de neuronas ocultas y una capa de salida que devuelve los resultados de la red.

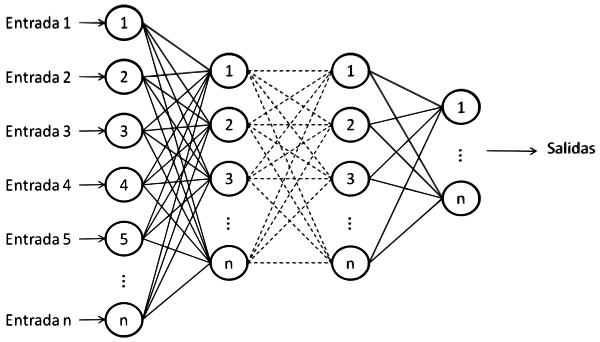

Figura 5.19: Estructura de una red neuronal MLP.

Algoritmo de retropropagación de errores

Debido a su sencillez y eficacia, el algoritmo de retropropagación de errores o de *backpropagation* [58] es uno de los algoritmos de aprendizaje supervisado más extendido para redes MLP. Básicamente, el algoritmo modifica iterativamente los pesos de las conexiones entre las neuronas con el objetivo de minimizar el error de la red. Partiendo de unos pesos aleatorios y de un conjunto de patrones de entrenamiento correctamente clasificados, el algoritmo procesa a través de la red cada patrón. El error cometido al procesar un patrón es cuantificado para modificar los pesos en consecuencia. La modificación iterativa de los pesos se realiza con el objetivo de que el resultado real tienda al deseado. La cuantificación del error medio-total de

la red se calcula como:

$$E(w|x,r) = \frac{1}{2}(r-y)^2 \tag{5.5}$$

, siendo r el resultado deseado para la entrada de un patrón a la red e y el resultado realmente obtenido. La modificación de los pesos trata de minimizar el error de la red, siendo el descenso de gradiente el método más habitual para ello, siempre y cuando la función para calcular el error sea diferenciable. Las variaciones de los pesos utilizando el descenso de gradiente se obtendrían a través del siguiente cálculo:

$$\Delta w_{hj} = -\alpha \frac{\partial E}{\partial w_{hj}} \tag{5.6}$$

El algoritmo de retropropagación de errores tiene dos fases principales: en una primera fase, los patrones de entrenamiento son presentados y analizados por la red y sus salidas son comparadas con los resultados esperados; en una segunda fase se modifican los pesos de la red calculando el error cometido y propagándolo desde la capa de salida hasta la capa de entrada.

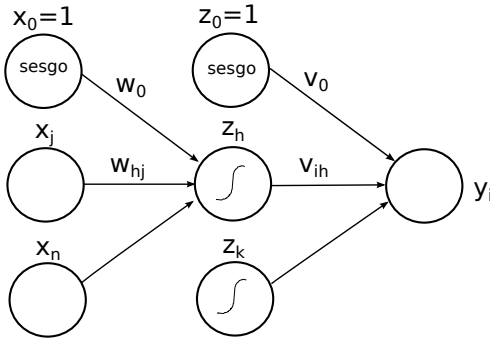

Figura 5.20: Ejemplo simplificado de una red MLP.

Actualización de los pesos Partiendo del ejemplo mostrado en la Figura 5.20 en el que la función de transferencia de las neuronas de la capa oculta es una sigmoide, la salida de la red, para un determinado patrón p, se obtiene a través del siguiente cálculo:

$$y_i^p = \sum_{q=1}^{k} v_{iq} z_q + v_0 \tag{5.7}$$

La variación de los pesos que unen la capa oculta con la capa de salida se calcularía tratando de minimizar el error de la red (véase Ecuación 5.5) a través del descenso de gradiente (véase Ecuación 5.6):

$$\Delta w_{hj} = \alpha \sum_p (r^p - y^p) z_h^p \tag{5.8}$$

Sin embargo, el cálculo de los pesos entre la capa de entrada y la oculta no puede hacerse directamente utilizando el error cuadrático medio, ya que se desconoce cuál es la salida deseada para las neuronas ocultas. La solución está en el uso de la regla de la cadena sobre la Ecuación 5.6 de la siguiente manera:

$$\begin{aligned}
\Delta w_{hj} &= -\alpha \frac{\partial E}{\partial w_{hj}} \\
&= -\alpha \sum_p \frac{\partial E^p}{\partial y^p} \frac{\partial y^p}{\partial z_h^p} \frac{\partial E^p}{\partial z_h^p} \frac{\partial z_h^p}{\partial w_{hj}} \\
&= -\alpha \sum_p \underbrace{-(r^p - y^p)}_{\frac{\partial E^p}{\partial y^p}} \underbrace{v_h}_{\frac{\partial y^p}{\partial z_h^p}} \underbrace{z_h^p (1 - z_h^p) x_j^p}_{\frac{\partial z_h^p}{\partial w_{hj}}} \\
&= \alpha \sum_p (r^p - y^p) v_h z_h^p (1 - z_h^p) x_j^p
\end{aligned} \tag{5.9}$$

Término momento De entre las mejoras desarrolladas sobre el algoritmo clásico cabe destacar la que introduce el término momento. Esta mejora consigue que la actualización de pesos sea proporcional al incremento de la iteración anterior, permitiendo acelerar y estabilizar el proceso de aprendizaje [96]. La actualización de pesos utilizando el término momento se calcula como:

$$W_{t+1} = W_t + [\Delta W_t + \beta \Delta W_{t-1}] \tag{5.10}$$

El término momento, identificado por la constante β, suele tomar un valor entre 0 y 1 y, generalmente, su valor es próximo a la cota superior e inversamente proporcional a la tasa de aprendizaje, es decir, cuanto más pequeño es el valor α más grande suele ser β.

Tanto la tasa de aprendizaje como el término momento son seleccionados empíricamente dependiendo del problema a resolver.

Clasificador

Se implementó una red neuronal MLP para clasificar los candidatos segmentados. Su arquitectura y los pesos de sus conexiones fueron establecidos a través de un proceso de entrenamiento, validación y test utilizando los conjuntos mostrados en la Figura 5.17.

La red desarrollada, cuya arquitectura final se muestra en la Figura 5.22, está compuesta de 3 capas de neuronas. El número de neuronas de la capa de entrada se estableció en 9, una por cada elemento del vector de características, mientras que en la capa de salida se utilizaron 2 neuronas para representar cada una de las posibles clases que se pueden asignar a un candidato. Una neurona de la capa de salida con valor '0' representa la total incompatibilidad con la clase asignada a dicha neurona y el caso opuesto es representado con el valor '1'. Valores intermedios, en el rango (0,1), representan una cierta probabilidad de pertenencia a la clase. Los estímulos externos a la red fueron normalizados en el rango [-1,1] como paso previo a su procesamiento.

Las neuronas utilizadas, tanto en la capa oculta como en la capa de salida, siguen el esquema de la Figura 5.21. Estas neuronas reciben los valores de salida de las neuronas de la capa anterior ponderados a través de los pesos asignados a las conexiones neuronales y un sesgo con valor '1' que se pondera a través de un peso asignado a cada neurona. Una función de propagación, que unifica y combina los valores de entrada en un único valor, sirve como entrada para la función de activación de la neurona. El resultado de la función de activación representa el valor de salida de la neurona.

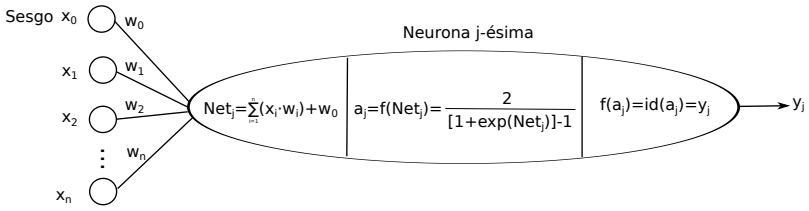

Figura 5.21: Esquema de la neuronas empleadas en la ANN desarrollada.

El número de neuronas de la capa oculta se estableció a través de un proceso comparativo entre diferentes arquitecturas. Utilizando el algoritmo de retropropagación de errores con la variante del término momento como proceso de aprendizaje de la red y basándose en la regla *ad hoc* de que el número de neuronas ocultas no debería ser superior al doble del número de neuronas de entrada [106], fueron realizadas diferentes pruebas en las que se emplearon

hasta un máximo de 18 neuronas. En cada prueba, los patrones normalizados del conjunto de entrenamiento fueron presentados iterativamente a la red y sus salidas se compararon con los resultados esperados. Cuando las salidas no resultaron satisfactorias, los pesos fueron actualizados según el algoritmo de aprendizaje.

Para evitar el sobreentrenamiento de la red, es decir, que la red aprendiese las particularidades propias de los patrones de entrenamiento limitando su capacidad de generalización, se utilizó un conjunto de validación. En cada iteración del proceso de entrenamiento se probó la red con el conjunto de validación y se calculó su eficacia. Por norma general, la eficacia de la red respecto al conjunto de entrenamiento irá mejorando en las sucesivas iteraciones del entrenamiento; sin embargo, la eficacia sobre el conjunto de validación alcanzará un punto de inflexión en el que la tendencia cambiará. La iteración donde se alcance la eficiencia máxima sobre el conjunto de validación marca el final del entrenamiento. Una vez finalizado el entrenamiento, la red entrenada y validada se probó nuevamente sobre el conjunto de test para confirmar su eficacia con un conjunto independiente.

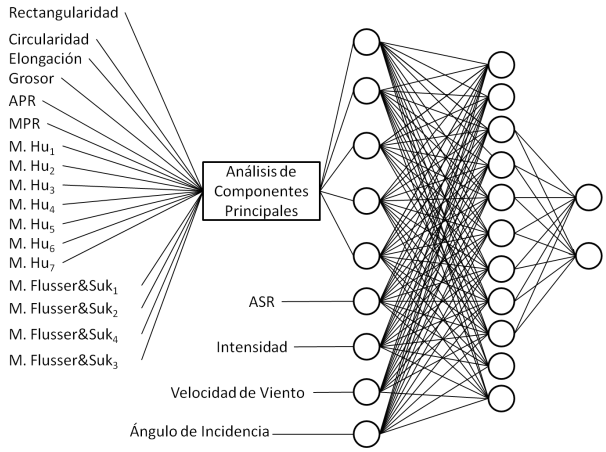

Figura 5.22: Arquitectura del clasificador desarrollado basado en una red neuronal MLP.

Las pruebas de arquitectura fueron realizadas con diferentes inicializaciones de la red, tasas de aprendizaje, valores para el término momento y funciones de transferencia; los mejores resultados se obtuvieron utilizando 11 neuronas en la capa oculta, con una tasa de aprendizaje de 0,3 y un valor de momento de 0,9. La función de propagación utilizada fue la suma ponderada de las entradas de cada neurona con los pesos asociadas a ellas (incluyendo el sesgo),

mientras que la función de transferencia seleccionada fue una sigmoide (tangente hiperbólica). Los resultados de la ejecución de la red sobre los candidatos contenidos en los conjuntos de validación y test, así como su comparativa con el otro clasificador, se muestran en detalle en el capítulo 6.

5.6.2. Árboles de decisión

Descripción

Un árbol de decisión [14] es un método de aprendizaje inductivo, supervisado y no paramétrico que puede ser utilizado tanto para la clasificación como para la regresión. Se implementa a través de una estructura jerárquica basada en una estrategia de divide y vencerás y, generalmente, se representa como un árbol binario, es decir, una estructura de datos formada por nodos en la que cada nodo que no sea final tiene dos hijos. Los nodos finales son conocidos como hojas mientras que los intermedios son denominados nodos decisión. De entre las diferentes implementaciones posibles que se pueden encontrar en la literatura, el clasificador desarrollado se basó en los Árboles de Clasificación y Regresión (CART), propuestos por Breiman et ál. [24].

Construcción y entrenamiento del árbol

El entrenamiento y construcción de un árbol de clasificación se realiza a partir de un conjunto de patrones de entrenamiento, previamente etiquetados, en el que se trata de encontrar, a través de sucesivas particiones, secciones en su espacio de representación en las que todos los patrones presentes pertenezcan a una misma clase o etiqueta. Cada partición se representa en el árbol como un nodo decisión en el que se evalúa una característica presente en los patrones, agrupándose éstos acorde al resultado de la evaluación en uno de los nodos hijo generados.

Sea m un nodo del árbol y N_m los patrones o instancias de entrenamiento que llegan a ese nodo desde el conjunto inicial N contenido en el nodo raíz; se considera N_m^i un subconjunto de N_m que contiene patrones asociados a la clase C_i donde

$$\sum_i N_m^i = N_m \tag{5.11}$$

Dado un patrón que alcanza el nodo m, la probabilidad de que pertenezca a la clase C_i se puede calcular como:

$$p_m^i = \frac{N_m^i}{N_m} \tag{5.12}$$

Un nodo se considera puro cuando todos los patrones que llegan a ese nodo pertenecen a una misma clase, lo cual se expresa matemáticamente de la siguiente manera:

$$m := puro \Longleftrightarrow \forall_i, p_m^i \in \{0,1\} \tag{5.13}$$

Los nodos puros representan hojas del árbol y se etiquetan con el valor de la clase asignada a sus patrones. Si los dos nodos hijo de una partición son puros, dicha partición se considera pura.

Aunque lo ideal es realizar particiones puras, la mayoría no lo son y las que lo son se sitúan en la parte inferior del árbol (ramas inferiores). Es necesario establecer un método para seleccionar la mejor partición en cada punto del árbol y, para ello, se debe cuantificar la impureza de las posibles particiones y seleccionar la que tenga menor valor. Existen diferentes métodos de cuantificar la impureza de una partición, siendo el método *Gini* [24] uno de los más populares y el empleado para desarrollar el clasificador de SENTINAZOS. El valor de *Gini* se calcula como:

$$1 - \sum_i (p_m^i)^2 \tag{5.14}$$

En la Figura 5.23 se muestra un sencillo ejemplo de construcción de un árbol de decisión binario. El conjunto de entrenamiento está formado 20 patrones pertenecientes a 2 tipos diferentes de clases diferentes y cada patrón contiene medidas para 2 características. Se observa que no todas las hojas del ejemplo son puras ya que, aunque es posible crear un árbol suficientemente grande donde todas sus hojas sean puras, lo más probable es que se caiga en el sobreentrenamiento; por tanto, es necesario utilizar reglas que gestionen el tamaño del árbol.

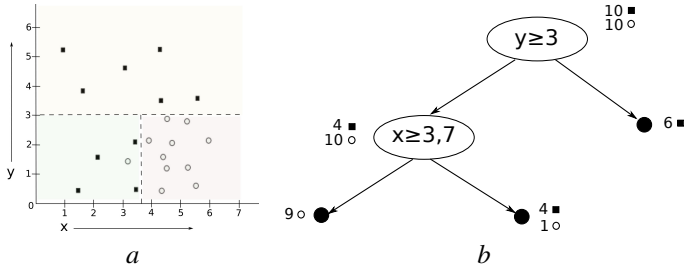

Figura 5.23: Ejemplo de desarrollo de un árbol de decisión binario: a) Patrones de entrenamiento en base a dos características. b) Árbol desarrollado a partir de los patrones de entrenamiento.

