3DO: 3D Observatory, un observatorio marino virtual

Proyecto Fin de Carrera

Sebastián Pérez Pérez

Las Palmas de Gran Canaria

Julio 2014

Proyecto fin de carrera de la Escuela de Ingeniería Informática de la Universidad de Las Palmas de Gran Canaria presentado por el alumno:

SEBASTIÁN PÉREZ PÉREZ

Título del Proyecto : 3DO: 3D Observatory, un observatorio marino virtual
Tutor : Javier Sánchez Pérez

DEDICATORIA

A mi familia

AGRADECIMIENTOS

Quiero dar las gracias a todas las personas que han hecho posible este proyecto.

En primer lugar, a mis padres Agustín y Luisa y a mi novia Zaida, por haberme apoyado a lo largo de todo este tiempo, por haberme ayudado en todo lo que ha estado en sus manos y más, por haber estado a mi lado, tanto en los momentos buenos como en los malos, por haberme soportado cuando el estrés y el agobio hacía mella en mí y haberme oxigenado para seguir adelante cuando más lo he necesitado. También tengo que hacer mención a mi abuelo Juan, una persona que me ha acompañado durante todo mi proceso formativo desde bien pequeño siendo en gran medida el despertador y la cafeína que me ha mantenido despierto en las madrugadas de estudio. Y por supuesto al resto de mi familia.

A mi tutor Javier Sánchez Pérez, la persona que ha hecho posible este proyecto. Gracias por toda la ayuda que me has brindado y por haberme ofrecido siempre total disponibilidad cuando lo he necesitado independientemente del trabajo que tuvieras.

A la Plataforma Oceánica de Canarias, y en especial a Eduardo Quevedo, David Horat y Eric Delory, por el tiempo que han invertido en solucionarme las dudas que me iban surgiendo, por su implicación en el proyecto desde el primer día y por haberme dado la posibilidad de disfrutar de una beca para conocer PLOCAN desde dentro y sus actividades.

Y por último, a las personas con las que he compartido esta bonita etapa, en especial a Aridane Rosario, mi compañero de batallas a lo largo de esta guerra.

Índice de contenido

Capítulo 1: Introducción

La Plataforma Oceánica de Canarias (PLOCAN) es una instalación pública para la investigación, el desarrollo y la innovación en el ámbito de las ciencias del mar y la tecnología relacionada con éstas. En PLOCAN se trabaja para ofrecer una forma de acceso a la formación e investigación del océano profundo con las herramientas y las infraestructuras necesarias.

Entre sus instalaciones destaca el proyecto de la plataforma oceánica que será construido al Este de Gran Canaria, a una profundidad de entre 50 y 100 metros, al borde de la plataforma continental teniendo acceso a altas profundidades a escasa distancia. Esta plataforma servirá como puerta al océano profundo mediante la operación de vehículos, conectados por cable o de forma remota, y resto de maquinaria submarina como instrumentos de observación, producción, aprovechamiento de recursos o para la instalación de servicios permanentes a altas profundidades.

Uno de los pilares fundamentales de PLOCAN es el observatorio oceánico. El observatorio cubre la permanente necesidad del estudio en tiempo real de las condiciones oceánicas y atmosféricas de la zona circundante. Este estudio puede ser llevado a cabo desde los sistemas y herramientas propias de la plataforma como desde otro conjunto de dispositivos colocados, de manera fija o móvil, en partes del océano de la que interesa recabar datos.

Para que toda esta información sea útil y llegue a la mayor cantidad de personas, se requiere un portal de divulgación donde se pueda exponer al mundo la gran cantidad de datos obtenidos por toda la red de sensores. Con esto se pretende dar soporte al observatorio oceánico y crear un observatorio virtual 3D, donde se puedan ver la plataforma y todos los artefactos y vehículos en un globo terráqueo digital. Cada uno de estos elementos estarán geolocalizados y mostrarán información de diferentes índoles sobre estos dispositivos y su captura de datos.

Con la intención de hacer visualmente atractiva la aplicación a usuarios no expertos en la disciplina oceanográfica, el muestreo de datos recabados por las plataformas de observación se hará mediante gráficos dinámicos donde el usuario pueda limitar periodos de muestreo.

Por otro lado la interacción de los usuarios con la aplicación debe permitir tanto una visualización guiada de un dispositivo dado, como ofrecerle al usuario la libertar de navegar libremente a través del globo terráqueo encontrando en él los dispositivos situados en la zona que desee.

1.1. La Plataforma Oceánica de Canarias

1.1.1. Motivación

PLOCAN surge en un marco donde se ha incrementado de manera muy notable las actividades económicas e industriales en los océanos. Principalmente actividades relacionadas con las comunicaciones, la pesca de arrastre y sobretodo las prospecciones en busca de recursos petrolíferos. Este tipo de actividades son propensas a generar una gran preocupación general por el medioambiente y la conservación de los océanos. Esta concienciación conlleva a que las instituciones sin intereses económicos en estas actividades necesiten herramientas y mecanismos con los que poder hacer frente y garantizar la seguridad y la buena salud medioambiental.

Este tipo de herramientas y protocolos necesitan de unos altos conocimientos científicos y tecnología avanzada. El problema es que los costes que ésto produce sumados a los generados por su movilización los convierte prácticamente en recursos inalcanzables para la mayoría de organizaciones y países.

Es entonces donde entra en juego PLOCAN, como puerta a todas esas instituciones que no tienen los recursos económicos requeridos a las tecnologías e infraestructuras necesarias para la investigación, desarrollo e innovación en el ámbito científico, tecnológico, socioeconómico o medioambiental al que no podrían tener acceso de manera privada y de una forma medioambientalmente sostenible.

En base a ésto, la Plataforma Oceánica de Canarias se distribuye en las siguientes líneas de actuación:

- Proveer a la comunidad científico-tecnológica internacional de las herramientas necesarias para la observación, estudios y experimentación en océano profundo.
- Proveer al mundo empresarial a nivel mundial de un banco de pruebas en océano profundo de alta calidad y con garantías medioambientales.
- Tener disponible y totalmente operativa una flota de vehículos para el trabajo y la observación en océano profundo.
- Servir como punto de encuentro entre la mas ilustre comunidad científico-técnica pública y las empresas innovadoras en el sector oceánico.
- Ofrecer programas formativos de alta calidad y excelencia en todos los niveles relacionados con los dispositivos y herramientas a las que PLOCAN ofrece acceso.
- Convertirse en una organización científico-técnica pública con personal muy cualificado gestionado eficientemente centrada en la innovación.

1.1.2. Proceso de Formación

A continuación, mediante una línea temporal se explica el proceso que fue llevado a cabo para la creación de la Plataforma Oceánica de Canarias.

Enero 2007	Se incluye a la Plataforma Oceánica de Canarias entre las Instalaciones Científicas y Técnicas Singulares (ICTS) con el objetivo de impulsar la ciencia, la investigación y las tecnologías.
Junio 2007	El Consejo de Gobierno de Canarias aprueba la partida de presupuesto para la cofinanciación de la Plataforma Oceánica de Canarias
Julio 2007	Se crea la Agencia Canaria de Investigación, Innovación y Sociedad de la Información
Septiembre 2007 ***Noviembre 2007***	El Consejo de Ministros realiza los trámites para la puesta en marcha de las nuevas ICTS incluidas a principio de año.
Diciembre 2007	Firma del convenio de colaboración entre el Ministerio de Educación y Ciencia (MEC) y el Gobierno de Canarias para la creación del "Consorcio para el diseño, construcción, equipamiento y explotación de la Plataforma Oceánica de Canarias".
Diciembre 2007	Se desarrolla el proceso de selección del director de la Plataforma Oceánica de Canarias mediante un concurso a nivel internacional.
Julio 2008	Se produce el nombramiento del Dr. Octavio Llinás como director de PLOCAN.
Noviembre 2009	El director de PLOCAN presenta la versión final del proyecto ante el Comité Asesor de Infraestructuras Singulares (CAIS), órgano consultivo de asesoramiento al MEC.
2010	Se convoca licitación mediante el procedimiento de diálogo competitivo del contrato para la redacción del proyecto y ejecución de las obras de la Plataforma Oceánica de Canarias.
2011	Se aprueban las reglas del procedimiento y la solución técnica definitiva finales. Se propone la adjudicación provisional del contrato.
2012	Se produce la adjudicación definitiva y se formaliza el contrato.

1.1.3. Estructura Organizativa

La estructura organizativa de PLOCAN esta dividida en varios órganos de gobierno y de asesoramiento. Cada uno de ellos con sus diferentes funciones.