Poda

El tamaño del árbol generado puede establecerse desde dos estrategias diferentes: por un lado, se puede definir a priori un umbral que establezca el número de patrones mínimo que debe alcanzar un nodo para que se permita crear una nueva partición, lo cual implica que el árbol dejará de crecer en el momento en que los patrones sean inferiores a ese umbral. Otra estrategia consiste en desarrollar completamente el árbol hasta conseguir que todas sus hojas sean puras o hasta que los patrones asociados sean inferiores a un umbral y a continuación podarlo.

El proceso de poda trata de encontrar las secciones del árbol (subárboles) que no aportan ventajas y se asocian al sobreentrenamiento. Existen diferentes estrategias de poda y, en el caso concreto del clasificador desarrollado, la estrategia utilizada fue la de seleccionar la poda que mejor clasificaba un conjunto de validación. Utilizando un grupo independiente de patrones o conjunto de validación y, de forma incremental desde las hojas hacia el nodo raíz, se sustituyó cada posible subárbol por una hoja etiquetada con la clase asociada a la mayoría de los patrones de entrenamiento que clasificaba ese subárbol. Si la eficiencia del nuevo árbol no era peor que la del original, la poda se llevaba a cabo pues la mayor complejidad no se justificaba.

La estrategia de poda obtiene mejores resultados que la estrategia basada en detener el crecimiento del árbol, ya que la poda permite que un subárbol de un nodo permanezca mientras que su «hermano» desaparece y esto sería imposible con una estrategia de parada, ya que el crecimiento se detendría por ambas ramas. Por otro lado, la estrategia de poda es más lenta y compleja, ya que es necesario analizar los diferentes subárboles de los que está compuesto el árbol original y encontrar la mejor poda.

Clasificador

Se desarrolló un clasificador basado en un árbol de decisión binario para clasificar los candidatos resultantes de la fase de segmentación. Al igual que para la ANN presentada en la Sección 5.6.1, para el entrenamiento y desarrollo del clasificador se utilizaron 3 conjuntos siguiendo el mismo esquema de la Figura 5.17: el 70 % de los elementos fueron utilizados para entrenar el árbol, un 15 % fue utilizado para podarlo (conjunto de evaluación) y el 15 % restante fue empleado en el test del clasificador resultante. El método *Gini* ha sido el empleado para cuantificar la impureza de las particiones en el proceso de desarrollo del árbol.

El árbol con hojas puras y generado únicamente a través del entrenamiento se muestra en la Figura 5.24; se puede observar que el tamaño del árbol es considerable y se le presupone un sobreentrenamiento. A través de un proceso de poda, en el que se calculó la eficacia de los diferentes subárboles sobre el conjunto de validación, se estableció que el mejor clasificador era el subárbol presentado en la Figura 5.25. Para confirmar su eficacia se utilizó el conjunto de test. Los resultados del clasificador sobre los candidatos contenidos tanto en el conjunto de test como en el de validación, se presentan en el Capítulo 6.

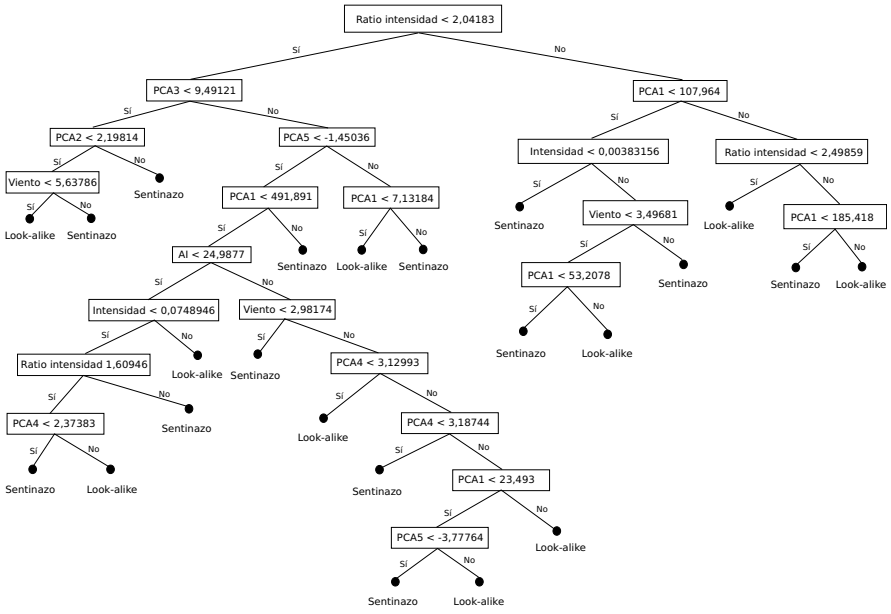

Figura 5.24: Árbol de decisión binario, sin podar y con hojas puras, generado por el proceso de entrenamiento.

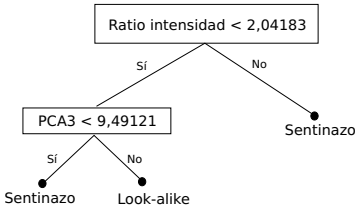

Figura 5.25: Árbol de decisión binario podado.

5.7. Validación del sistema de detección de vertidos

5.7.1. Desarrollo de un software de escritorio

La necesidad de evaluar los algoritmos presentados en las secciones anteriores condujo al desarrollo de un software de escritorio. La implementación del prototipo se basó en software de código libre y abierto con el objetivo de que pudiese ser libremente distribuido y así promover la colaboración entre la comunidad científica.

Visualización

Para facilitar el trabajo con los datos SAR, se implementó un visor específico que permitiese visualizar e interactuar (navegar, aumentar y reducir) con las imágenes, así como representarlas sobre un mapa de contexto mundial para mejorar la interpretación de los resultados. El visor (véase la Figura 5.26) también se diseñó para acceder a los metadatos contenidos en las cabeceras de las imágenes.

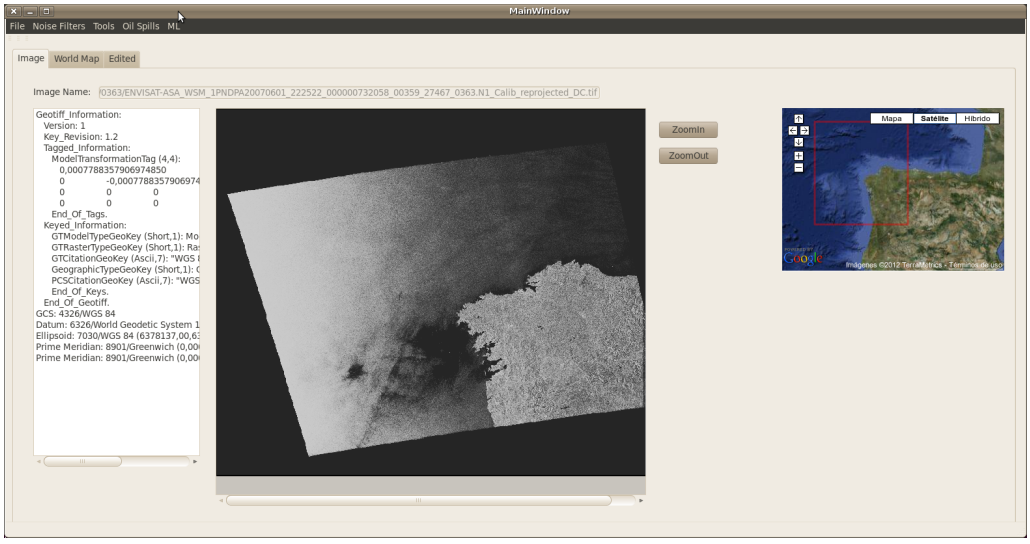

Figura 5.26: Pantalla principal del software de escritorio.

Identificación de sentinazos

El prototipo implementado contiene las diferentes funciones y métodos de detección de sentinazos presentados en los apartados anteriores. Existen dos formas de ejecutarlo: por un lado, automáticamente a través de los parámetros, filtros y funciones por defecto; por otro lado, haciendo uso de una personalización que permite configurar y/o modificar el procesado de la imagen:

- Filtro de ruido: la aplicación dispone de un filtro de mediana, un filtro de Gauss y un filtro bilateral.
- Filtro de área: el usuario puede seleccionar el valor máximo del filtro del área que se aplicará a la imagen.
- Clasificadores: la aplicación permite emplear los diferentes clasificadores a través de su configuración por defecto o cargando una nueva base de conocimiento.

El resultado de cada acción ejecutada sobre la imagen es mostrado en el visor, lo que permite evaluar sus efectos. La Figura 5.27 muestra un ejemplo del resultado obtenido tras procesar una imagen SAR empleando los parámetros por defecto: máscara de tierra (véase la Sección 5.3.2), filtro de mediana de 3x3 (véase la Sección 5.3.3), segmentación por umbralización adaptativa en base a los vientos (véase la Sección 5.4.1), filtro de vientos menores a 3 m/s (véase la Sección 5.4.3), filtro de áreas inferiores a 0,3 Km^2 (véase la Sección 5.4.4) y clasificación empleando la ANN (véase la Sección 5.6.1).

5.7.2. Integración en RETELAB

Una vez finalizado y validado el sistema de detección de vertidos de hidrocarburos se procedió a su integración en RETELAB. Al contrario que el prototipo de escritorio que permitía ir visualizando el efecto de ejecutar diferentes acciones sobre los datos SAR, la versión Grid de SENTINAZOS fue pensada para ser ejecutada en modo no interactivo, de forma que el usuario debe seleccionar en un primer momento los parámetros de la tarea y a continuación procesarla y esperar a los resultados finales.

A pesar de la limitación en cuanto a la interacción, las ventajas que proporciona la computación Grid al sistema SENTINAZOS son importantes:

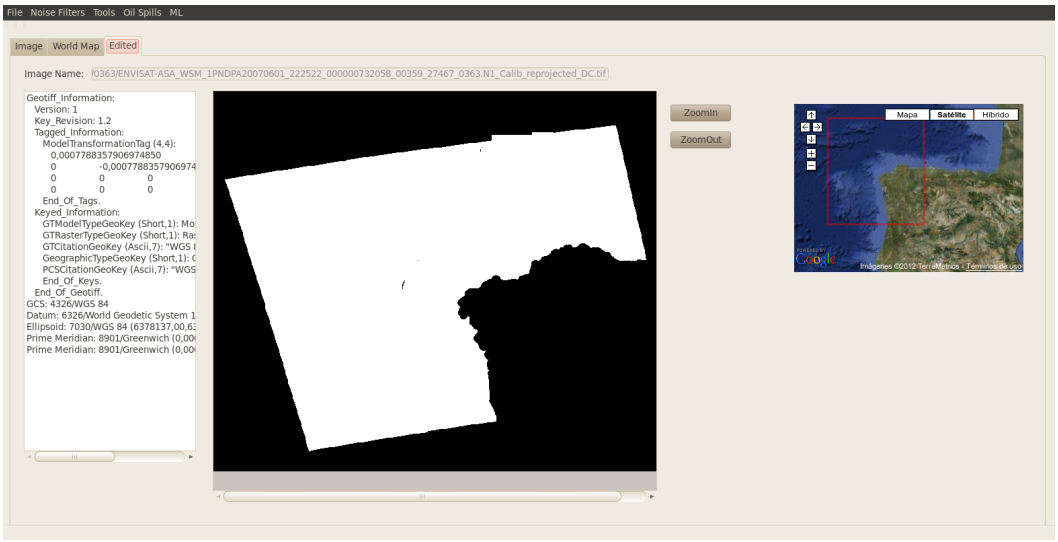

Figura 5.27: Captura de pantalla del resultado de procesar una imagen SAR a través de los parámetros por defecto y un clasificador ANN.

– Permite compartir los datos (vientos, datos SAR, etc.) entre los diferentes centros participantes.

– Los resultados obtenidos pueden ser distribuidos y utilizados para nuevos estudios por el resto de los participantes de la red: por ejemplo, para realizar predicciones de la evolución de los vertidos o para ejecutar algoritmos de *backtracking* con el objetivo de detectar su fuente.

– El entorno de RETELAB permite ejecutar diferentes instancias de SENTINAZOS y así procesar diferentes áreas simultáneamente.

– La presencia de recursos computacionales de gran potencia puede aportar una mayor velocidad de procesamiento.

El portlet para la detección de sentinazos permite seleccionar diferentes acciones a realizar sobre los datos SAR como, por ejemplo, segmentar, clasificar una imagen previamente segmentada o segmentar y clasificar. Para la clasificación se permite escoger uno de los dos clasificadores implementados (ANN o árboles de decisión) y los parámetros escogidos tanto para los filtros (ruido, área, vientos) como para los clasificadores, están prefijados.

Al igual que el resto de los portlets, éste ha sido integrado con el Data Grid (véase la Sección 3.3) tanto para poder acceder y utilizar los datos de la DB virtual como para compartir los resultados generados. Además, el portlet también se ha integrado con el monitor de tareas para realizar un seguimiento del estado de las tareas y acceder a los resultados que generen.

La Figura 5.28 muestra el portlet desarrollado para SENTINAZOS y su integración en el laboratorio de RETELAB.

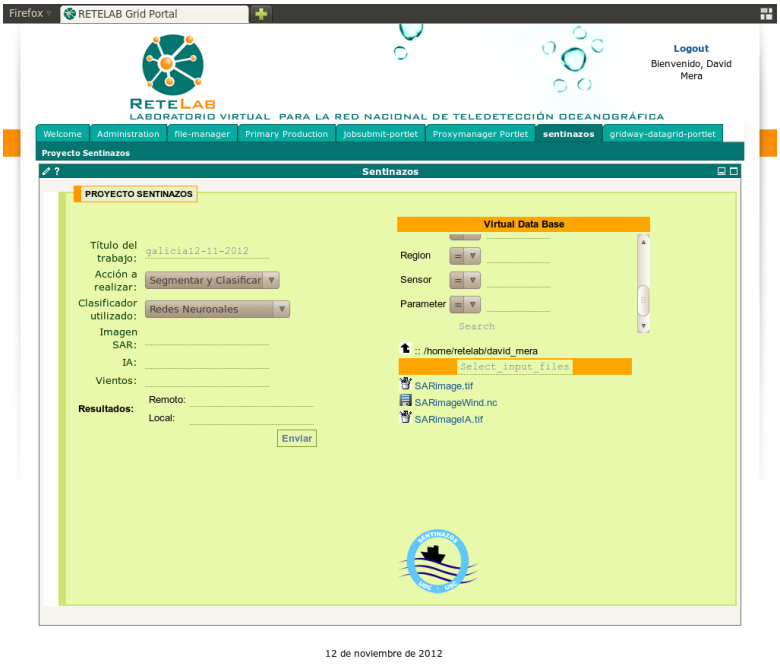

Figura 5.28: Captura de pantalla del portlet desarrollado para integrar SENTINAZOS en el laboratorio virtual.