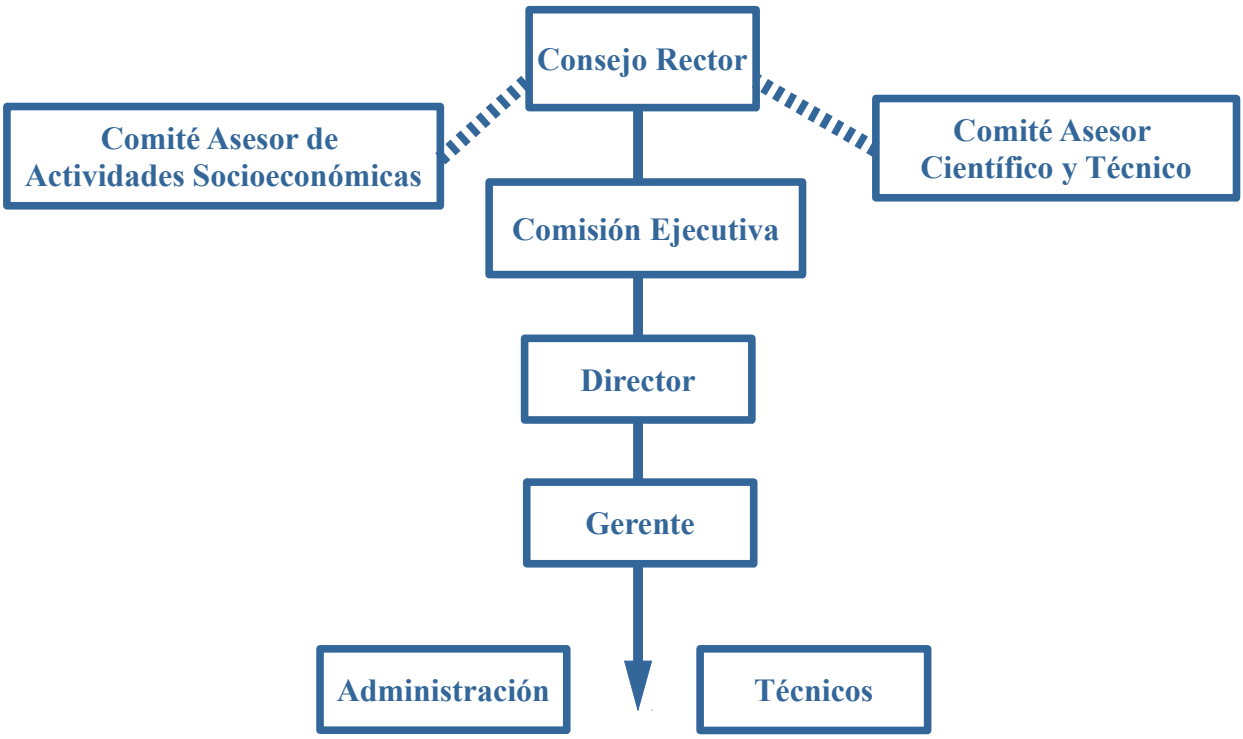
El **Consejo Rector** es el máximo órgano de gobierno y esta compuesto por representantes del Ministerio de Economía y Competitividad y del Gobierno de Canarias. Las funciones de este órgano son las de fijar las directrices del proyecto, las reglas, el funcionamiento del Consorcio,la estrategia de gestión y por último aprobar el presupuesto y las cuentas anuales de la institución.

La **Comisión Ejecutiva** es el órgano encargado del seguimiento y ejecución de las actividades desarrolladas por PLOCAN y por lo tanto los responsables de proponer el presupuesto anual y la línea de actuación del siguiente año. Su composición también está formada por el Ministerio y el Gobierno de Canarias.

La Presidencia de ambos órganos es rotatoria entre el MEC y el Gobierno de Canarias donde una institución presidirá el Consejo Rector y la otra la Comisión Ejecutiva, ello en ciclos de dos años alternándose un órgano y otro.

Por último, en el ámbito del Gobierno de PLOCAN, el siguiente nivel, ya internamente, está compuesto en primera instancia por el **Director** como máximo responsable, seguidamente del **Gerente**.

Además del aparato de gobierno existen dos órganos consultivos de asesoramiento formados por expertos en la materia. Por un lado, el **Comité Asesor de Actividades Socioeconómicas**, encargado de asesorar sobre oportunidades y estrategias a seguir. Por otro lado, el **Comité Asesor Científico y Técnico**, se centra en el asesoramiento de actividades científico-tecnológicas y de aceptación de personal técnico y científico externo a la institución.



1.1.4. Estructura Física

Como se ha comentado en los subapartados previos, la instalación principal de PLOCAN será una plataforma situada al borde de la plataforma continental y con acceso cercano a altas profundidades. Lógicamente, por los costes que una infraestructura de este tipo conlleva, es un espacio limitado y por lo tanto el número de personas que podrá hacer uso de ella al mismo tiempo es reducido. Debido a ésto surge la necesidad de unas instalaciones mas extensas en tierra firme donde el resto de personas que no puedan estar en la plataforma puedan desarrollar su trabajo. Por lo tanto, PLOCAN estará estructurada en dos espacios, una plataforma oceánica y unas instalaciones terrestres, además de las pertinentes comunicaciones necesarias para el correcto funcionamiento de sus actividades.

La plataforma oceánica

Es una estructura autónoma fija en medio del océano al Este de Gran Canaria donde pueden haber 40 personas trabajando a la vez. Está será la infraestructura principal que servirá como base para los vehículos y herramientas submarinas, como soporte principal para el banco de ensayos y experimentación en materia científica y tecnológica en océano profundo y de aprovechamiento de energías renovables. La plataforma debe permitir el acceso y la evacuación tanto por vía marítima como en helicóptero. Es primordial además un buen sistema de comunicación para permitir el acceso a los datos al personal de tierra.

La base en tierra

El edificio en tierra firme situado en Taliarte será el centro administrativo y además dispondrá de laboratorios y despachos para el trabajo del personal que no puede estar en la plataforma. Esta instalación está equipada con todo el material de oficina necesario además de taller y laboratorios para la investigación y el desarrollo.

1.1.5. Estructura Funcional

Desde el punto de vista funcional, la Plataforma Oceánica de Canarias está organizada en cinco importantes módulos individuales esenciales para la institución, y que pretenden cubrir las distintas necesidades captadas en el mundo científico, académico, tecnológico y socioeconómico de la sociedad internacionalmente. A continuación se detallan cada uno de estos módulos funcionales.

El observatorio oceánico

Se basa en un extenso sistema de dispositivos y sensores que permiten la observación y medición de magnitudes atmosféricas y oceánicas que son de vital utilidad para infinidad de aplicaciones científicas y medioambientales. En esta parte de PLOCAN es donde se centra el actual Proyecto 3DO, que tiene la misión de hacer llegar al máximo posible de personas toda esta información de la manera más sencilla y cómoda para el receptor.

El banco de ensayos

Ofrece una infraestructura donde sea muy fácil hacer experimentos y pruebas a altas profundidades o en condiciones de océano profundo, que de no ser por la plataforma sería muy dificultoso y caro.

La base de vehículos

Facilita el despliegue y recogida de vehículos submarinos. Sin la plataforma, cada vez que se quiere desplegar un dispositivo para una misión es necesario salir desde puerto en una embarcación y navegar durante millas para alcanzar las condiciones y profundidades necesarias. Y esto conlleva un coste económico además de estar sujeto a las condiciones meteorológica y estado del mar. Por lo tanto, con ello se pretende proveer de un puerto en alta mar donde realizar este tipo de actividades de forma sencilla, con las herramientas necesarias disponibles y sin limitaciones meteorológicas.

El centro de formación

PLOCAN tiene las herramientas, los profesionales y los dispositivos necesarios para ofrecer la mejor formación en el ámbito oceánico. Dado que está formado por profesionales altamente cualificados, que dispone de las últimas tecnologías en cuanto a observación oceanográfica se refiere y dispone de instalaciones preparadas para actividades formativas, aprovecha esta coyuntura para ejecutar programas formativos internacionales de alta especialización.

El centro de innovación

Haciendo uso de las posibilidades que ofrece la plataforma, su personal especialista, las herramientas y la financiación PLOCAN centra muchos esfuerzos en la innovación científica y tecnológica en el ámbito de las ciencias del mar.