CAPÍTULO 6

RESULTADOS

6.1. Eficacia

Los clasificadores presentados en las secciones 5.6.1 y 5.6.2 han sido desarrollados utilizando la DB de candidatos presentada en la Figura 5.17, es decir, se han utilizado los mismos conjuntos para entrenar, validar y probar los clasificadores, lo que permitió realizar una comparativa entre ellos en términos de eficacia (véase Tabla 6.1). Los datos relativos al rendimiento del árbol de decisión original (sin podar) son meramente ilustrativos, ya que reflejan cómo el sobreentrenamiento afecta a la clasificación de otros conjuntos que no sean el de entrenamiento. En la práctica, sólo el árbol podado y la red MLP han sido utilizados como clasificadores.

	Validación		Test	
	Sentinazos	Look-alikes	Sentinazos	Look-alikes
Red MLP	85,7 %	85,2 %	92,9 %	96,3 %
Árbol decisión original	50,0 %	88,9 %	35,7 %	88,9 %
Árbol decisión podado	92,9 %	85,2 %	92,9 %	92,6 %

Tabla 6.1: Eficacia de los diferentes clasificadores sobre los conjuntos de validación y test.

La eficacia de los clasificadores fue comprobada directamente sobre las imágenes SAR al ser éstos implementados como parte del prototipo descrito en la sección 5.7.1. La Figura 6.1 muestra una composición, realizada sólo con el objeto de facilitar la visualización, de 2 imágenes SAR (Figura 6.1a) y su correspondiente segmentación (Figura 6.1b), así como los resultados obtenidos por los clasificadores (Figura 6.1c y Figura 6.1d). Enmarcados en

color rojo se identifican varios sentinazos, confirmados a través de misiones de vigilancia de SASEMAR, que se detallan en la Figura 6.2 junto al resultado de clasificarlos mediante la red MLP.

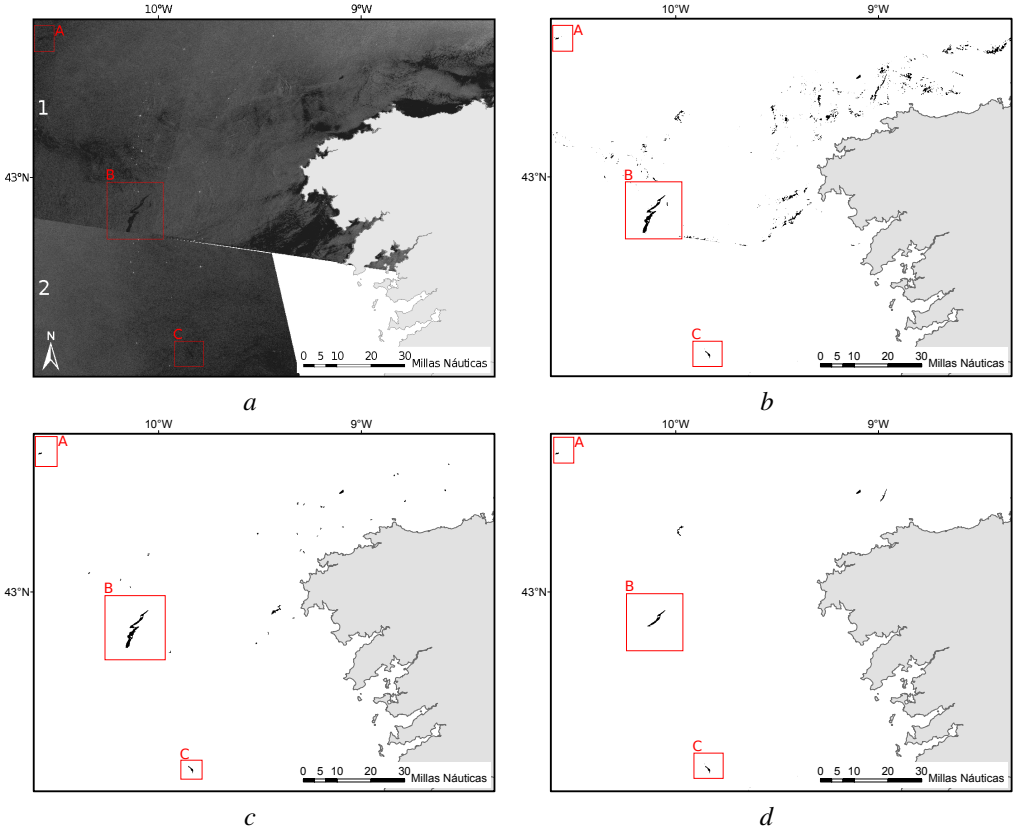

Figura 6.1: a) Composición, a efectos de visualización, de 2 imágenes SAR. La imagen '1' fue obtenida el 28/03/2011 y la imagen '2' el 14/10/2011. b) Composición formada por la segmentación de las imágenes SAR. c) Montaje formado por los resultados de aplicar el clasificador basado en el árbol de decisión a las imágenes segmentadas. d) Montaje formado por los resultados de aplicar el clasificador basado en la red MLP a las imágenes segmentadas.

Se puede observar que los resultados obtenidos a través del clasificador basado en el árbol de decisión son muy similares a los ofrecidos por el que utiliza la red MLP (sobre todo desde el punto de vista de la detección de sentinazos), si bien la ANN (véase la Figura 6.1d) genera una menor cantidad de falsos positivos.

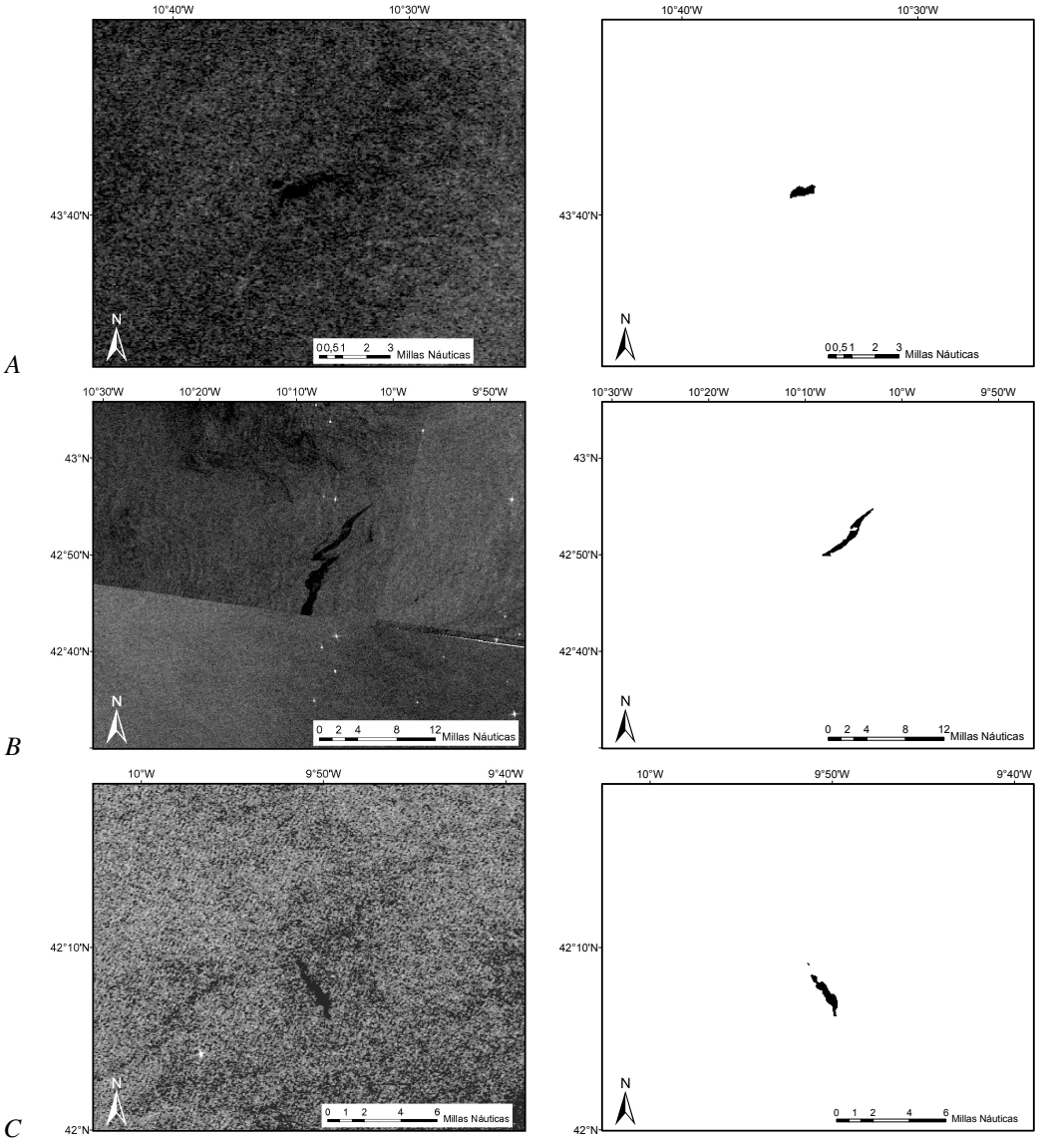

Figura 6.2: A la izquierda, detalle de los sentinazos confirmados e identificados en la Figura 6.1; a la derecha, resultados obtenidos a través del clasificador basado en una red MLP.

6.2. Tiempo de procesamiento

El tiempo requerido para el procesamiento de las imágenes en la búsqueda de vertidos de hidrocarburos es crítico y, por ello, los algoritmos presentados en las secciones anteriores han sido desarrollados con el objetivo de minimizarlo pero sin sacrificar su eficacia, sobre todo en cuanto a la no detección de sentinazos (falsos negativos). Además de tener un especial cuidado en el diseño e implementación del software, el código se desarrolló pensando en obtener el mejor rendimiento de los microprocesadores multinúcleo presentes en la mayoría de las computadoras actuales. Para la implementación se empleó el API de OpenMP [36], que permite adaptar y paralelizar automáticamente la ejecución del código entre los diferentes núcleos de los que disponga el procesador donde se ejecuta la aplicación y así obtener el máximo rendimiento.

Fueron probadas dos versiones del algoritmo (clásica y optimizada) utilizando un nodo de computación Core 2 Duo - 2,13 Ghz con 3 GB de RAM y, después de 100 iteraciones sobre una imagen de 8.088 x 6.481 píxeles, la versión optimizada empleó una media de 19,4 segundos por ejecución, mejorando un 29,6 % el tiempo de ejecución del algoritmo clásico.

Capítulo 7

Discusión

Debido a las condiciones, tanto geográficas como climatológicas, de la costa gallega (área de estudio), los vientos de baja intensidad generan la mayor parte de los falsos positivos o *look-alikes*. El algoritmo desarrollado los filtra, en su mayor parte, en la fase de segmentación al introducir una umbralización basada en la velocidad del viento. Una de las consecuencias de este proceso de segmentación es que algunos vertidos pueden quedar parcialmente ocultos en la salida si se encuentran situados en áreas afectadas por vientos de baja intensidad. Modificar el umbral de viento mínimo mejora la segmentación en algunos de los casos pero a costa de incrementar notablemente el número de falsos positivos. Es muy complicado detectar sentinazos en áreas con vientos de baja intensidad debido a que la superficie del mar no genera el brillo suficiente en la imagen SAR para que se produzca un contraste entre la superficie afectada por el sentinazo y el mar limpio y, además, la intensidad de la señal retrodispersada que llega al satélite se aproxima al ruido de fondo (véase la Sección 5.2.1) dificultando discernir los fenómenos presentes en la superficie.

A pesar de que la utilización de los datos de viento mejoró considerablemente la segmentación, ésta podría ser más precisa si se utilizasen medidas reales o provenientes de modelos de vientos más precisos que el CMOD5 que trabaja directamente con los datos SAR. El modelo CMOD5 es bastante exacto cuando se trata de velocidades de viento inferiores a 15 m/s y empeora con vientos de mayor intensidad [62]. El error producido por el modelo, aunque sea pequeño, podría ser clave para que una zona sea identificada como *look-alike*, sobre todo si no alcanza el umbral mínimo de viento.

Otro de los inconvenientes que se presentan al emplear un modelo basado en los propios datos SAR es la dificultad para calcular los datos de viento en las áreas próximas a la costa. CMOD5 sólo calcula la velocidad del viento en la superficie oceánica y necesita una «ventana» de datos alrededor del área de estudio, por lo que la presencia de la costa puede distorsionar los resultados. La propia costa también genera muchos falsos positivos al proteger del viento a la superficie marina generando áreas de baja intensidad. Para evitar estos inconvenientes se ha usado una máscara de tierra extendida (véase la Sección 5.3.2) pero los resultados siguen mostrando una mayor cantidad de *look-alikes* en las áreas más cercanas a la costa. Aumentar la máscara para reducir el efecto de la costa obtendría mejores resultados desde el punto de vista de los *look-alikes* pero a costa de ocultar posibles sentinazos que se hayan desplazado y que puedan ser potencialmente peligrosos para el medioambiente.

La comparativa entre los diferentes sistemas de detección de sentinazos presentes en la literatura es especialmente complicada debido a que cada uno utiliza su propia DB para entrenar y para validar, se basa en diferentes tipos y resoluciones de imágenes SAR, usa diferentes vectores de características, etc. Además, es habitual que las DB contengan imágenes con vertidos que solamente han sido identificados por expertos y directamente sobre la propia imagen pero sin confirmación *in situ*. Los porcentajes de eficacia de los diferentes sistemas deben ser tomados a modo de referencia pero de ellos no puede extrapolarse ninguna conclusión definitiva. Algunos de los sistemas de detección de hidrocarburos presentes en la literatura, así como sus porcentajes de eficacia, son mostrados en la Tabla 7.