1.2. Objetivos

El objetivo fundamental del proyecto es desarrollar un portal web que funcionará como observatorio 3D multimedia para las actividades de divulgación del observatorio oceánico de La Plataforma Oceánica de Canarias (PLOCAN). Las tareas que se van a abordar para realizar dicha gestión son las siguientes:

a) Visualización de vehículos, plataformas de observación y dispositivos de adquisición de datos, fijos o móviles, situadas en el medio marino, a través de un globo terráqueo virtual en el que el usuario pueda navegar.

b) Gestión de los elementos del observatorio tales como la propia plataforma y dispositivos de instrumentación, tanto conjuntamente en forma de plataformas de observación como individualmente en forma de sensores.

c) Gestión de los usuarios y perfiles de permisos y controlar el acceso a partes de la aplicación en base a estos permisos.

d) Importación y muestra de datos oceánicos y atmosféricos recabados por las plataformas de observación usando estándares OGC.

Además, como valor añadido, se pretende usar como base de desarrollo todo el software libre que sea posible, investigando y buscando alternativas al software propietario con herramientas similares bajo licencias abiertas para conseguir así una disminución del costo y por lo tanto un mayor atractivo desde el punto de vista comercial.

1.3. Estructura del Documento

A lo largo del presente documento se dará una visión clara de cada una de las etapas que se han ido llevando a cabo durante el desarrollo del proyecto. Mediante gráficos, diagramas y textos explicativos se intentará plasmar detalladamente el proceso de creación del software desde las primeras fases hasta la obtención final del producto.

Se recomienda al lector que siga el flujo normal del documento, ya que en la mayoría de los casos cada apartado depende del anterior, sobre todo en el capítulo 5 sobre el desarrollo del proyecto. La organización del documento es la siguiente:

- **Estado actual del arte.** En este capítulo se hace un estudio dando a conocer las diferentes aplicaciones que guardan relación con el problema que se pretende resolver con el proyecto. Se intentará explicar sus características y qué partes del problema consigue solucionar, o por el contrario, cuáles no son resueltas.
- **Planificación del proyecto.** Este capítulo explica la planificación que se ha llevado a cabo para el desarrollo del proyecto, así como la metodología de desarrollo que se ha usado. También incluye el plan de trabajo y el presupuesto detallado del coste total del proyecto.

- **Herramientas y tecnologías.** En el presente capítulo se explican cada una de las tecnologías o herramientas que han sido necesarias para desarrollar el proyecto. Además se enumeran tanto los recursos software como hardware participantes en la implementación del proyecto.

- **Desarrollo del proyecto.** Este capítulo concentra toda información propia del desarrollo completo de la aplicación. Está estructurado en grandes apartados que se corresponden con las fases clásicas de un proyecto software (requisitos, análisis, diseño, implementación y pruebas). Coincide con el flujo de trabajo llevado a cabo desde la lista de características, conocimiento del dominio de la aplicación, obtención de los casos de uso e identificación de los actores del sistema, el análisis y posterior diseño, implementación y pruebas del producto final.

- **Conclusiones y trabajo futuro.** Se detallan las conclusiones obtenidas tras la finalización del proyecto y se hace una reflexión sobre posibles ampliaciones en sucesivas versiones de la aplicación.

- **Anexos.** Se complementa con información que puede resultar útil al lector como un manual de usuario de la aplicación.

Capítulo 2: Estado Actual del Arte

En este segundo capítulo se pretende dar a conocer al lector una muestra representativa de distintas aplicaciones que podrían tener aspectos en común con este proyecto. Debido a que no se ha encontrado ninguna aplicación que ofrezca las mismas funcionalidades que cubre 3DO, se abordará dividiéndolo en secciones basadas en la funcionalidad que ofrecen. Por lo tanto, el presente capítulo estará organizado de la siguiente manera:

1. Se hará un muestreo de aplicaciones cuyo objetivo sea promover la divulgación científica y publicar datos científicos recabados por expediciones o proyectos en el ámbito de la oceanografía.
2. Se introducirán los sistemas de información geográfica o GIS. En esta categoría se incluyen tanto las aplicaciones de mapas planos como de globos terráqueos virtuales. Estos sistemas están basados en la georreferenciación.

2.1. Divulgación de Datos Oceanográficos

En este apartado se mostrarán cinco aplicaciones web cuyo objetivo es la divulgación de datos oceanográficos obtenidos a partir de una red de plataformas de observación.

Cada una de estas herramientas comparten la geolocalización de dispositivos en mapas y la obtención de datos y metadatos. Por otro lado, se diferencian en formatos y formas de presentar dichos datos al usuario final.

Entre las herramientas contamos con algunas gestionadas por organizaciones norteamericanas, otra por europeas e incluso una en la que participa la Plataforma Oceánica de Canarias, organización para la cuál se esta desarrollando este proyecto.

2.1.1. National Data Buoy Center (NDBC)

El Centro Nacional de Datos de Boyas (NDBC), parte de la Administración Nacional Oceanográfica y Atmosférica Americana (NOAA), opera y controla datos de más de 100 boyas fondeadas, 39 en océano profundo. NDBC controla y distribuye datos medioambientales a más de 570 estaciones asociadas para la preparación de alertas meteorológicas principalmente. Estas boyas hacen mediciones de las siguientes magnitudes: presión atmosférica, dirección del viento, velocidad y ráfaga, temperatura del aire y el agua, espectro de energía de las olas (no direccional y direccional), altura de la columna de agua (detección de tsunamis), humedad relativa, velocidad de corrientes marinas, precipitación, salinidad, radiación solar, visibilidad, nivel y calidad del agua, etc.

Todas estas mediciones están accesibles desde una herramienta web, donde cada una de estas boyas están geolocalizadas en un mapa. Cuando un usuario selecciona alguna de estas boyas se obtiene la información y las mediciones mediante gráficas estáticas.

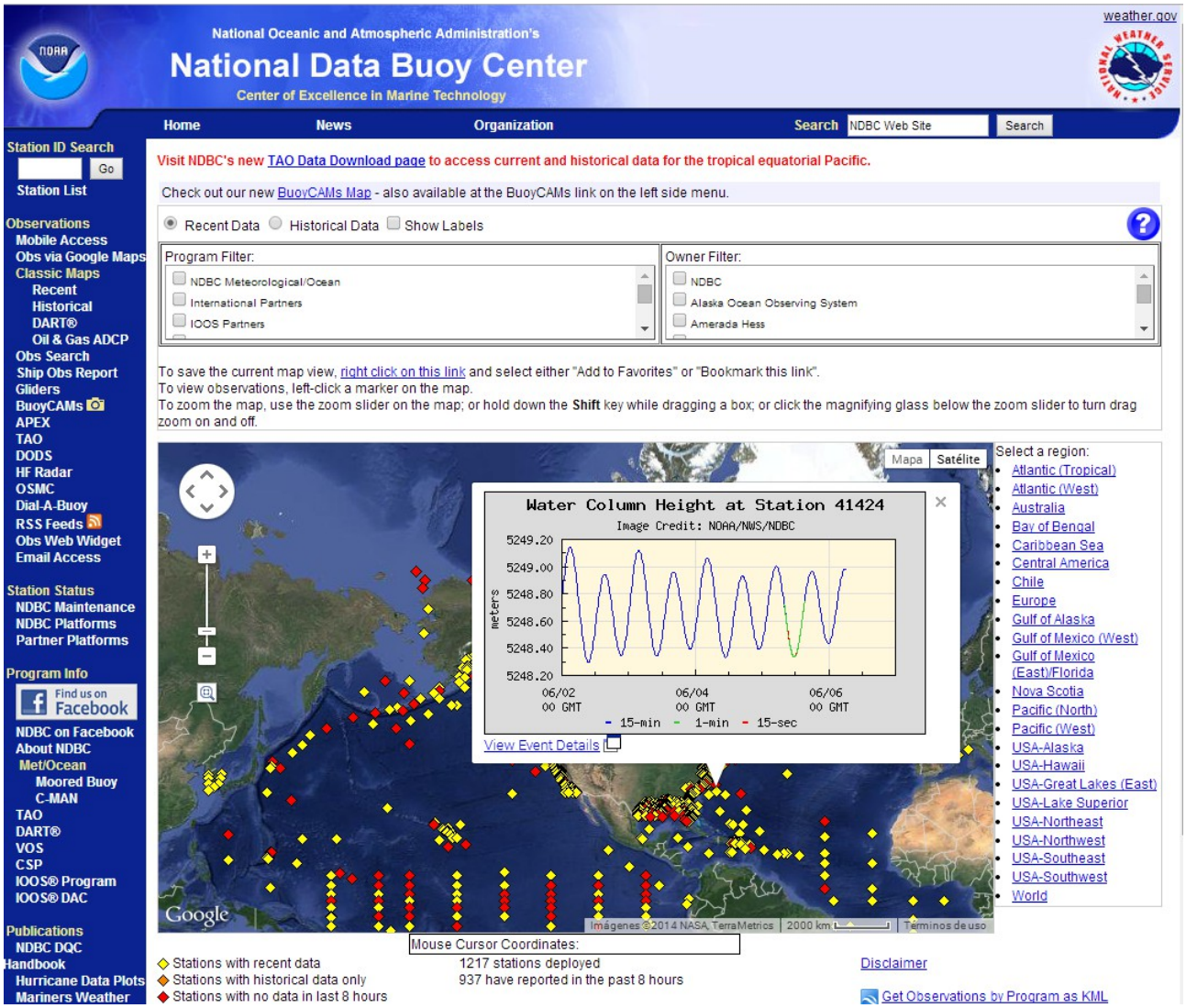


Figura 2.1. NDBC

2.1.2. Integrated Ocean Observing System (IOOS)

El Sistema Integrado de Observación del Océano (IOOS) conecta los datos de las miles de herramientas, desde los satélites a los sensores submarinos, que los recogen del océano y de las costas. IOOS ha ido ampliando las fuentes de datos y aumentando el acceso a los datos existentes.

La información está disponible en tiempo real, así como retrospectivamente. Este catálogo de datos ofrece un fácil acceso a esta información mediante un mapa de plataformas de observación que da acceso a los datos recabados y a la descripción formal de sus sensores en formato SensorML.

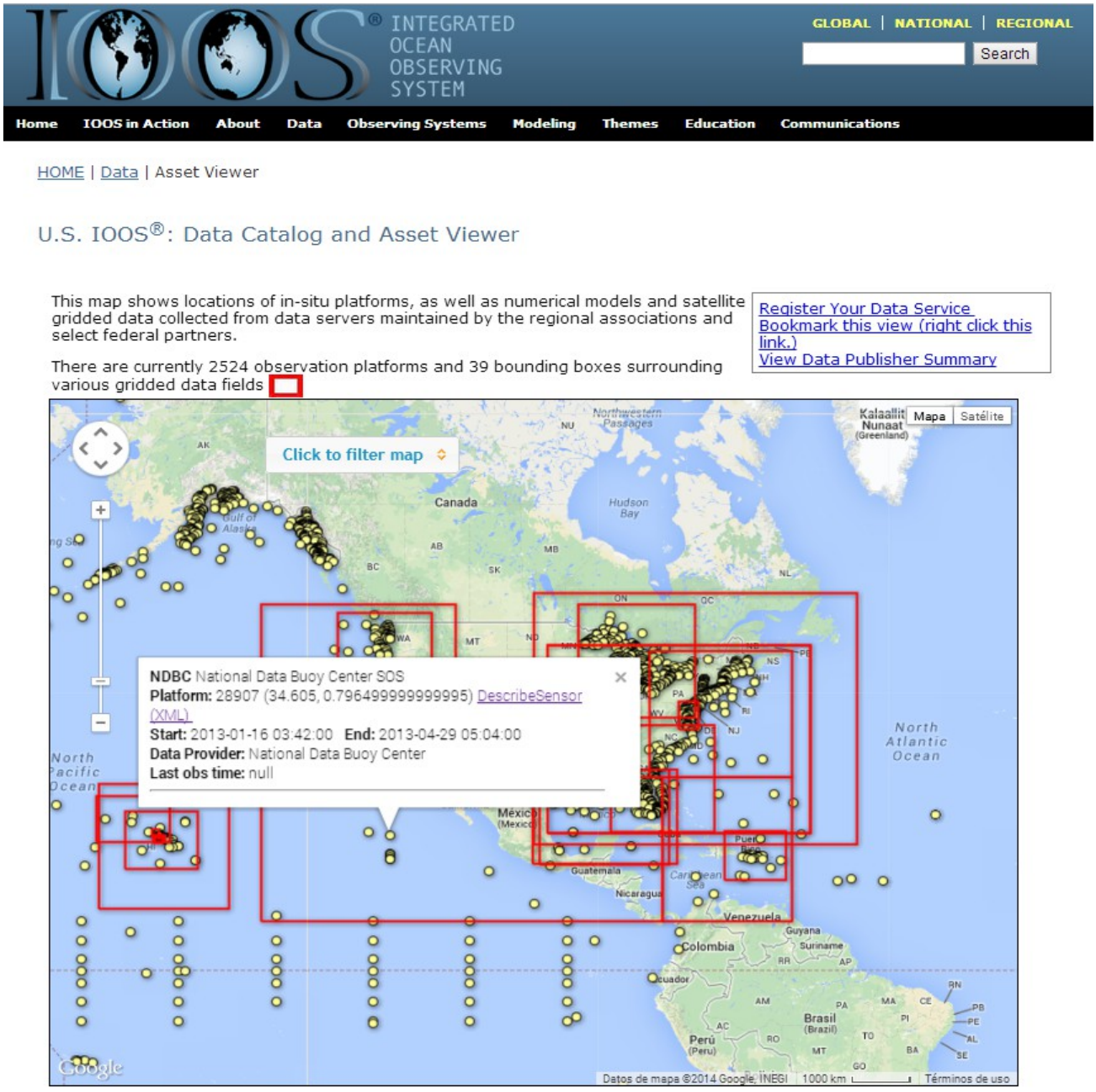


Figura 2.2. IOOS

2.1.3. SeaDataNet CDI

Las infraestructuras de SeaDataNet conectan 90 centros nacionales de datos oceanográficos y centros de datos marinos de 35 países ribereños para todos los mares europeos. Los centros de datos manejan grandes conjuntos de datos marinos y oceánicos. El objetivo principal es proporcionar un sistema integrado que ofrezca una visión general y permita el acceso a estos recursos de datos utilizando un enfoque de red distribuida. Ésto se logra mediante el servicio Common Data Index (CDI) que permite recuperar datos (temperatura, oxígeno, salinidad, nutrientes , ...) a partir de toda su red de plataformas de observación. La interfaz de consulta CDI permite realizar búsquedas por un conjunto de criterios. Entonces se muestra los resultados y se ubican en un mapa. Al hacer clic en el icono de la pantalla recupera los metadatos del conjunto de datos dando la información sobre el qué, dónde, cuándo, cómo y quién del conjunto de datos. La interfaz cuenta con un mecanismo de compras. Todos los usuarios pueden consultar libremente y navegar en el CDI; Sin embargo la presentación de las solicitudes de acceso a datos a través de la cesta de la compra requiere que los usuarios se registren en SeaDataNet.