Ref.	DB	Clasificador	Eficacia
[37]	71 Sentinazos, 68 Look-alikes	Red MLP (validación cruzada)	82 % Look-alikes, 90 % Sentinazos
[108]	34 Sentinazos, 45 Look-alikes	Red MLP	91 % Look-alikes, 87 % Sentinazos
[103]	71 Sentinazos, 6.980 Look-alikes	Modelo estadístico + Sist. basado en reglas	99 % Look-alikes, 94 % Sentinazos
[104]	37 Sentinazos, 12.110 Look-alikes	Modelo estadístico + Sist. basado en reglas	99,4 % Look-alikes, 78,4 % Sentinazos
[26]	41 Sentinazos 12.245, Look-alikes	Modelo estadístico + Sist. basado en reglas + Estimador de confianza	89,7 % Look-alikes, 92,07 % Sentinazos (sin aplicar el estimador de confianza)
[41]	11 Sentinazos, 6 Look-alikes, 4 Desconocidos	Distancia de Mahalanobis	100 % Look-alikes, 82 % Sentinazos, 0 % Desconocidos
		Clasificador probabilístico	67 % Look-alikes, 90,9 % Sentinazos, 50 % Desconocidos
[54]	1.915 píxeles (552 Sentinazos, 557 Bajo viento, 806 Superficie limpia)	Red MLP	98,25 % Look-alikes, 97,02 % Sentinazos

Tabla 7.1: Diferentes sistemas de detección de vertidos y su porcentaje de eficacia en la clasificación.

Los resultados de los clasificadores implementados (véase Tabla 6.1) son muy prometedores a la vista de los porcentajes de eficacia obtenidos por los clasificadores presentes en la literatura. No obstante, es sorprendente que un clasificador tan sencillo como el árbol de decisión binario obtenga unos resultados comparables a la mayoría de clasificadores siendo éstos, en general, mucho más complejos. Esto puede ser debido a varios factores: por un lado, la mayoría de los *look-alikes* son filtrados, gracias al empleo de datos meteorológicos, en la fase de segmentación; por otro lado, la mayoría de los candidatos segmentados pueden ser clasificados basándose sólo en una de las variables del vector (ratio entre la intensidad de los píxeles de la mancha respecto al área circundante), mientras que el uso de las otras variables del vector permitirá discernir los casos más problemáticos, siendo esto lo que marca la diferencia entre la red neuronal implementada y el árbol.

La mayoría de los sistemas dan sus porcentajes de eficacia en función de la clasificación de los candidatos pero, en otros casos, como Óscar Garcia-Pineda et ál. [54], se presenta la eficacia en función de la correcta clasificación a nivel de píxel. En general, se espera una eficacia mayor a nivel de píxel dado que las DB suelen estar formadas por un mayor número de representantes y los errores, si son pocos, quedan más diluidos en los porcentajes. Además, si se analizan imágenes completas, la cantidad de píxeles que representan hidrocarburos es claramente inferior a la que representa superficie limpia, por lo que se espera un porcentaje de eficacia media muy alto de píxeles correctamente etiquetados.

El tiempo de procesamiento es crítico en un sistema que debe dar respuesta en tiempo cuasi real pero la mayoría de los sistemas de detección de hidrocarburos presentes en la literatura lo relegan a un segundo plano en pos de la eficacia. Sin embargo, una rápida detección puede marcar la diferencia a la hora de ser efectivos gestionando un vertido o incluso en la búsqueda de los responsables. Teniendo en cuenta que, en la práctica, la mayoría de los sistemas funcionan de forma semiautomática, es decir, que un operador suele revisar los resultados antes de enviar los medios de vigilancia, se hace más plausible que el sistema se desarrolle con el objetivo de detectar sentinazos en el intervalo de tiempo más breve posible incluso si para ello se sacrifica eficacia en la eliminación de los falsos positivos o *look-alikes*. El sistema presentado, con una media de 19,4 s de tiempo de procesamiento, representa una importante mejora respecto a otros sistemas como: el TCNNA (65 min), el cual se aplica a imágenes completas; Solberg et ál. [104] (1,45 min), que utiliza factores de escape en su algoritmo para no procesar completamente la imagen, es decir, no utiliza todos los datos con el consecuente riesgo de no conseguir los mejores resultados; o Topouzelis et ál. [108] (2-5 min), que utiliza

subsecciones de la imagen de entre 4 y 16 MB previamente seleccionadas por un operador. En otros casos, se opta por reducir la resolución radiométrica y trabajar sólo con 256 niveles de grises [90] [54], lo que conlleva un incremento en la velocidad de procesamiento de las imágenes pero a costa de perder información.

Es importante tener en cuenta que los datos presentados en cuanto al tiempo de procesamiento están íntimamente relacionados con las características de la computadora utilizada, es decir, que podrían ser fácilmente mejorados disponiendo de un hardware con mayor potencia y, aunque esto pueda parecer obvio para cualquier software, en el caso presentado cobra mayor relevancia puesto que fue desarrollado para adaptarse automáticamente y utilizar los diferentes núcleos disponibles.

Epílogo

Capítulo 8

Resultados y conclusiones

En este trabajo se ha planteado el desarrollo, despliegue y validación de un laboratorio virtual oceanográfico. Partiendo de unos requisitos iniciales ambiciosos en los que se abordaron diferentes líneas de investigación, se desarrolló un entorno de trabajo basado en computación Grid enfocado a la ejecución de aplicaciones de oceanografía. Para validar el entorno se desarrolló un sistema semiautomático de detección de vertidos de hidrocarburos en la superficie oceánica a través de datos de satélite.

Las principales aportaciones del presente trabajo han sido:

- **Desarrollo de un laboratorio virtual de altas prestaciones**: se ha presentado un entorno de trabajo unificado y versátil que permite a los investigadores realizar una gran parte de su trabajo sin necesidad de otras herramientas. Como si de un laboratorio real se tratase, el entorno permite:
  - Administrar el acceso de los usuarios.
  - Gestionar, analizar y visualizar datos.
  - Procesar datos utilizando recursos computacionales de altas prestaciones.
  - Distribuir y compartir los resultados de los proyectos con la comunidad.
- **Desarrollo de una interfaz amigable**: tanto la arquitectura del sistema como su interfaz han sido diseñadas para abstraer a los usuarios finales de la tecnología subyacente. Internet es un entorno cómodo y familiar para la mayor parte de los usuarios y, por ello, se escogió un portal Web como medio de acceso al sistema Grid. El portal fue

implementado empleando GridSphere, que proporciona un entorno de desarrollo de portales Web basado en portlets. El uso de portlets permite desplegar aplicaciones independientes del *middleware* Grid utilizado en el sistema, así como reutilizarlas en otros portales. La solución tecnológica desarrollada permite al usuario centrar sus esfuerzos en el desarrollo de sus trabajos y no en el aprendizaje de las herramientas.

- **Sistema de autenticación y autorización simplificado**: buscando una solución bien equilibrada entre seguridad y facilidad de uso, se diseñó y desarrolló un sistema de autenticación y autorización integrando y mejorando diferentes tecnologías. El sistema de registros de usuarios PURSe fue desplegado para abstraer a los usuarios del uso de los DCs. Tanto PURSe como GridSphere fueron modificados para gestionar roles de usuarios a través de ACs y así poder emplearlos como medio de autorización en el Grid. El uso de un sistema de autorización del tipo RBAC en un Grid simplifica la administración de los recursos y, por esta razón, en RETELAB se integró el software PERMIS que permite gestionar políticas de seguridad basadas en roles.

- **Despliegue de un sistema de autenticación SSO**: los usuarios que pertenezcan a centros incluidos en la red de confianza de RETELAB pueden acceder al portal sin necesidad de registrarse, a través del sistema de autenticación de sus centros. Para realizar este proceso se integró Shibboleth y se desarrollaron procedimientos que generan, de forma temporal, DCs y ACs para los usuarios externos. Los datos necesarios para los diferentes certificados son proporcionados por las organizaciones en el proceso de autenticación.

- **Despliegue de un sistema de almacenamiento distribuido basado en metadatos**: se ha diseñado y desplegado un sistema de almacenamiento distribuido que permite a la comunidad de RETELAB compartir los datos de sus investigaciones. Esta DB virtual, que soporta formatos estándar en oceanografía, está dirigida por metadatos para facilitar el etiquetado y descripción de los datos almacenados y así mejorar los sistemas de búsquedas. Para unificar el uso de los metadatos se ha seleccionado el estándar para describir información geográfica ISO 19115.

- **Despliegue de un metaplanificador y un monitor de trabajos**: se ha diseñado una interfaz de ejecución y monitorización de trabajos que permite a los usuarios configurar, ejecutar y monitorizar sus trabajos sin que sea necesario que conozcan la infraestructura hardware. Para ello, se ha desplegado una versión mejorada del metaplanificador

GridWay que se encarga de gestionar, en nombre del usuario, la ejecución de las tareas. El monitor permite visualizar el estado de las tareas y acceder a los resultados una vez que hayan finalizado. Tanto la versión del GridWay utilizada como la interfaz de ejecución y monitorización de los trabajos ha sido realizada por el CESGA en el marco del proyecto RETELAB.

– **Validación y distribución de los resultados**: utilizando la interfaz de monitorización, el usuario puede acceder a los resultados de las tareas ejecutadas. Desde la interfaz de resultados, el usuario puede descargar los ficheros generados o compartirlos con la comunidad, publicándolos, previa descripción a través de metadatos, en la DB virtual, lo que realimenta el sistema. El entorno cuenta con una herramienta avanzada para la visualización y validación de los datos llamada IDV (desarrollada por Unidata). Este software se descarga e instala en el puesto de trabajo del usuario de forma temporal y transparente y se ejecuta con los datos seleccionados por el usuario.

– **Desarrollo de un sistema de detección de vertidos de hidrocarburos**: el laboratorio virtual se ha validado a través del despliegue de varias aplicaciones y, entre ellas, por su entidad, alcance y necesidad de investigación adicional del estado del arte, destaca SENTINAZOS. Éste es un sistema de detección de vertidos de hidrocarburos en la superficie oceánica que fue desarrollado con el doble propósito de validar el sistema y de encontrar una solución para un problema diario de nuestras costas. Las principales aportaciones de este sistema son:

  • **Desarrollo de un procedimiento de segmentación adaptativo basado en la velocidad del viento**: partiendo de una DB de imágenes SAR enriquecida, tanto con datos de viento como con información de vertidos de hidrocarburos, se implementó un segmentador basado en una umbralización adaptativa. El establecimiento de este umbral tiene como base el estudio de centenares de datos muestreados en los que se analizó la velocidad del viento, el IA del satélite respecto al punto muestreado y su valor de intensidad. Tras el estudio de la muestra, se constató que si el IA era similar, las muestras asociadas a hidrocarburos tenían valores de intensidad menores a las que representaban superficie limpia pero, además, se comprobó que estos valores tenían una relación directa con el viento que afectaba a la zona. Empleando estos datos, se estableció una función definida por partes que representaba el umbral de intensidad del hidrocarburo según el IA y la velocidad del

viento. Este procedimiento de segmentación obtiene un menor número de falsos positivos que otras alternativas, lo que permite facilitar el trabajo del clasificador y obtener mejores resultados.

- **Caracterización de los candidatos a través de la forma**: diferentes estudios han constatado que los sentinazos adquieren formas que siguen patrones singulares debido a su origen y a su antigüedad. Partiendo de la hipótesis de que la forma es un elemento diferenciador, se desarrolló un procedimiento para caracterizar los candidatos que generaba un vector dominado por las características relacionadas con la forma. La dimensionalidad de dicho vector fue reducida a través de un PCA, siendo el resultado utilizado como entrada para el clasificador.
- **Sistema semiautomático de detección de vertidos**: se desarrollaron y analizaron dos clasificadores para separar los falsos positivos de los sentinazos. Tanto la ANN como el árbol de decisión obtuvieron resultados muy prometedores. El uso de un segmentador optimizado permite emplear clasificadores más sencillos, por lo que se reduce el tiempo de procesamiento sin sacrificar la tasa de aciertos.
- **Disminución del tiempo de procesamiento**: empleando un paradigma de programación paralela con memoria compartida, se ha implementado un sistema que permite procesar completamente un producto SAR en un tiempo considerablemente inferior al usado por aplicaciones similares encontradas en la literatura. Esta mejora en el tiempo de procesamiento permite emplear el sistema de detección como una herramienta de ayuda a la toma de decisiones en tiempo casi real.

Capítulo 9

Trabajo futuro

9.1. Trabajo futuro en el proyecto RETELAB

El futuro del laboratorio virtual puede verse desde dos puntos de vista: por un lado, la evolución del propio entorno, que se centrará en el análisis de los posibles puntos de confluencia entre la computación Grid y la Cloud. Por otro, el del desarrollo y mejora de las aplicaciones desplegadas, en particular, la mejora de los algoritmos para la detección de vertidos, así como la adaptación de éstos para trabajar con datos provenientes de fuentes diferentes al Envisat, como por ejemplo el futuro satélite Sentinel-1.

9.1.1. Computación Cloud

La computación Cloud es un nuevo paradigma de computación distribuida que ha surgido con fuerza estos últimos años. Al igual que ocurrió con la computación Grid, su definición ha sido objeto de debate tanto en el mundo empresarial como entre la comunidad científica. La literatura presenta multitud de definiciones pero en general se observa que las que provienen del mundo empresarial se centran en la perspectiva del usuario final y su experiencia de uso, mientras que desde la comunidad científica se incluyen también aspectos relativos a su arquitectura. Foster et ál. [48] definen la computación Cloud como:

«*Un paradigma de computación distribuida de gran envergadura dirigido por economías de escala en el que un conjunto administrado de recursos (procesamiento, almacenamiento, entornos y aplicaciones) abstractos, virtualizados y dinámicamente escalables se proporcionan, bajo demanda, a clientes vía Internet.*»

La computación Cloud es un paradigma de computación distribuida altamente especializado que presenta las siguientes diferencias respecto a anteriores paradigmas [48]:

- Es altamente escalable.
- Puede ser encapsulado como una entidad abstracta y proporcionar diferentes niveles de servicios a los clientes finales.
- Está dirigido por economías de escala.
- Los servicios son proporcionados bajo demanda y pueden ser configurados dinámicamente a través de la virtualización.

Su arquitectura puede dividirse en capas [48] y éstas mantienen una estrecha relación con los servicios proporcionados (véase la Figura 9.1):

- Capa de infraestructura: contiene los recursos hardware más básicos como son los recursos computacionales, los de almacenamiento o los relacionados con la infraestructura de red.
- Capa de recursos unificados: empleando los recursos básicos, se encapsulan, a través de la virtualización, nuevos recursos más complejos que pueden ser proporcionados a las capas superiores o directamente a los usuarios finales.
- Capa de plataforma: proporciona a los recursos unificados herramientas especializadas, *middleware* y servicios. Su finalidad es preparar un entorno de desarrollo o de despliegue que pueda ser utilizado por la capa de aplicaciones o proporcionado como servicio a los usuarios.
- Capa de aplicaciones: contiene aplicaciones que son ejecutadas en el Cloud y expuestas como servicios para ser utilizadas por los usuarios finales.

Los Clouds, en general, proporcionan bajo demanda hasta tres tipos de servicios a los que el usuario puede acceder a través de un navegador Web o de un API bien definida [105]:

- Infraestructura como Servicio (IaaS): proporciona recursos computacionales (procesamiento y almacenamiento) virtualizados. Esto permite que los usuarios puedan obtener recursos con unas determinadas características (número de procesadores, memoria, espacio en disco, etc.) para cubrir unas necesidades concretas.

– Plataforma como Servicio (PaaS): los servicios de plataforma están pensados para proporcionar a los usuarios entornos específicos, bien sea para desarrollar software o bien sea para desplegarlo, permitiendo que los usuarios se abstraigan del hardware y/o configuraciones.

– Sofware como Servicio (SaaS): proporciona software específico para que sea accesible remotamente por los clientes a través de Internet.

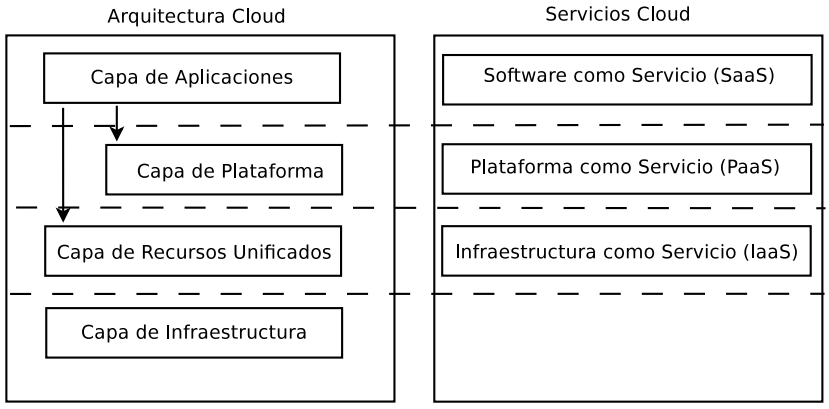

Figura 9.1: Arquitectura del Cloud relacionada con los servicios que proporciona. Figura adaptada de [105].

9.1.2. Integración de la computación Cloud en RETELAB

La integración de la computación Cloud en el proyecto RETELAB podría proporcionar una mayor flexibilidad al laboratorio. Existen varias líneas de futuro interesantes para desarrollar:

– Agregación de recursos bajo demanda: los recursos proporcionados por RETELAB están limitados tanto en su número como en sus características. Si la demanda de recursos por parte de los usuarios excede la capacidad del laboratorio, ya sea porque la cantidad de trabajos ejecutados es mayor de la que puede absorber, porque los trabajos requieren recursos con características superiores (número de procesadores, memoria, etc.) o porque necesitan de un determinado entorno de ejecución que no está disponible, el Cloud podría proporcionar una solución rápida y puntual. Aprovechando que el Cloud

proporciona IaaS, podrían agregarse bajo demanda los recursos necesarios y gracias al PaaS sería posible obtener recursos con una determinada configuración. La tarea de instanciar los recursos recaería sobre el metaplanificador GridWay del que existe un plugin (Broker GW-SLA) que permite negociar con entidades externas para incorporar recursos al Grid y que ya fue utilizado con éxito en proyectos previos [55].

– Modificación dinámica de los recursos: el uso de la computación Cloud también sería útil para gestionar los propios recursos de RETELAB, ya que podría optimizar su uso. Cuando se ejecuta un trabajo en el laboratorio, el metaplanificador necesita conocer sus necesidades para poder gestionar su ejecución. Puede darse el caso de que un trabajo sea asignado a un recurso libre y lo infrautilice, ya que no necesita de todo su potencial (procesadores, memoria, etc.). Mientras tanto, puede llegar a la cola un trabajo que, aunque realmente necesite mejores condiciones para que su ejecución sea óptima, podría aprovechar la parte ociosa del recurso a la espera de mejores recursos. Desplegando un Cloud sobre los recursos disponibles podrían virtualizarse y optimizarse generando recursos «a medida». Asociando a cada cada tarea información sobre cuáles serían los requisitos mínimos y cuáles los deseados para su ejecución, se podría virtualizar y asignar un recurso en el momento en que estuviesen disponibles los recursos mínimos. A medida que los recursos base fuesen quedando ociosos, se aprovecharía la escalabilidad dinámica del Cloud para mejorar el recurso virtual asignado a la tarea hasta alcanzar los requisitos deseados. El metaplanificador sería el encargado de gestionar este proceso, por lo que sería necesario extender el JSDL para soportar dicha información.

9.2. Evolución de las aplicaciones desplegadas: SENTINAZOS

9.2.1. Fuentes de datos

Los algoritmos desarrollados para localizar vertidos de hidrocarburos están basados en datos obtenidos por el satélite Envisat a través del modo de operación WSM. A pesar de que la base de los algoritmos es válida para otras fuentes de datos, es cierto que éstos deberían ser adaptados para ajustarse a sus particularidades como, por ejemplo, a las diferentes resoluciones espaciales. La adaptación de los algoritmos para trabajar con diferentes fuentes siempre fue considerado como un trabajo futuro que actualmente y debido a la pérdida de contacto con el satélite Envisat, ha cobrado mayor relevancia. A corto plazo los algoritmos deberían ser adaptados para analizar datos provenientes del Radarsat, mientras que a medio plazo sería

interesante adaptarlos para trabajar con los datos que provengan del Sentinel-1, satélite cuyo lanzamiento está previsto para el año 2013.

9.2.2. Mejora de los algoritmos

Existen varias líneas de trabajo que podrían conducir a una mejora de los algoritmos actuales para la detección de vertidos:

- Información contextual: dado que los sentinazos son generados habitualmente por buques que transitan por el ESTF o por sus cercanías, sería interesante añadir información contextual respecto a su entorno como, por ejemplo, la proximidad a una posible fuente del vertido o su localización respecto al ESTF. Esta información podría ser obtenida empleando los propios datos SAR, ya que es posible utilizarlos para detectar embarcaciones [89] y, al estar georreferenciados, también es posible posicionar los sentinazos respecto al ESTF.

- Incorporación de datos provenientes de los Sistemas Automáticos de Identificación (AISs) de buques: la Organización Internacional Marítima (IMO) requiere que la mayor parte de las embarcaciones posean un sistema AIS para identificarlos y localizarlos. Estos sistemas envían sus datos a estaciones receptoras que almacenan y procesan la información. SASEMAR ha implantado cobertura para AIS a lo largo de toda la costa española, por lo que la información recogida podría ser utilizada en el sistema de detección para acotar los posibles causantes del vertido.

- Incorporación de nuevos datos de viento: unos pocos informes de las misiones aéreas a las que tuvo acceso SASEMAR contenían medidas de la velocidad del viento de las zonas en las que fueron detectados vertidos. Estas medidas no se ajustaban correctamente a las obtenidas por el modelo CMOD5 y, aunque la comparación es complicada debido a que las misiones aéreas eran enviadas después de obtener y procesar la imagen SAR (no correspondían al mismo momento), se considera que una posible línea futura es incorporar datos reales o de otros modelos que puedan ser más precisos y así poder comparar los resultados con los obtenidos actualmente.

- Análisis de nuevos clasificadores: a pesar de que los resultados obtenidos con las ANNs y con los árboles de decisión son satisfactorios, sería deseable analizar otros clasificadores con el objetivo de alcanzar un mejor porcentaje de aciertos. De entre las diferentes

posibilidades, las Máquinas de Soporte Vectorial (SVM) resultan especialmente interesantes ya que, en problemas tanto de clasificación como de regresión, tratan de definir un hiperplano en un espacio n-dimensional que separe las posibles clases de una forma estricta desde el punto de vista matemático. Otras opciones posibles serían las redes funcionales o los clasificadores basados en lógica borrosa.

Bibliografía

[1] European Space Agency (ESA). https://earth.esa.in.

[2] PURSE: Portal-based User Registration Service. http://www.Grids-center.org/solutions/purse, (Retrieved October 2012).

[3] Sakai project Web Page. http://www.sakaiproject.org/ (Retrieved October 2012).

[4] The Apache Velocity Project. http://velocity.apache.org (Retrieved October 2012).

[5] Tupelo Web Page. http://tupeloproject.ncsa.uiuc.edu/ (Retrieved October 2012).

[6] Xen Web page. http://www.xen.org/ (Retrieved October 2012).

[7] ISO 19115:2003 Geographic information - Metadata. Technical report, International Organization for Standardization, 2003.

[8] *The Grid 2, Second Edition: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann, 2 edition, December 2003.

[9] Ahmar Abbas. *Grid computing a practical guide to technology and applications*. Charles River Media, 2004.

[10] A. Abdelnur and S. Hepper. JSR 168: Portlet Specification. Technical report, Sun Microsystems, October 2003. Also available online: http://www.jcp.org/en/jsr/detail?id=168 (Retrieved Sept 2012).

[11] International Energy Agency. *Medium-Term Oil and Gas Markets*. IEA Publications, 2011. Also available online: http://www.iea.org/publications/freepublications/publication/MTOGM2011_Unsecured.pdf (Retrieved Sept 2012).

[12] Jay Alameda, Marcus Christie, Geoffrey Fox, Joe Futrelle, Dennis Gannon, Mihael Hategan, Gopi Kandaswamy, Gregor von Laszewski, Mehmet A. Nacar, Marlon Pierce, Eric Roberts, Charles Severance, and Mary Thomas. The Open Grid Computing Environments collaboration: portlets and services for science gateways. *Concurrency Computat.: Pract. Exper.*, 19(6):921–942, 2007.

[13] William Allcock, John Bresnahan, Rajkumar Kettimuthu, Michael Link, Catalin Dumitrescu, Ioan Raicu, and Ian Foster. The Globus Striped GridFTP Framework and Server. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, volume 0 of *SC '05*, Washington, DC, USA, 2005. IEEE Computer Society.

[14] Ethem Alpaydin. Introduction to Machine Learning (Adaptive Computation and Machine Learning series). chapter 9, pages 185–208. The MIT Press, second edition, December 2009.

[15] A. Andronico, R. Barbera, A. Falzone, G. Lo Re, A. Pulvirenti, and A. Rodolico. GENIUS: a web portal for the grid. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 502(2-3):433–436, April 2003.

[16] A. Anjomshoaa, F. Brisard, M. Drescher, D. Fellows, A. Ly, S. McGough, D. Pulsipher, and A. Savva. ob submission description language (jsdl) specification, version 1.0. Technical report, Open Grid Forum, November 2005. Also available online: http://www.ogf.org/documents/GFD.56.pdf (Retrieved October 2012).

[17] Mario Antonioletti, Malcolm Atkinson, Rob Baxter, Andrew Borley, Neil P. Chue Hong, Brian Collins, Neil Hardman, Alastair C. Hume, Alan Knox, Mike Jackson, Amy Krause, Simon Laws, James Magowan, Norman W. Paton, Dave Pearson, Tom Sugden, Paul Watson, and Martin Westhead. The design and implementation of Grid database services in OGSA-DAI. *Concurrency Computat.: Pract. Exper.*, 17(2-4):357–376, 2005.

[18] C. F. Balseiro, P. Carracedo, B. Gómez, P. C. Leitão, P. Montero, L. Naranjo, E. Penabad, and V. Pérez-Muñuzuri. Tracking the Prestige oil spill: An operational experience in simulation at MeteoGalicia. *Weather*, 58(12):452–458, 2003.

[19] R. Barbera, A. Falzone, and A. Rodolico. The genius grid portal. In *Computing in High Energy and Nuclear Physics*, volume 24, page 28, 2003.

[20] M. J. Behrenfeld and P. G. Falkowski. A consumer's guide to phytoplankton primary productivity models. *Limnology and Oceanography*, 1997.

[21] G. Benelli and A. Garzelli. Oil-spills detection in SAR images by fractal dimension estimation. In *Geoscience and Remote Sensing Symposium, 1999. IGARSS '99 Proceedings. IEEE 1999 International*, volume 1, pages 218–220 vol.1. IEEE, 1999.

[22] Fran Berman, Geoffrey Fox, and Tony Hey. The Grid: Past, Present, Future. In *Grid Computing*, pages 9–50. John Wiley & Sons, Ltd, 2003.

[23] D. Bernholdt, S. Bharathi, D. Brown, K. Chanchio, M. Chen, A. Chervenak, L. Cinquini, B. Drach, I. Foster, P. Fox, and Others. The earth system grid: Supporting the next generation of climate modeling research. *Proceedings of the IEEE*, 93(3), 2005.

[24] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[25] Camilla Brekke and Anne H. S. Solberg. Oil spill detection by satellite remote sensing. *Remote Sensing of Environment*, 95(1):1–13, March 2005.

[26] Camilla Brekke and Anne H. S. Solberg. Classifiers and Confidence Estimation for Oil Spill Detection in ENVISAT ASAR Images. *IEEE Geoscience and Remote Sensing Letters*, 5(1):65–69, January 2008.

[27] D. Chadwick, A. Otenko, and E. Ball. Role-based access control with X.509 attribute certificates. *Internet Computing, IEEE*, 7(2):62–69, March 2003.

[28] David W. Chadwick and Alexander Otenko. The PERMIS X.509 role based privilege management infrastructure. *Future Generation Computer Systems*, 19(2):277–289, February 2003.

[29] C. F. Chen, K. S. Chen, L. Y. Chang, and A. J. Chen. The use of satellite imagery for monitoring coastal environment in Taiwan. In *Geoscience and Remote Sensing, 1997. IGARSS '97. Remote Sensing - A Scientific Vision for Sustainable Development., 1997 IEEE International*, volume 3, pages 1424–1426 vol.3. IEEE, August 1997.

[30] R. Chinnici, J. J. Moreau, A. Ryman, and S. Weerawarana. Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. Technical report, W3C, June 2007.

[31] E. Chuvieco. Radar. In *Teledetección ambiental*, chapter 3, pages 108–116. Ariel Publications, 1 edition, May 2002.

[32] P. Cornillon, J. Gallagher, and T. Sgouros. OPeNDAP: Accessing data in a distributed, heterogeneous environment. *Data Science Journal*, 2:164–174, 2003.

[33] Carmen Cotelo, Andrés Gómez, J. Ignacio López, David Mera, José M. Cotos, J. Pérez Marrero, and Constantino Vázquez. Retelab: A geospatial grid web laboratory for the oceanographic research community. *Future Generation Computer Systems*, 26(8):1157–1164, October 2010.

[34] José M. Cotos and Alejandro Tobar. Teledetección de la marea negra del petrolero Prestige. In *Teledetección de pesquerías y predicción de mareas tóxicas*, chapter 9, pages 177–182. Netbiblo, first edition, 2002.

[35] L. da Fontoura Costa and R. M. Cesar. *Shape analysis and classification: theory and practice*. CRC, 2001.

[36] L. Dagum and R. Menon. OpenMP: an industry standard API for shared-memory programming. *IEEE Computational Science and Engineering*, 5(1):46–55, January 1998.

[37] F. Del Frate, A. Petrocchi, J. Lichtenegger, and G. Calabresi. Neural networks for oil spill detection using ERS-SAR data. *IEEE Transactions on Geoscience and Remote Sensing*, 38(5):2282–2287, September 2000.