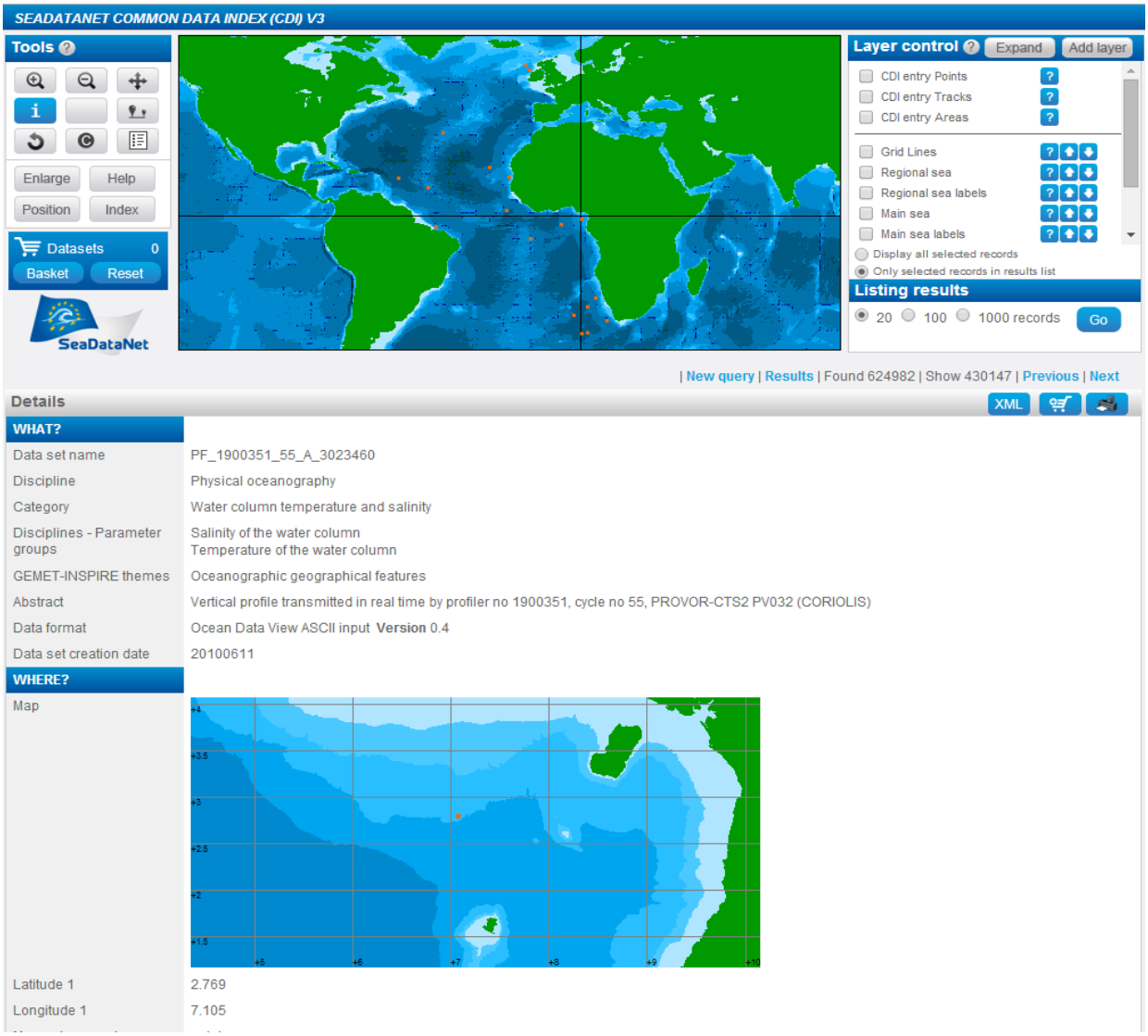


Figura 2.3. SeaDataNet CDI

2.1.4. Observing System Monitoring Center (OSMC)

La OSMC reúne las plataformas de observación oceanográfica (barcos, boyas, mareógrafos, etc) en un único sistema integrado. Esta herramienta proporciona monitoreo en tiempo real de los activos del sistema de observación integrados, que hace que los datos estén accesibles de forma fácil de usar y a la vez los unifica.

Los datos son servidos utilizando una serie de webservices como OPENDAP y SOS y formatos de archivo descargable, por lo que se puede acceder en los navegadores web y herramientas de análisis de escritorio.

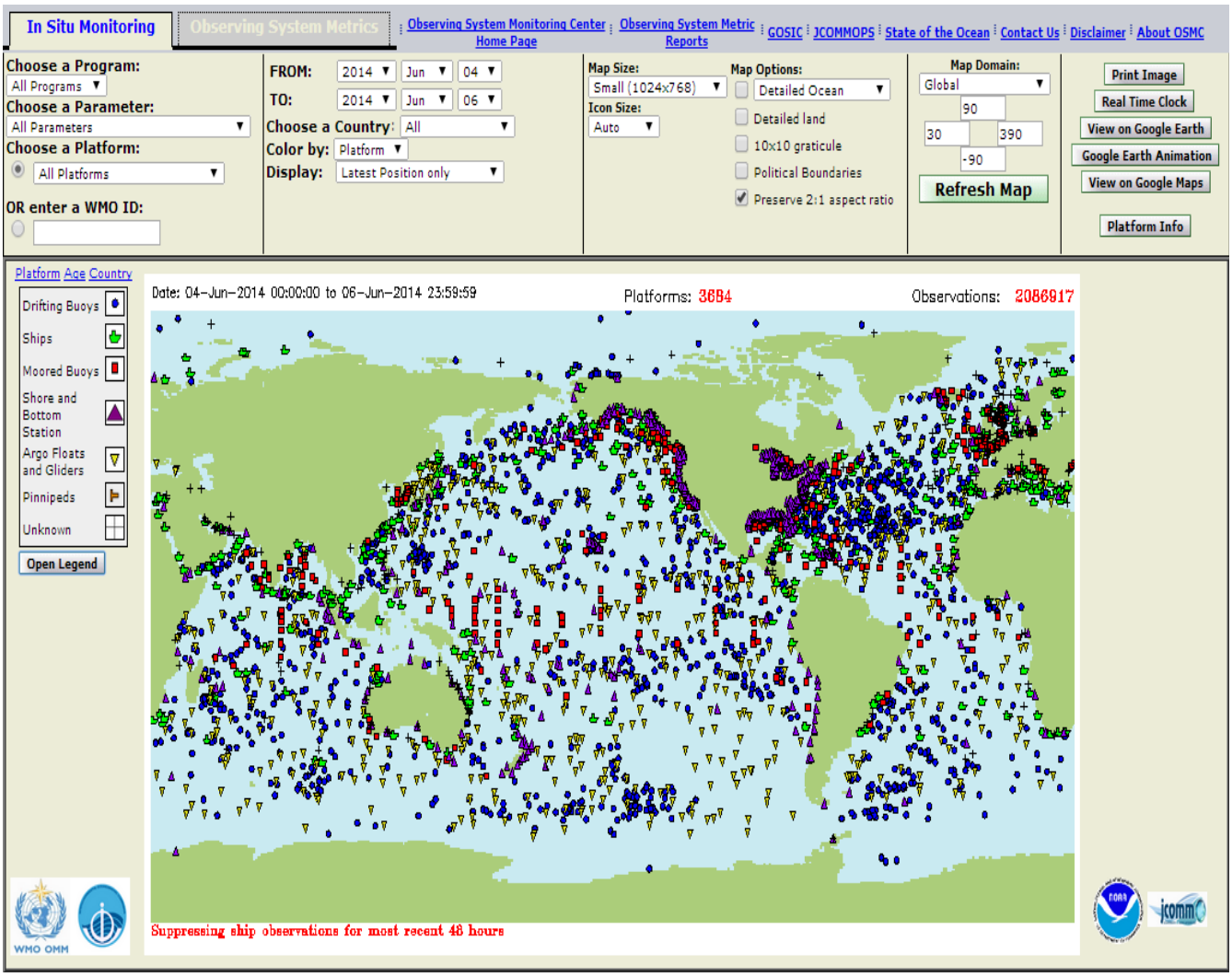


Figura 2.4. OSMC

2.1.5. Red Marino Marítima Macaronésica (R3M)

La Red Marino Marítima Macaronésica tiene la misión de aumentar la cantidad y calidad de la observación oceanográfica para intentar predecir los fenómenos que ocurren en el océano y sus repercusiones medioambientales y socioeconómicas.

Esta iniciativa ofrece una herramienta cuyo objetivo es hacer compatibles y accesibles todas las observaciones que se realizan en el entorno marino de la Macaronesia, con independencia de la institución o entidad que las realice. La herramienta R3M permite la accesibilidad a observaciones históricas de diferentes plataformas en el área macaronésica, como las observaciones "in situ" de instrumentos fijos y a la deriva, superficiales u ondulantes, así como observaciones remotas desde satélite. Estas observaciones están localizadas en un mapa que al seleccionar una de ellas redirige al usuario a las distintas fuentes externas donde la información es mostrada. Por lo tanto funciona a forma de geoíndice de observaciones que referencia a las fuentes originales.