[38] Dietmar Erwin and David Snelling. UNICORE: A Grid Computing Environment. In Rizos Sakellariou, John Gurd, Len Freeman, and John Keane, editors, *Euro-Par 2001 Parallel Processing*, volume 2150 of *Lecture Notes in Computer Science*, pages 825–834. Springer Berlin / Heidelberg, 2001.

[39] Esa. Oil pollution monitoring, in ERS and its applications: Marine. Technical report, 1998.

[40] H. A. Espedal. Satellite SAR oil spill detection using wind history information. *International Journal of Remote Sensing*, 20(1):49–65, January 1999.

[41] B. Fiscella, A. Giancaspro, F. Nirchio, P. Pavese, and P. Trivero. Oil spill detection using marine SAR images. *International Journal of Remote Sensing*, 21(18):3561–3566, 2000.

[42] FAO Fisheries and Aquaculture Dept. *The state of world fisheries and aquaculture 2012.* Food and Agriculture Organization of the United Nations., 2012.

[43] J. Fortuny-Guasch. Improved Oil Slick Detection and Classification with Polarimetric SAR. In *Applications of SAR Polarimetry and Polarimetric Interferometry*, volume 529 of *ESA Special Publication*, April 2003.

[44] I. Foster, J. Geisler, B. Nickless, W. Smith, and S. Tuecke. Software infrastructure for the I-WAY high-performance distributed computing experiment. In *High Performance Distributed Computing, 1996., Proceedings of 5th IEEE International Symposium on*. IEEE, 1996.

[45] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *International Journal of High Performance Computing Applications*, 11(2), 1997.

[46] I. Foster and C. Kesselman. Computational Grids. In *The grid: blueprint for a new computing infrastructure*, chapter 2, pages 15–53. Morgan Kaufmann, 1998.

[47] I. Foster and C. Kesselman. The Globus project: A status report. *Future Generation Computer Systems*, 15(5), 1999.

[48] I. Foster, Y. Zhao, I. Raicu, and S. Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE'08*. Ieee, 2008.

[49] Ian Foster. What is the Grid? - a three point checklist. *GRIDtoday*, 1(6), July 2002.

[50] Ian Foster and Carl Kesselman. Concepts and Architecture. In *The Grid 2, Second Edition: Blueprint for a New Computing Infrastructure.*, chapter 4, pages 37–63. Morgan Kaufmann, 2 edition, December 2003.

[51] Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke. The physiology of the grid. In *Grid Computing: Making The Global Infrastructure a Reality*, pages 217–249. John Wiley & Sons, Ltd, 2003.

[52] Ian Foster, Carl Kesselman, and Steven Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *Int. J. High Perform. Comput. Appl.*, 15(3):200–222, August 2001.

[53] Ian Foster, Veronika Nefedova, Mehran Ahsant, Rachana Ananthakrishnan, Lee Liming, Ravi Madduri, Olle Mulmo, Laura Pearlman, and Frank Siebenlist. Streamlining Grid Operations: Definition and Deployment of a Portal-based User Registration Service. *Journal of Grid Computing*, 4(2):135–144, June 2006.

[54] O. Garcia-Pineda, B. Zimmer, M. Howard, W. Pichel, Li XiaoFeng, and I. R. MacDonald. Using SAR images to delineate ocean oil slicks with a texture-classifying neural network algorithm (TCNNA). *Canadian Journal of Remote Sensing*, 35(5):411–421, 2009.

[55] A. Gómez. Remote Conputational Tools for Radiotherapy Cancer Treatment Planning. In K. Stanoevska-Slabeva, T. Wozniak, and S. Ristol, editors, *Grid and cloud computing: a business perspective on technology and applications*, chapter 9, pages 147–158. Springer Publishing Company, Incorporated, 2009.

[56] W. W. Gregg, M. E. Conkright, P. Ginoux, J. E. O'Reilly, and N. W. Casey. Ocean primary production and climate: Global decadal changes. *Geophys. Res. Lett*, 30(15):1809, 2003.

[57] Andrew S. Grimshaw, Wm, and CORPORATE The Legion Team. The Legion vision of a worldwide virtual computer. *Commun. ACM*, 40(1):39–45, January 1997.

[58] Hecht-Nielsen. Theory of the backpropagation neural network. In *International Joint Conference on Neural Networks*, pages 593–605 vol.1. IEEE, June 1989.

[59] S. Hepper. *Comparing the JSR 168 Java Portlet Specification with the IBM Portlet API*, December 2003. Also available online: http://www.ibm.com/developerworks/websphere/library/techarticles/0312_hepper/hepper.html (Retrieved Sept 2012).

[60] H. Hersbach, A. Stoffelen, and S. de Haan. An improved C-band scatterometer ocean geophysical model function: CMOD5. *Journal of Geophysical Research*, 112(C3):C03006, March 2007.

[61] S. Hesmer, P. Fischer, T. Buckner, and I. Schuster. *Portlet development guide*, 1 edition, April 2002. Also available online: ftp://ftp.software.ibm.com/software/webserver/portal/V41PortletDevelopmentGuide.pdf (Retrieved Sept 2012).

[62] J. Horstmann and W. Koch. Measurement of ocean surface winds using synthetic aperture radars. *Oceanic Engineering, IEEE Journal of*, 30(3):508–515, July 2005.

[63] H. A. Hovland, J. A. Johannessen, and G. Digranes. Slick detection in SAR images. In *Proceedings of IGARSS '94 - 1994 IEEE International Geoscience and Remote Sensing Symposium*, pages 2038–2040. IEEE, 1994.

[64] Eduardo Huedo, Ruben S. Montero, and Ignacio M. Llorente. A framework for adaptive execution in grids. *Softw: Pract. Exper.*, 34(7):631–651, 2004.

[65] Eduardo Huedo, Rubén S. Montero, and Ignacio M. Llorente. A modular meta-scheduling architecture for interfacing with pre-WS and WS Grid resource management services. *Future Generation Computer Systems*, 23(2):252–261, February 2007.

[66] C. Jackson and J. Apel, editors. *Synthetic Aperture Radar Marine User's Manual*. U.S. Departement of Commerce : National Oceanic and Atmospheric Administration, 1st edition, April 2005. Also available online: http://www.sarusersmanual.com (Retrieved May 2012).

[67] I. T. Jolliffe and MyiLibrary. *Principal component analysis*, volume 2. Wiley Online Library, 2002.

[68] Maozhen Li and Mark Baker. Applications. In *The grid : core technologies*, pages 377–400. Wiley, May 2005.

[69] Maozhen Li and Mark Baker. Grid Portals. In *The grid : core technologies*, pages 335–376. Wiley, May 2005.

[70] Maozhen Li and Mark Baker. Grid Security. In *The grid : core technologies*, pages 124–152. Wiley, May 2005.

[71] R. F. López and J. M. F. Fernández. *Las Redes Neuronales Artificiales*. Netbiblo Sl, 2008.

[72] Maged Marghany. RADARSAT for oil spill trajectory model. *Environmental Modelling and Software*, 19(5):473–483, 2004.

[73] J. M. M. Marín and S. B. Cámara. Evolución de las tecnologías Grid de la información. In *Las tecnologías Grid de la información como nueva herramienta empresarial*, pages 87–111. Septem, 2008.

[74] S. W. McCandless and C. Jackson. Principles of Synthetic Aperture Radar. chapter 1, pages 1–23. U.S. Department of Commerce : National Oceanic and Atmospheric Administration, 1st edition, April 2005. Also available online: http://www.sarusersmanual.com (Retrieved May 2012).

[75] David Mera, José Cotos, Joaquín Trinanes, and Carmen Cotelo. An Integrated Solution to Store, Manage and Work with Datasets Focused on Metadata in the Retelab Grid Project. In Sigeru Omatu, Miguel Rocha, José Bravo, Florentino Fernández, Emilio Corchado, Andrés Bustillo, and Juan Corchado, editors, *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, volume 5518 of *Lecture Notes in Computer Science*, chapter 71, pages 491–494. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2009.

[76] David Mera, José Cotos, José Viqueira, and José Varela. A User Management Web System Based on Portlets for a Grid Environment Integrating Shibboleth, PURSe, PERMIS and Gridsphere. In Juan Corchado, Sara Rodríguez, James Llinas, and José Molina, editors, *International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008)*, volume 50 of *Advances in Soft Computing*, chapter 3, pages 19–23. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2009.

[77] David Mera, José M. Cotos, Pedro Saco, and Andrés Gómez. An integrated Solution to the Security User Access in the RETELAB Grid Project, using a Web System based on Portlets and a RBAC Model by means of User Attribute Certificates and PKI. In Fernando Silva, Gaspar Barreira, and Ligia Ribeiro, editors, *2nd Iberian Grid Infrastructure Conference Proceedings*, pages 296–307, 2008.

[78] David Mera, José M. Cotos, José Varela, Carmen Cotelo, and J. Ignacio López. An Integration of Several Technologies in the Architecture Definition and Deployment of a Geospatial Grid Web Portal. In Hamid R. Arabnia and George A. Gravvanis, editors,

Proceedings of the 2009 International Conference on Grid Computing and Applications, pages 86–91. CSREA Press, 2009.

[79] David Mera, José M. Cotos, José Varela-Pet, and Oscar Garcia-Pineda. Adaptive thresholding algorithm based on SAR images and wind data to segment oil spills along the northwest coast of the Iberian Peninsula. *Marine Pollution Bulletin*, 64(10):2090–2096, October 2012.

[80] David Mera, José M. Cotos, José R. R. Viqueira, and Carmen Cotelo. Software Integration in the Development of a Spatial Data Grid Prototype based on Metadata. In Vicente H. García, Gaspar Barreira, Ignacio B. Espert, and Jorge Gomes, editors, *3rd Iberian Grid Infrastructure Conference Proceedings*, pages 305–314, 2009.

[81] M. Migliaccio, M. Tranfaglia, and S. A. Ermakov. A physical approach for the observation of oil spills in SAR images. *Oceanic Engineering, IEEE Journal of*, 30(3):496–507, July 2005.

[82] Daniel Minoli. *A networking approach to grid computing*. Wiley, 2005.

[83] R. L. Morgan, Scott Cantor, Steven Carmody, Walter Hoehn, and Ken Klingenstein. Federated Security: The Shibboleth Approach. *EDUCAUSE Quarterly*, 27(4):12–17, 2004.

[84] D. Murray, J. McWhirter, S. Wier, and S. Emmerson. The Integrated Data Viewer—a web-enabled application for scientific analysis and visualization. In *19th Intl Conf. on IIPS for Meteorology, Oceanography and Hydrology*, 2003.

[85] J. Novotny. The grid portal development kit. *Concurrency and Computation: Practice and experience*, 14(13-15), 2002.

[86] J. Novotny, S. Tuecke, and V. Welch. An online credential repository for the Grid: MyProxy. In *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, pages 104–111. IEEE, 2001.

[87] Jason Novotny, Michael Russell, and Oliver Wehrens. GridSphere: a portal framework for building collaborations. *Concurrency Computat.: Pract. Exper.*, 16(5):503–513, 2004.

[88] M. A. Oliver and R. Webster. Kriging: a method of interpolation for geographical information systems. *International journal of geographical information systems*, 4(3):313–332, July 1990.

[89] R. B. Olsen and T. Wahl. The ship detection capability of ENVISAT's ASAR. In *IGARSS 2003. 2003 IEEE International Geoscience and Remote Sensing Symposium. Proceedings (IEEE Cat. No.03CH37477)*, pages 3108–3110. IEEE, 2003.

[90] J. M. Torres Palenzuela, L. González Vilas, and M. Sacau Cuadrado. Use of ASAR images to study the evolution of the Prestige oil spill off the Galician coast. *International Journal of Remote Sensing*, 27(10):1931–1950, 2006.

[91] Joon S. Park, Ravi Sandhu, and Gail J. Ahn. Role-based access control on the web. *ACM Trans. Inf. Syst. Secur.*, 4(1):37–71, February 2001.

[92] P. Pavlakis, D. Tarchi, and A. Sieber. On the monitoring of illicit vessel discharges using spaceborne sar remote sensing - a reconnaissance study in the Mediterranean sea. *Annals of Telecommunications*, 56(11):700–718, November 2001.

[93] R. Rew and G. Davis. NetCDF: an interface for scientific data access. *Computer Graphics and Applications, IEEE*, 10(4):76–82, July 1990.

[94] M. Romberg. The UNICORE architecture: seamless access to distributed resources. In *High Performance Distributed Computing, 1999. Proceedings. The Eighth International Symposium on*, pages 287–293. IEEE, 1999.

[95] Betlem Rosich and Peter Meadows. Absolute Calibration of ASAR Level 1 Products Generated with PF-ASAR. Technical Report Iss. 1 rev. 5, ESA, October 2004.

[96] D. E. Rumelhart, G. E. Hintont, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088), 1986.

[97] M. Russell, J. Novotny, and O. Wehrens. Gridsphere's Grid Portlets. *Computational Methods In Science and Technology*, 12(1), 2006.

[98] Alexander F. Shchepetkin and James C. McWilliams. The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean Modelling*, 9(4):347–404, January 2005.

[99] J. Shiers. The worldwide LHC computing grid (worldwide LCG). *Computer physics communications*, 177(1), 2007.

[100] G. Singh, S. Bharathi, A. Chervenak, E. Deelman, C. Kesselman, M. Manohar, S. Patil, and L. Pearlman. A Metadata Catalog Service for Data Intensive Applications. In *Supercomputing, 2003 ACM/IEEE Conference*, page 33. IEEE, November 2003.

[101] J. Sirott, J. Callahan, and S. Hankin. Inside the live access server. In *17th Intl Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology, AMS*, 2001.

[102] Chris Smith and Ian Lumb. The Open Grid Services Architecture. In *Grid computing a practical guide to technology and applications*, chapter 9, pages 159–187. Charles River Media, 2004.

[103] A. H. S. Solberg, G. Storvik, R. Solberg, and E. Volden. Automatic detection of oil spills in ERS SAR images. *IEEE Transactions on Geoscience and Remote Sensing*, 37(4):1916–1924, July 1999.

[104] Anne H. S. Solberg, Camilla Brekke, and Per O. Husoy. Oil Spill Detection in Radarsat and Envisat SAR Images. *IEEE Transactions on Geoscience and Remote Sensing*, 45(3):746–755, March 2007.

[105] K. Stanoevska-Slabeva and T. Wozniak. Cloud Basics- An introduction to Cloud Computing. In K. Stanoevska-Slabeva, T. Wozniak, and S. Ristol, editors, *Grid and cloud computing: a business perspective on technology and applications*. Springer Publishing Company, Incorporated, 2009.

[106] Kevin Swingler. Applying Neural Networks: A Practical Guide. chapter 3.2.3, pages 53–56. Morgan Kaufmann, pap/dsk edition, May 1996.