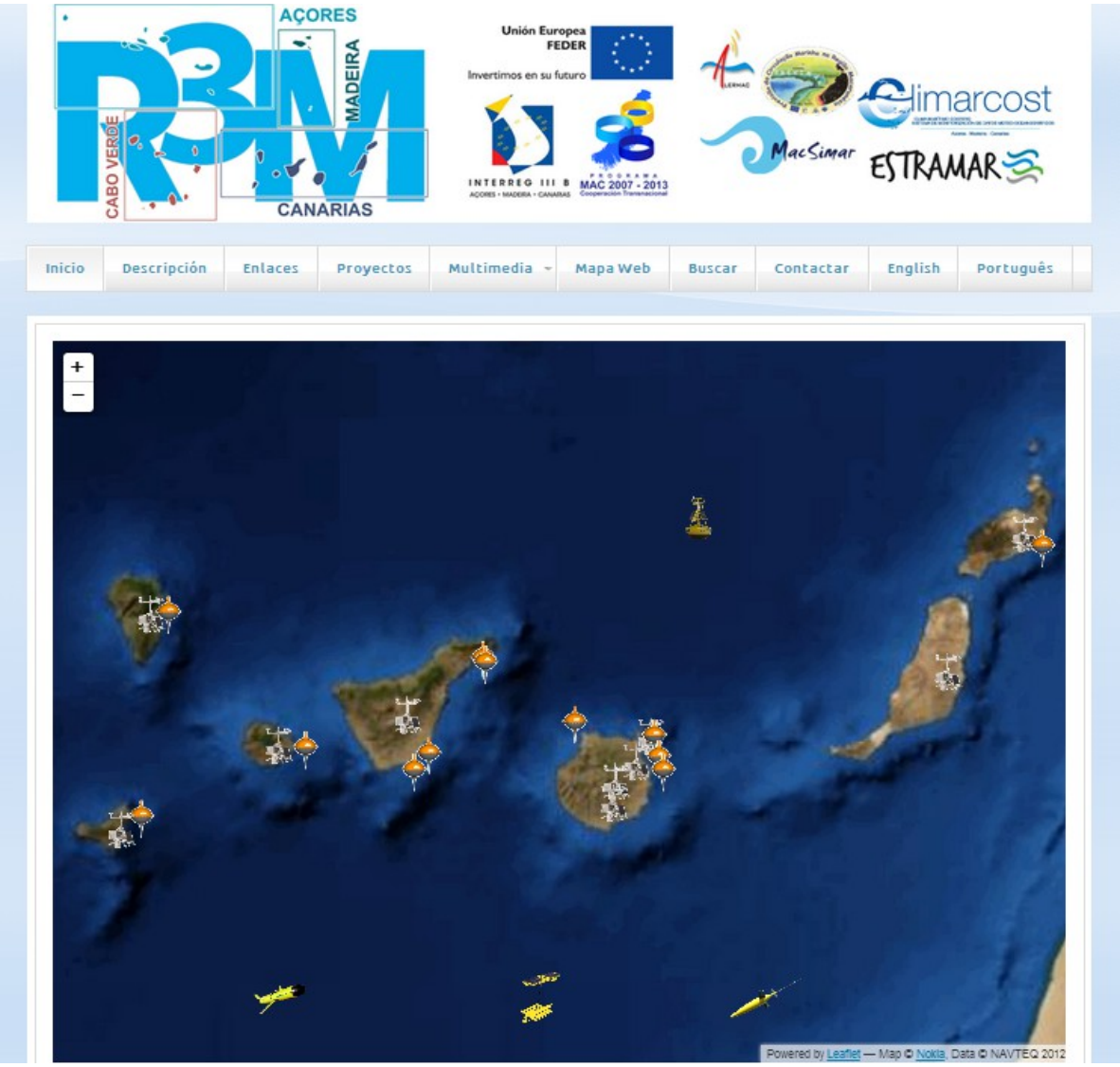


Figura 2.5. R3M

2.2. Sistemas de Información Geográfica (GIS)

Los sistemas de información geográfica son una integración de software que maneja datos geográficos sobre un hardware. Esta combinación está diseñada para capturar, almacenar, manipular analizar y desplegar toda la información geográfica con el fin de resolver problemas complejos. Estos sistemas están basados en la georreferenciación.

La georreferenciación es un concepto que representa la localización de un objeto espacial en un sistema de coordenadas determinado. Este objeto puede ser representado de diferentes formas, ya sea como punto, área, volumen, etc.

En una primera instancia, la georreferenciación era un concepto ligado a la sociedad científica, donde se entendía como la existencia de cosas en un espacio físico, donde se establecían relaciones entre una representación de un objeto y su proyección geográfica sobre un sistema de coordenadas.

Con el tiempo, con el avance de herramientas informáticas en este sentido como Google Maps y el acceso del público general a estas herramientas, el concepto dejó de ser exclusivamente del ámbito científico y se extendió a toda persona con ordenador y acceso a Internet.

A partir de ese momento la georreferenciación tiene un impacto sociológico muy importante donde gran cantidad de los datos que circulan en Internet dejan de ser meros datos y se convierten en datos que además están geolocalizados dando lugar al enriquecimiento de la información.

Con este escenario y con la proliferación de la web 2.0 se da el marco ideal para la aparición de una gran cantidad de sistemas de información geográfica con muy diversa temática. Siguen existiendo sistemas enfocados en la ciencia pero surgen otros muchos de índole más social como pudiera ser fotografías, vídeos, webcams, etc.

Muchas de estas herramientas normalmente no implementan sus propios sistemas de información geográficas o mapas virtuales sino que hacen uso de otros que dan la base geográfica y se le añaden capas con la información y representación de los datos que se quieren compartir.

A continuación se enumerarán los principales sistemas de información geográfica de los que se hacen uso en herramientas de este tipo y en las que se encuentra englobado el presente proyecto. Nos centraremos en aquellas que se puedan ejecutar en navegador web para que sea de utilidad a la naturaleza del proyecto.

2.2.1. Google Earth

Es un programa informático que muestra un globo virtual con base en la fotografía satelital. Fue desarrollada por Keyhole Inc, financiada por la Agencia Central de Inteligencia Americana y comprada por Google en 2004. Actualmente el programa está disponible en móviles, tablets, como aplicación de escritorio y para ejecución en navegador web.

El mapa está compuesto por superposición de imágenes por satélite, fotografía aérea, información geográfica proveniente de modelos de datos SIG de todo el mundo y modelos creados por ordenador. Los datos geoespaciales tridimensionales son soportados mediante los archivos Keyhole Markup Language o kml.

Google Earth permite introducir el nombre de un hotel, colegio o calle y obtener la dirección exacta, un plano o vista del lugar. También se pueden visualizar imágenes vía satélite del planeta. Por otra parte, ofrece características 3D como dar volumen a valles y montañas, y en algunas ciudades incluso se han modelado los edificios. Además, es posible compartir con otros usuarios enlaces, medir distancias geográficas, ver la altura de las montañas, ver fallas o volcanes y cambiar la vista tanto en horizontal como en vertical.

Figura 2.6. Google Earth

2.2.2. Here

Here, anteriormente Nokia Maps y Ovi Maps es un servicio de mapas gratuito de Nokia, creado en un principio para sus teléfonos móviles y dispositivos multimedia y que posteriormente fue implementada su versión web.

Los mapas de Here están programados en HTML5 y cuentan con un sistema de carga híbrido que almacena información en la caché del navegador para cargarlos más rápido. También permiten guardar fragmentos de mapas para acceder a ellos sin conexión a Internet. En Here se puede acceder a la vista 360º a pie de calle y a la vista aérea con modelos 3D de los edificios que se muestran mediante tecnología WebGL.

Here incluye una herramienta denominada Map Creator, que permite a cualquier persona editar el mapa completamente: añadir calles, definir su nombre y sentidos, situar locales y bifurcaciones, etc. Esta información está sujeta a moderación.