[107] M. Thomas, S. Mock, M. Dahan, K. Mueller, D. Sutton, and J. R. Boisseau. The GridPort toolkit: a system for building Grid portals. In *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, pages 216–227. IEEE, 2001.

[108] K. Topouzelis, V. Karathanassi, P. Pavlakis, and D. Rokos. Detection and discrimination between oil spills and look-alike phenomena through neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62(4):264–270, September 2007.

[109] K. Topouzelis, D. Stathakis, and V. Karathanassi. Investigation of genetic algorithms contribution to feature selection for oil spill detection. *Int. J. Remote Sens.*, 30(3):611–625, January 2009.

[110] Konstantinos Topouzelis. Oil Spill Detection by SAR Images: Dark Formation Detection, Feature Extraction and Classification Algorithms. *Sensors*, 8(10):6642–6659, October 2008.

[111] Konstantinos Topouzelis and Apostolos Psyllos. Oil spill feature selection and classification using decision tree forest on SAR image data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 68:135–143, March 2012.

[112] P. Tröger, R. Brobst, D. Gruber, M. Mamonski, and D. Templeton. Distributed Resource Management application API Version 2 (DRMAA). Technical report, Open Grid Forum, January 2012. Also available online: http://www.ogf.org/documents/GFD.194.pdf (Retrieved October 2012).

[113] G. Von Laszewski, I. Foster, J. Gawor, and P. Lane. A Java commodity grid kit. *Concurrency and Computation: practice and experience*, 13(8-9), 2001.

[114] Ping Wang, Y. Tony Song, Yi Chao, and Hongchun Zhang. Parallel Computation of the Regional Ocean Modeling System. *International Journal of High Performance Computing Applications*, 19(4):375–385, 2005.

[115] Barry Wilkinson. System Infrastructure II: Grid Computing Services. In *Grid computing : techniques and applications*, pages 201–222. CRC Press, 2010.

[116] Chongjie Zhang, Ian Kelley, and Gabrielle Allen. Grid portal solutions: a comparison of GridPortlets and OGCE. *Concurrency Computat.: Pract. Exper.*, 19(12):1739–1748, 2007.

Anexos

ANEXO A

CÓDIGO FUENTE

A.1. Procedimiento para el registro de una JCE

Un error no corregido del JCE Cryptix impide su uso en el caso de que algún otro JCE sea registrado después de su utilización. El orden de registro de los proveedores funciona como una lista *Last In First Out* (LIFO), por lo que se desarrolló un procedimiento para cambiar el orden de la lista en caso de que éste no fuese el adecuado. Una explicación más detallada del problema se presenta en la Sección 3.2.1.

```
//Cambiando Proveedores
RetelabSigningUtility signingUtility = new RetelabSigningUtility();
if (java.security.Security.getProviders().length>1 && java.security.Security.
     getProviders()[0].getName().equals(new String("IAIK")))
{
    java.security.Provider provider_iaik=java.security.Security.getProvider("IAIK");
    java.security.Security.removeProvider("IAIK");
    java.security.Security.insertProviderAt(provider_iaik, java.security.Security.
         getProviders().length);
}
```

Código A.1: Modificación del orden de los proveedores JCE para evitar el *bug* del proveedor Cryptix.

A.2. Gestión de certificados y roles desde la interfaz de GridSphere

GridSphere gestiona la autorización de los usuarios a los portlets a través de roles, mientras que la integración de PERMIS en RETELAB permitió hacer lo propio sobre los recursos del Grid. En el caso del PERMIS, los roles son gestionados a través de ACs, por lo que para simplificar la administración se implementaron varias modificaciones sobre el código de GridSphere para que éste pudiese gestionar los roles de los usuarios vía ACs y así ser administrados desde un punto único. Una explicación más detallada del desarrollo se presenta en la Sección 3.2.1.

```
private void saveUserRole(FormEvent event, User user) {
        log.debug("Entering saveUserRole()");
        PortletRequest req = event.getPortletRequest();
        // Get selected role
        PortletRole selectedRole = getSelectedUserRole(event);
        req.setAttribute("role", selectedRole);
        // If super role was chosen
        if (selectedRole.equals(PortletRole.SUPER)) {
          log.debug("Granting super role");
         // Grant super role
        this.aclManagerService.grantSuperRole(user);
        } else {
        // Revoke super role (in case they had it)
        //this.aclManagerService.revokeSuperRole(user);
        // Create appropriate access request
        Set groups = portalConfigService.getPortalConfigSettings().getDefaultGroups
             ();
        Iterator it = groups.iterator();
        GroupRequest groupRequest;
        while (it.hasNext()) {
                PortletGroup group = (PortletGroup) it.next();
                GroupEntry ge = aclManagerService.getGroupEntry(user, group);
                if (ge != null) {
                 groupRequest = this.aclManagerService.editGroupEntry(ge);
                }  else {
                groupRequest = this.aclManagerService.createGroupEntry();
        }
        groupRequest.setUser(user);
        groupRequest.setRole(selectedRole);
        groupRequest.setGroup(group);
        log.debug("Granting " + selectedRole + " role in gridsphere");
        // Submit changes
```

```
        this.aclManagerService.saveGroupEntry(groupRequest);
    }
    /*****************Start Retelab Code*************************/
     String path= req.getRealPath(".");
     log.debug("Save Role PATH: "+path);
     String ldapBaseDN="";
     GregorianCalendar notBefore=null;
     GregorianCalendar notAfter=null;
     String p12="";
     String passwordP12="";
     String bigInteger="";
     int port=389;
     String ldap_host="";
     String admin="";
     String password_ldap="";
     LdapConnector  ldap_conector= new LdapConnector();
     try{
        log.debug("Fichero de propiedades: "+path+File.separatorChar+"WEB-INF"+File.
             separatorChar+"permis_ldap.properties");
        FileInputStream file_properties =new FileInputStream(path+File.separatorChar
             +"WEB-INF"+File.separatorChar+"permis_ldap.properties");
        log.debug("Cargando Fichero de Propiedades...");
        //cargamos las propiedades del LDAP y de los certificados de atributos de
             ficheros de propiedades
        Properties retelab_properties=new Properties();
        retelab_properties.load(file_properties);
        ldapBaseDN=retelab_properties.getProperty("ldap.base.dn");
        p12=retelab_properties.getProperty("certificate.p12");
        passwordP12=retelab_properties.getProperty("password.certificate.p12");
        int year=new Integer(retelab_properties.getProperty("validity.period.before.
             year")).intValue();
        int month=new Integer(retelab_properties.getProperty("validity.period.before
             .month")).intValue();
        int day=new Integer(retelab_properties.getProperty("validity.period.before.
             day")).intValue();
        notBefore=new GregorianCalendar(year,month,day);
        year=new Integer(retelab_properties.getProperty("validity.period.after.year"
             )).intValue();
        month=new Integer(retelab_properties.getProperty("validity.period.after.
             month")).intValue();
        day=new Integer(retelab_properties.getProperty("validity.period.after.day"))
             .intValue();
        notAfter=new GregorianCalendar(year,month,day);
        bigInteger=retelab_properties.getProperty("certificate.biginteger");
        ldap_host=retelab_properties.getProperty("ldap.host");
        admin=retelab_properties.getProperty("ldap.admin");
        port=new Integer(retelab_properties.getProperty("ldap.port")).intValue();
        password_ldap=retelab_properties.getProperty("ldap.admin.password");
```

```
        ldap_conector.setLdap_host(ldap_host);
        ldap_conector.setAdmin(admin);
        ldap_conector.setPort(port);
        ldap_conector.setPassword_ldap(password_ldap);
        String dn="cn="+user.getUserName()+","+ldapBaseDN;
        Vector roles=new Vector();
        roles.add(selectedRole.getName());
        log.debug("Generando un certificado para: "+dn);
        CertificateGenerator generator=new CertificateGenerator(dn,notBefore,
             notAfter,bigInteger,roles.toArray(), p12, passwordP12);
        byte [] certificate=generator.makeCertificate();
        //Eliminamos el certificado anterior
        ArrayList atributos=new ArrayList();
        atributos.add(new  LDAPAttribute("attributeCertificateAttribute"));
        int tipo_modificacion=1;//0=ADD; 1=DELETE
        ldap_conector.modifyAttributes(dn, atributos, tipo_modificacion);
        //Creamos el nuevo certificado
        atributos=new ArrayList();
        atributos.add(new  LDAPAttribute("attributeCertificateAttribute",certificate
             ));
        tipo_modificacion=0;//0=ADD; 1=DELETE
        ldap_conector.modifyAttributes(dn, atributos, tipo_modificacion);
        }
        catch(FileNotFoundException fnfe){System.out.println("Fichero no encontrado"
             +fnfe);}
                        catch(SecurityException se){System.out.println("Permisos
                             incorrectos");}
                        catch(IOException ioe){System.out.println("IO exception");}
                        catch(NumberFormatException nfe){System.out.println("El
                             periodo de validez del certificado no es un entero "+
                             nfe);}
                        catch(Exception e){System.out.println("Excepcion no
                             controlada "+e+" End");}
    /******************End Retelab Code****************************/
}
log.debug("Exiting saveUserRole()");
}
```

Código A.2: Integración del sistema RBAC con GridSphere. Ejemplo de alta de usuarios.

A.3. Gestión de la ejecución de las tareas usando PERMIS

A continuación se muestra un ejemplo de cómo utilizar PERMIS para gestionar, a través de los roles, la autorización de un usuario para ejecutar una tarea en un recurso del Grid concreto.

```
<securityConfig xmlns="http://www.globus.org">
    <method name="createManagedJob">
        <auth-method>
            <GSITransport/>
            <GSISecureMessage/>
            <GSISecureConversation/>
        </auth-method>
    </method>
        <authz value="permisAuthz:issrg.gt4.PermisPDP gridmap"/>
    <reject-limited-proxy value="true"/>
</securityConfig>
```

Código A.3: Modificación de la política de autorización del método *createManagedJob* para integrar un sistema RBAC.

A.4. Integración de Shibboleth en el sistema

RETELAB cuenta con un sistema de autenticación SSO soportado por el software Shibboleth. El acceso al laboratorio de usuarios externos y sin registrar trae consigo dificultades para gestionar su autorización, ya que RETELAB se basa en DCs y ACs para administrar el uso de los recursos. Fue necesario desarrollar procedimientos para generar certificados temporales para los usuarios de Shibboleth. A continuación se muestran los métodos implementados para la creación de los DCs (Código A.4) y de los ACs (Código A.5) para los usuarios de Shibboleth.

```
private void createCertificate4Shibboleth(String name,String mail,String password)
        {
                //Variables Autoridad Certificadora
                String password_ca=new String();
                String bin_dir=new String();
                String tmp_dir=new String();
```

```
String ca_dir=new String();
String ca_hash=new String();

//variables MyProxy
String binDir_=new String();
String hostName_=new String();
int portNumber_=0;
String dn_=new String();
String dir_=new String();
int expirationLeadTime_=0;


log.debug("Fichero de propiedades: "+path+"WEB-INF"+File.
     separatorChar+"permis_ldap.properties");

try{
        FileInputStream file_properties =new FileInputStream(path+"
             WEB-INF"+File.separatorChar+"permis_ldap.properties");
        //Cargando Fichero de Propiedades de la CA y MyProxy
        Properties retelab_properties=new Properties();
        retelab_properties.load(file_properties);

        //simpleCA Properties
        password_ca=retelab_properties.getProperty("password.ca");
        ca_dir=retelab_properties.getProperty("path.dir.ca");
        tmp_dir=retelab_properties.getProperty("path.tmp.ca");
        bin_dir=retelab_properties.getProperty("path.bin.ca");
        ca_hash=retelab_properties.getProperty("hash.ca");

        //MyProxy Properties
        binDir_=retelab_properties.getProperty("myProxyBin");
        hostName_=retelab_properties.getProperty("myProxyHost");
        portNumber_=Integer.parseInt(retelab_properties.getProperty(
             "myProxyPort"));
        dn_=retelab_properties.getProperty("myProxyDn");
        dir_=retelab_properties.getProperty("myProxyDir");
        expirationLeadTime_=Integer.parseInt(retelab_properties.
             getProperty("expirationLeadTime"));



        //Opciones generales de la CA
        CertificateGenerationOptions cert_options= new
             CertificateGenerationOptions(bin_dir,tmp_dir,ca_dir,
             ca_hash);
        UserCertificateGeneration.initialize(cert_options);
        String certDir=null;
```

```
                        certDir = UserCertificateGeneration.generate(name,mail,
                             password);
                        log.debug("----->Certificado Generado<-------");
                        // Use CA to sign the certs
                        UserCertificateGeneration.signCerts(certDir, password_ca);
                        log.debug("----->Certificado Firmado<-------");
                        //MyProxy options
                        MyProxyOptions myProxyOpts= new MyProxyOptions(binDir_,
                             hostName_,portNumber_,dn_, dir_, expirationLeadTime_);
                        MyProxyManager.initialize(myProxyOpts);
                        //Cambiar orden de los providers JCE debido al bug de
                             Cryptix
                        changeProviders();
                        // Upload creds to MyProxy
                         MyProxyManager.storeRemoteCredential(name,password,tmp_dir+
                              "/"+name+"/");
                        log.debug("----->Certificado Subido a MyProxy<-------");
                        // Delete local certificates.
                        RegisterUtil.deleteCerts(certDir);
                        log.debug("----->Eliminados certificados locales<-------");
                        file_properties=null;
                        retelab_properties=null;
                    }catch(FileNotFoundException fnfe){System.out.println("Fichero no
                         encontrado"+fnfe);}
                     catch(SecurityException se){System.out.println("Permisos
                          incorrectos");}
                     catch(IOException ioe){System.out.println("IO exception");}
                      catch(MyProxyAccessException myExc){System.out.println("Error
                           subiendo las credenciales a MyProxy"+myExc);}
                     catch(CertificateGenerationException certExp){System.out.println("
                          Error creando las credenciales para el usuario"+certExp);}
                     catch(Exception e){System.out.println("Excepcion no controlada "+e
                          +" End");}


        }
```