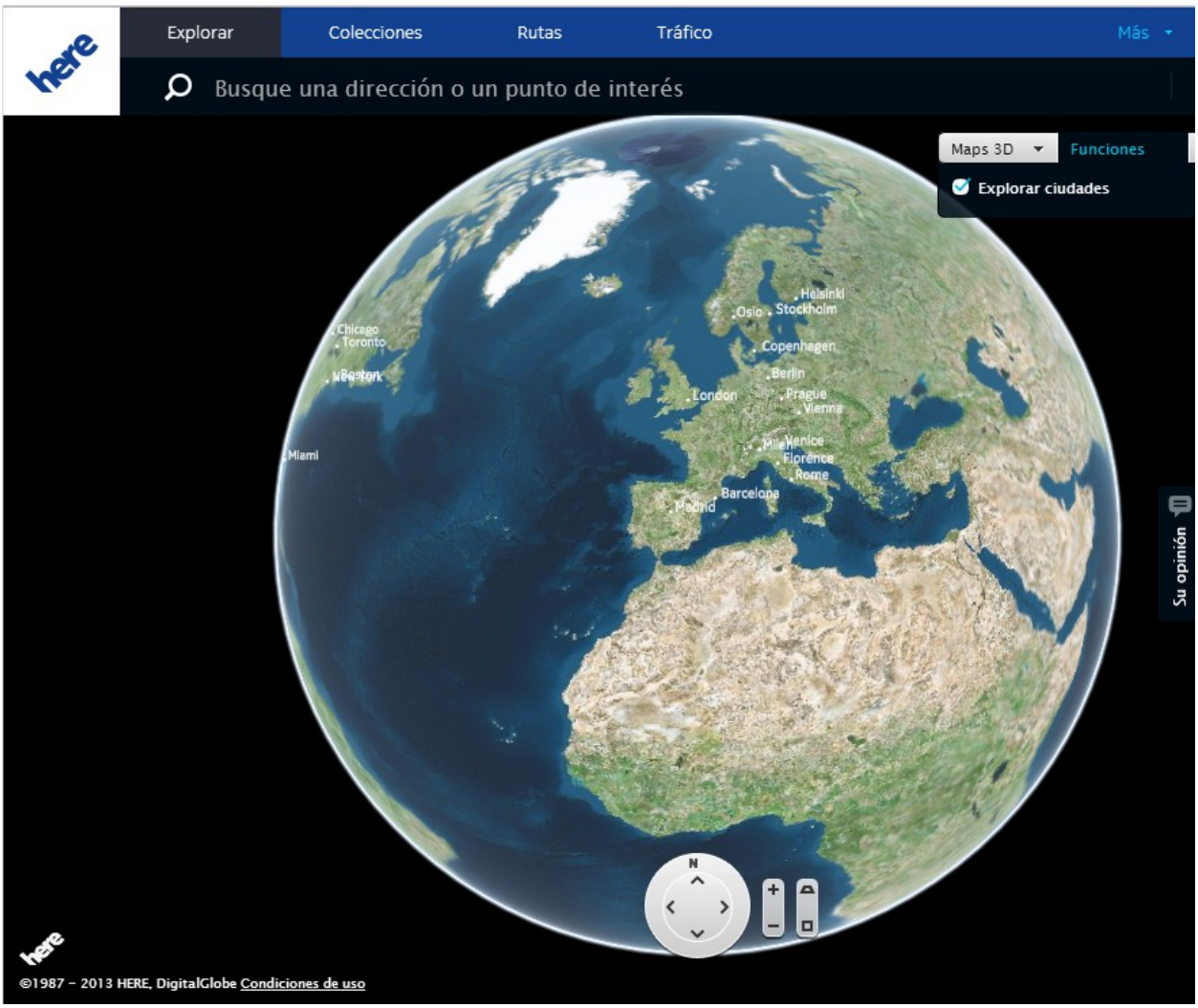


Figura 2.7. Here

2.2.3. Google Maps

Google Maps es un servidor de aplicaciones de mapas en la web que pertenece a Google. Ofrece imágenes de mapas desplazables, así como fotografías por satélite del mundo e incluso imágenes a pie de calle con Google Street View. En su buscador se pueden consultar calles, edificios e incluso negocios en una zona determinada.

Una de las herramientas más interesantes que ofrece Google Maps es el calculador de rutas, tanto a pie, en bicicleta, en transporte público, como en vehículo motorizado, donde el usuario marca un punto de origen y otro de destino y la aplicación ofrece una ruta visual y también un itinerario textual.

Google Maps ofrece API's para permitir a los desarrolladores realizar aplicaciones aprovechando la potencialidad de esta herramienta dando la posibilidad de integrar gran parte de sus características en un sitio web de terceros incluyendo la información que se desee.

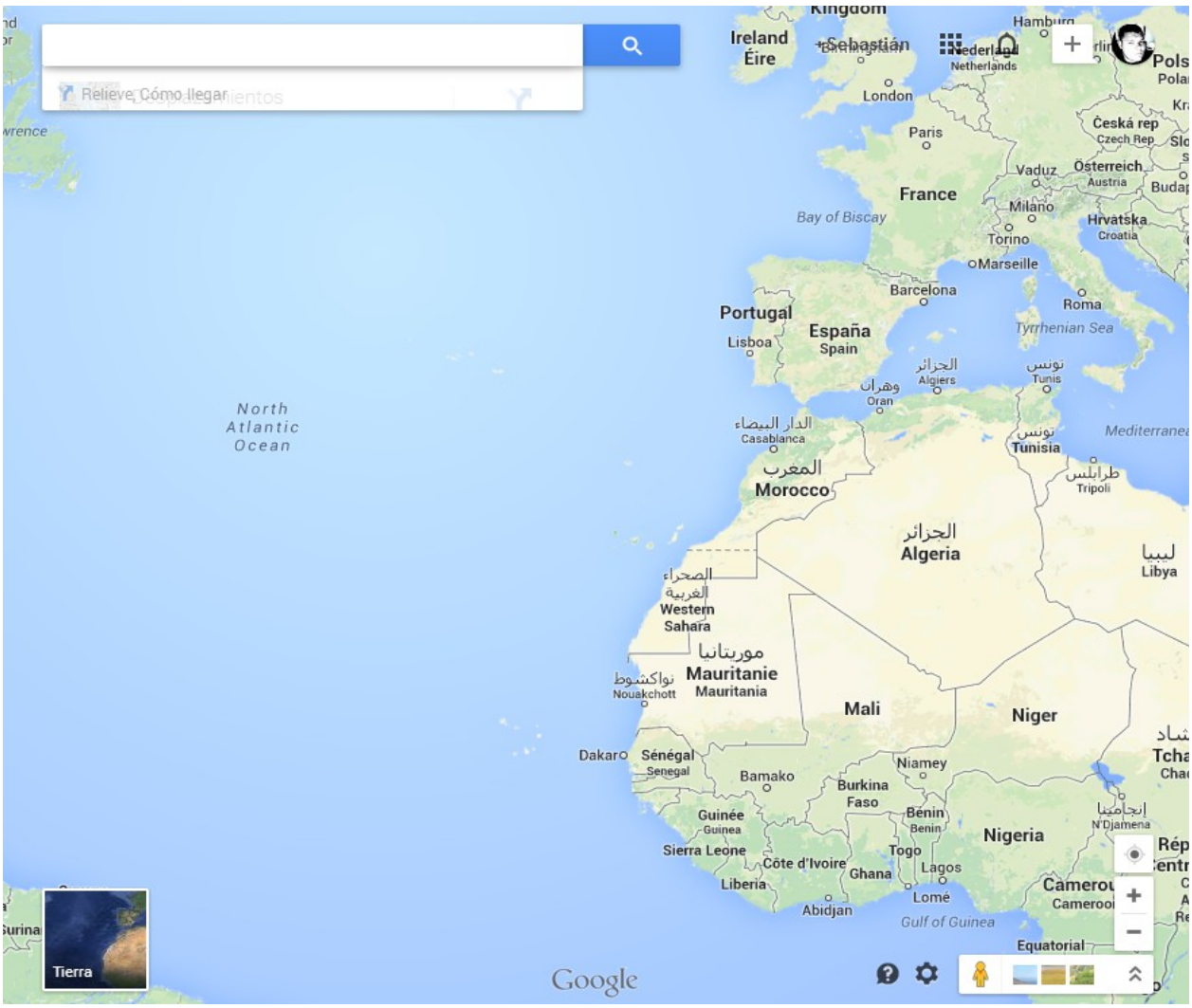


Figura 2.8. Google Maps

2.2.4. Bing Maps

Bing Maps (anteriormente Live Search Maps, Windows Live Maps y Windows Live Local) es un servicio de mapas web parte del motor de búsqueda Bing de Microsoft. Se basó en tecnologías existentes de Microsoft, como Microsoft MapPoint y TerraServer.

En Bing Maps los usuarios pueden examinar los mapas de calle topográficamente para muchas ciudades en todo el mundo. Los mapas incluyen ciertos puntos de interés integrados, tales como estaciones de metro, estadios, hospitales y otras instalaciones. También es posible navegar por puntos de interés públicos creados por los usuarios. La búsqueda puede cubrir colecciones públicas, empresas u otros tipos de negocios, lugares o personas. También incluye imágenes de satélite e imágenes aéreas. La característica de mapas 3D permite al usuario ver edificios en 3D, con la capacidad agregada de girar y el ángulo además de panorámica y zoom de inclinación. Además de todo ello permite calcular rutas, visualización del tráfico, etc.