Código A.4: Método para crear y almacenar DCs para los usuarios de Shibboleth.

```
private void createAttributeCertificate4Shibboleth(SportletUser user,String role) {

        String ldapBaseDN="";
        String defaultUser=null;
        GregorianCalendar notBefore=null;
        GregorianCalendar notAfter=null;
```

```
String p12="";
String passwordP12="";
String bigInteger="";
int port=389;
String ldap_host="";
String admin="";
String password_ldap="";
String defaultGridsphereUser="";
String pathGridMap="";
String password_ca=new String();
String bin_dir=new String();
String tmp_dir=new String();
String ca_dir=new String();
String ca_hash=new String();
LdapConnector  ldap_conector= new LdapConnector();

/*************generate certificate*************************/
try{
        log.debug("Fichero de propiedades: "+path+"WEB-INF"+File.
             separatorChar+"permis_ldap.properties");
        //Cargando el fichero de propiedades del LDAP y de los Certificados
             de atributos
        FileInputStream file_properties =new FileInputStream(path+"WEB-INF"+
             File.separatorChar+"permis_ldap.properties");
        Properties retelab_properties=new Properties();
        retelab_properties.load(file_properties);
        ldapBaseDN=retelab_properties.getProperty("ldap.base.dn");
        p12=retelab_properties.getProperty("certificate.p12");
        passwordP12=retelab_properties.getProperty("password.certificate.p12
             ");
        int year=new Integer(retelab_properties.getProperty("validity.period
             .before.year")).intValue();
        int month=new Integer(retelab_properties.getProperty("validity.
             period.before.month")).intValue();
        int day=new Integer(retelab_properties.getProperty("validity.period.
             before.day")).intValue();
        notBefore=new GregorianCalendar(year,month,day);
        year=new Integer(retelab_properties.getProperty("validity.period.
             after.year")).intValue();
        month=new Integer(retelab_properties.getProperty("validity.period.
             after.month")).intValue();
        day=new Integer(retelab_properties.getProperty("validity.period.
             after.day")).intValue();
        notAfter=new GregorianCalendar(year,month,day);
        bigInteger=retelab_properties.getProperty("certificate.biginteger");
        ldap_host=retelab_properties.getProperty("ldap.host");
        admin=retelab_properties.getProperty("ldap.admin");
```

```
port=new Integer(retelab_properties.getProperty("ldap.port")).
     intValue();
password_ldap=retelab_properties.getProperty("ldap.admin.password");
defaultGridsphereUser=retelab_properties.getProperty("gridsphere.
     default.user");
defaultUser=retelab_properties.getProperty("globus.default.user");
pathGridMap=retelab_properties.getProperty("globus.gridmap.path");
password_ca=retelab_properties.getProperty("password.ca");
ca_dir=retelab_properties.getProperty("path.dir.ca");
tmp_dir=retelab_properties.getProperty("path.tmp.ca");
bin_dir=retelab_properties.getProperty("path.bin.ca");
ca_hash=retelab_properties.getProperty("hash.ca");

file_properties=null;
retelab_properties=null;


ldap_conector.setLdap_host(ldap_host);
ldap_conector.setAdmin(admin);
ldap_conector.setPort(port);
ldap_conector.setPassword_ldap(password_ldap);



LDAPAttributeSet attributeSet = new LDAPAttributeSet();
attributeSet.add( new LDAPAttribute("objectclass", new String[]{"top
     ","organizationalPerson","inetOrgPerson","pmiUser"}));
attributeSet.add( new LDAPAttribute("cn",user.getUserName()));
attributeSet.add( new LDAPAttribute("givenname",new String[]{user.
     getUserName(),user.getFullName() }));
attributeSet.add( new LDAPAttribute("sn", user.getFamilyName()));
attributeSet.add( new LDAPAttribute("mail",user.getEmailAddress()));
String dn="cn="+user.getUserName()+","+ldapBaseDN;
Vector roles=new Vector();
roles.add(role);

log.debug("Generando un certificado para: "+dn);
CertificateGenerator generator=new CertificateGenerator(dn,notBefore
     ,notAfter,bigInteger,roles.toArray(), p12, passwordP12);
byte [] certificate=generator.makeCertificate();
attributeSet.add(new LDAPAttribute( "attributeCertificateAttribute",
     generator.makeCertificate()));
ldap_conector.addEntry(attributeSet, dn);
//Creamos un certificado para el usuario y lo subimos a MyProxy
createCertificate4Shibboleth(user.getUserName(),user.getEmailAddress
     (),(String)hs.getAttribute("shibboleth.user.password"));
//anhadir usuario al  GRIDMAP
String entry=new String();
```

```
                entry= convertDnToGridMapEntry(dn,defaultUser);
                if (entry!=""){
                        log.debug("Adding user to Grid-mapFile: "+entry);
                        addEntryToFile (pathGridMap,entry);
                   }
                }catch(FileNotFoundException fnfe){System.out.println("Fichero no
                     encontrado"+fnfe);}
                          catch(SecurityException se){System.out.println("Permisos
                               incorrectos");}
                          catch(IOException ioe){System.out.println("IO exception");}
                          catch(NumberFormatException nfe){System.out.println("El
                               periodo de validez del certificado no es un entero "+
                               nfe);}
                          catch(Exception e){System.out.println("Excepcion no
                               controlada "+e+" End");}
  }
```

Código A.5: Método para crear y almacenar ACs para los usuarios Shibboleth.

A.5. Despliegue del sistema de almacenamiento distribuido

El sistema de almacenamiento de datos de RETELAB se construyó utilizando como base el RLS de GT4 y el MCS. La integración de estos servicios con la interfaz Web no fue inmediata y fue necesario realizar ciertas modificaciones. Tal y como se describe en los detalle de implementación de la Sección 3.3, la clase *GenericserviceFetcher* fue modificada para hacer implícita la clase necesaria para establecer la conexión con el servicio de datos.

```
private String getWSDL(final URL url) throws IOException
   {
      /*********Start Retelab******************/
         URL url2 = new URL(url,"",(java.net.URLStreamHandler)new org.globus.net.
              protocol.https.Handler());
          InputStreamReader input = new InputStreamReader(url2.openConnection().
               getInputStream());
      /*********End Retelab******************/

       //InputStreamReader input = new InputStreamReader(url.openStream());

       StringBuffer buf = new StringBuffer();
       char[] charArray = new char[1024];
       int readBytes;
```

```
        while ((readBytes = input.read(charArray)) >= 0)
        {
            buf.append(charArray, 0, readBytes);
        }
        String wsdl = buf.toString();
        return wsdl;
    }
```

Código A.6: Modificación de la clase *GenericserviceFetcher* para permitir la conexión con el servicio de metadatos.

La integración del servicio RLS con la interfaz Web implicó cambiar ligeramente el modo de autenticar los usuarios del portal. En lugar de utilizar el proxy del usuario como un parámetro de un método de autenticación del servicio, fue necesario descargarlo de la sesión del portal y almacenarlo local y temporalmente para que el servicio pudiese «encontrarlo», ya que no soportaba el paso por variable.

```
private DataGridWorker conectarWorker( String username) throws org.ietf.jgss.
     GSSException
        {
                GSSCredential proxy;
                GlobusCredential globusCred = null;
                proxy=ProxyManager.getDefaultProxy(username);
                if(proxy instanceof org.globus.gsi.gssapi.GlobusGSSCredentialImpl)
                {
                       globusCred = ((GlobusGSSCredentialImpl)proxy).
                            getGlobusCredential();
                       //Ruta para almacenar el CD
                       String proxyfoldername=System.getProperty("user.home")+
                            System.getProperty("file.separator")+propiedades.
                            getProperty("PATH.USER.PROXIES");
                       String proxyfilename=proxyfoldername+System.getProperty("
                            file.separator")+"activeproxy"+username;
                       FileOutputStream  out =null;
                       File file=null;
                       try {
                                boolean create=true;
                                file=new File(proxyfoldername);
                                if (!file.exists())
                                       create=file.mkdir();
```

```
                                if (create)
                                 {
                                        file=new File(proxyfilename);
                                        if (!file.exists())
                                                create=file.createNewFile();
                                 }

                                if(create){
                                        out = new FileOutputStream(file);
/*Since Java does not provide an API for setting the permissions of a file, the
     JGlobus module will attempt to execute the /bin/chmod program
in the background to set the permissions of the given file. If that program cannot
     be executed for any reason or fails to execute correctly,
a proxy file might end up with incorrect file permissions (depending on umask
     setting).
Usually a warning will be displayed if that occurs (especially on Windows since /bin
     /chmod is not supported  on that platform).*/
                                        if (!org.globus.util.Util.setFilePermissions
                                             (proxyfilename, 600))
                                                 System.err.println("Por favor,
                                                      compruebe que: los permisos de
                                                       su archivo de proxy son de
                                                      solo lectura para el
                                                      propietario.");

                                        globusCred.save(out);
                                        out.close();
                                    }
                                }catch(IOException io){io.printStackTrace();}
                                finally{
                                   out=null;
                                    file=null;
                                }

                 }

                GlobusGSSCredentialImpl gssci  = new GlobusGSSCredentialImpl(
                     globusCred,GSSCredential.INITIATE_AND_ACCEPT);
                GlobusCredential.setDefaultCredential(gssci.getGlobusCredential());
                //Creamos el DataGridWorker. Ya dispone todos los datos necesarios
                     para autenticar al usuario
                return new DataGridWorker(propiedades.getProperty("METADATA.CATALOG.
                     GRID.SERVICE"),username);
        }
}
```

Código A.7: Procedimiento para autenticar a un usuario en el servicio RLS.

Listado de acrónimos

AATSR Radiómetro Avanzado de Exploración Longitudinal. 81

AC Certificado de Atributos. 39–41, 43, 44, 56, 124, 148, 151, 156

AI Inteligencia Artificial. 13

AIS Sistema Automático de Identificación. 131

ANN Red Neuronal Artificial. 75, 98–100, 103, 107, 110, 111, 114, 126, 131

API *Application Programming Interface*. 24, 26–28, 55, 116, 128

ASAR Radar de Apertura Sintética Avanzado. 82–84

AZTI Centro Tecnológico del Mar y los Alimentos del País Vasco. 11

CA Autoridad Certificadora. 35, 39

CART Árboles de Clasificación y Regresión. 105

CERN Organización Europea para la Investigación Nuclear. 28

CESGA Centro de Supercomputación de Galicia. 11, 33, 34, 54, 59, 125

CLI Interfaz de Línea de Comandos. 8, 21, 55

CoG *Java Commodity Grid Kit*. 24, 27

DB Base de Datos. 3, 5, 7, 44, 46–49, 52, 53, 57, 75, 76, 80, 85, 86, 89, 91, 97, 112, 113, 118, 119, 124, 125

DC Certificado Digital. 29, 35–44, 49, 56, 124, 151, 153

DORIS Orbitografía y Radioposicionamiento Doppler Integrado por Satélite. 81

DRMAA *Distributed Resource Management Application*. 55, 56

EMSA Agencia Europea de Seguridad Marítima. 79, 85

ESA Agencia Espacial Europea. 65, 80, 83, 85

ESG *Earth System Grid*. 29

ESTF Esquema de Separación de Tráfico de Fisterra. 77–79, 85–87, 89, 131

GMES Monitorización Global para el Medioambiente y la Seguridad. 81

GOMOS Monitorización Global del Ozono mediante la Ocultación de Estrellas. 81

GPDK *Grid Portal Development Kit*. 23

GPIR *GridPort Information Repository*. 27

GSI *Security Grid Infrastructure*. 35–37

GT *Globus Toolkit*. 18–20, 23, 24, 28, 32, 34, 35, 37, 41, 45–49, 156

GWSDL *Grid Web Service Definition Language*. 19

HPC Computación de Alto Rendimiento. 11

HTC Computación de Altas Prestaciones. 6, 11

IA Ángulo de Incidencia. 68, 69, 73, 83, 84, 89, 90, 92, 93, 95, 125

IaaS Infraestructura como Servicio. 128, 130

ICCM Instituto Canario de Ciencias del Mar. 11, 60

IDV *Integrated Data Viewer*. 53, 54, 57, 61, 125

IMO Organización Internacional Marítima. 131

JAR ARchivo Java. 40, 47

JavaWS *Java Web Start*. 30, 53

JCE Extensión Criptográfica de Java. 41, 147

JNLP *Java Networking Launching Protocol*. 53

JRE Entorno de Ejecución de Java. 53

JSDL *Job Submission Description Language*. 55, 130

JSP *Java Server Pages*. 23, 24, 28

LAS *Live Access Server*. 52, 53

LDAP *Lightweight Directory Access Protocol*. 39–41

LFN Nombre Lógico de Fichero. 45, 46, 49

LHC *Large Hadron Collider*. 3, 28

LIFO *Last In First Out*. 147

LRC *Local Replica Catalog*. 46

LRR Retrorreflector Láser. 82

MAMS *Meta Access Management System*. 43, 44

MCS *Metadata Catalog System*. 46, 48, 156

MEC Ministerio de Educación y Ciencia. 10

MERIS Espectrómetro de Imágenes de Resolución Media. 81

MIPAS Interferómetro de Michelson para Ecografía Atmosférica Pasiva. 82

MLP Perceptrón Multicapa. 100, 101, 103, 104, 113–115, 118

MPI *Message Passing Interface*. 59

MWR Radiómetro de Microondas. 81, 82

NetCDF *Network Common Data Form*. 51, 54, 59

OGCE *Open Grid Computing Environments*. 27, 28

OGSA *Open Grid Service Architecture*. 19, 20, 46

OGSI *Open Grid Services Infrastructure*. 19

OpenMP *Open Multiprocessing*. 59, 116

PaaS Plataforma como Servicio. 129, 130

PCA Análisis de Componentes Principales. 95, 126

PDP *Policy Decision Point*. 41, 42

PFN Nombre Físico de Fichero. 45, 46

PKI *Public Key Infrastructure*. 9, 35

PP Producción Primaria. 2, 51, 60

PURSe *Portal-based User Registration Service*. 29, 30, 38–40, 124

RA-2 Altímetro de Radar. 81, 82

RBAC Control de Acceso Basado en Roles. 9, 11, 37, 39, 41, 124, 150, 151

RETEMAR Red Española de Teledetección Marina. 8, 31

RLI *Replica Location Index*. 46

RLS *Replica Location Service*. 45, 49, 156–158

ROI Región de Interés. 73

ROMS *Regional Ocean Model System*. 58, 59

SaaS Sofware como Servicio. 129

SAR Radar de Apertura Sintética. 12, 50, 61, 68–73, 77, 79, 82, 84, 86, 88, 89, 91, 93, 109–111, 113, 114, 117, 118, 125, 126, 131

SASEMAR Agencia española de Salvamento y Seguridad Marítima. 79, 80, 85, 114, 131

SCIAMACHY Espectrómetro de Absorción de Exploración e Imágenes para Cartografía Atmosférica. 82

SLAR Radar Lateral Aerotransportado. 66

SSO *Single-Sign On*. 43, 44, 124, 151

SVM Máquinas de Soporte Vectorial. 132

UDC Universidad de la Coruña. 12

UNICORE *Uniform Interface to Computing Resources*. 18, 23

USC Universidad de Santiago de Compostela. 11, 12, 33

VGPM Vertically Generalised Production Model. 60

VO Organización Virtual. 3–5, 7, 17, 20, 37

WSDL *Web Services Description Language*. 19, 48

WSM *Wide Swath Mode*. 76, 82, 84–86, 130

WSRF *Web Service Resource Framework*. 19, 20

Índice alfabético