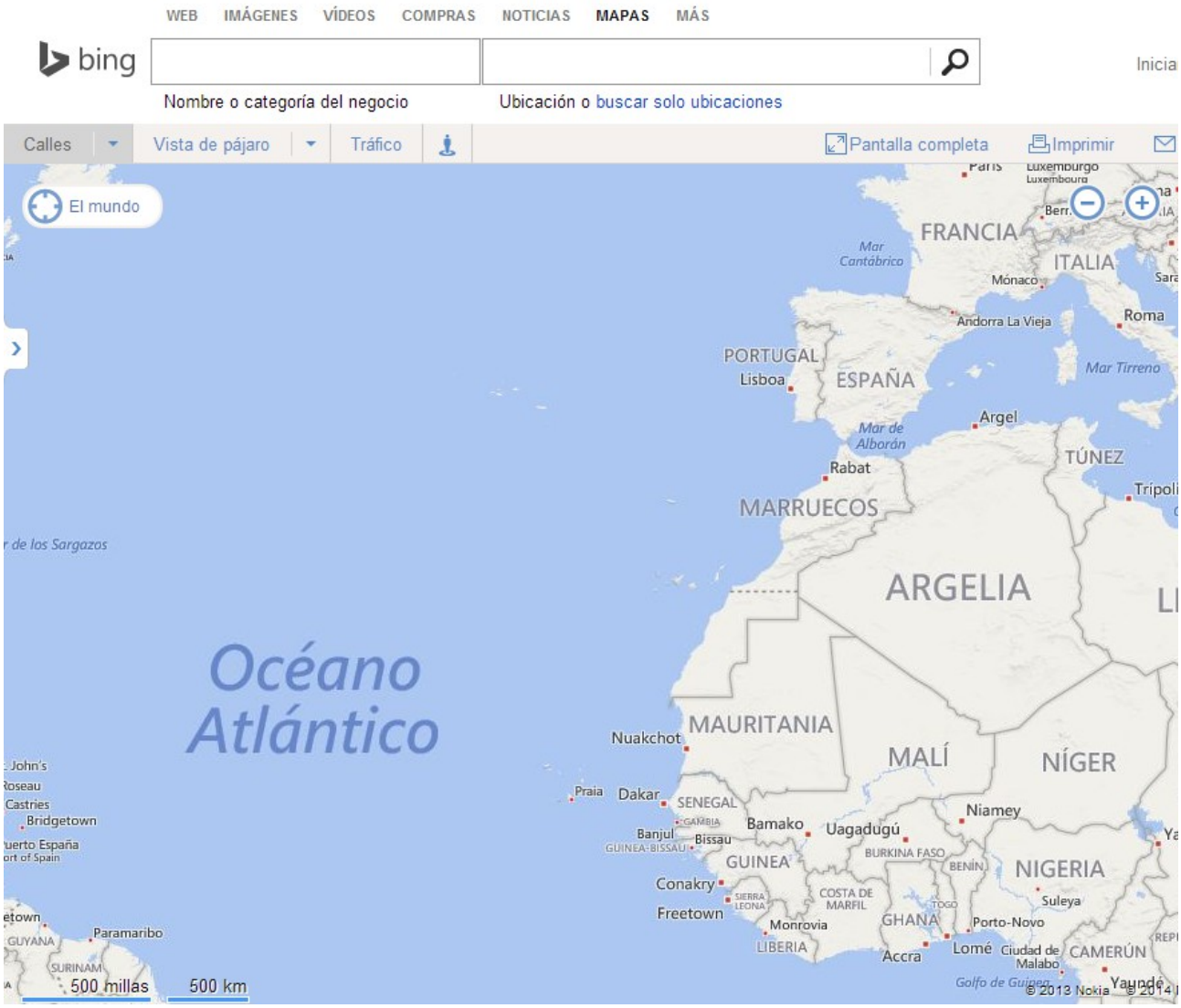


Figura 2.9. Bing Maps

2.2.5. ArcGIS

ArcGIS es el nombre de un conjunto de productos de software en el campo de los Sistemas de Información Geográfica producido y comercializado por ESRI. Se agrupan varias aplicaciones para la captura, edición, análisis, tratamiento, diseño, publicación e impresión de información geográfica.

Como sistema de información, ArcGIS es accesible desde clientes de escritorio, navegadores web, y terminales móviles que se conectan a servidores de departamento, corporativos o con arquitecturas de Cloud Computing. Para los desarrolladores, ArcGIS proporciona herramientas que les permitirán crear sus propias aplicaciones.

Figura 2.10. ArcGIS

2.2.6. OpenStreetMap

OpenStreetMap, también conocido como OSM, es un proyecto colaborativo para crear mapas libres y editables. Los mapas se crean utilizando información geográfica capturada con dispositivos GPS móviles, ortofotografías y otras fuentes libres. Esta cartografía, tanto las imágenes creadas como los datos vectoriales almacenados en su base de datos, se distribuye bajo licencia abierta Open Database License. Los usuarios registrados pueden subir sus trazas desde el GPS y crear y corregir datos vectoriales mediante herramientas de edición creadas por la comunidad. OpenStreetMap facilita los datos en bruto para su descarga desde su propia página web. Éstos pueden ser modificados para cada proyecto así como presentados con estilos de renderizado personalizados.

A partir de los datos del proyecto OpenStreetMap no sólo se pueden producir mapas de carreteras, sino también para la creación de mapas de senderismo, mapas de rutas ciclistas, mapas náuticos, mapas de estaciones de esquí, etc. También se usan en aplicaciones para el cálculo de las rutas óptimas para vehículos y peatones. Gracias a su licencia abierta los datos brutos son de libre acceso para el desarrollo de otras aplicaciones.

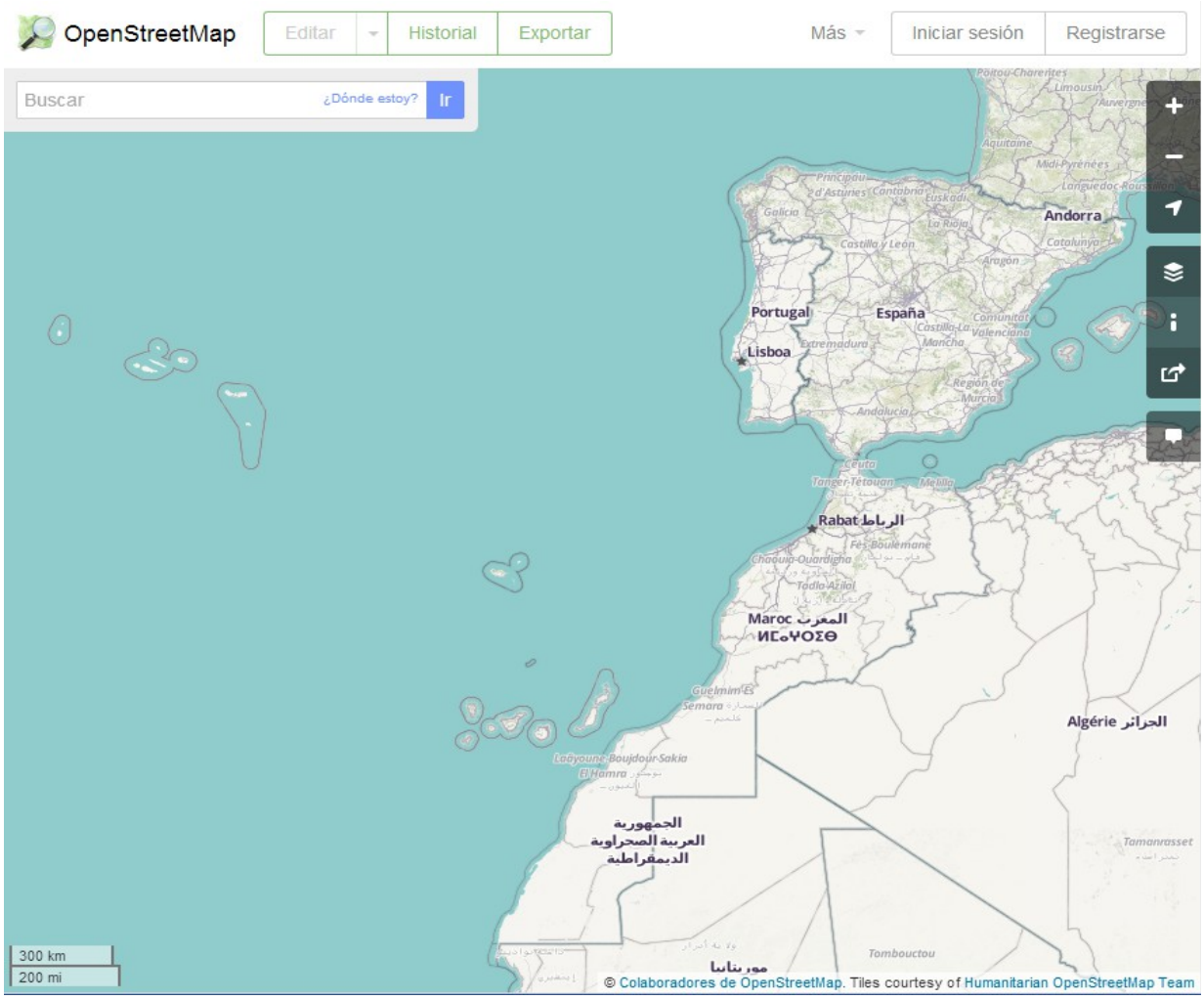


Figura 2.11. OpenStreetMap

2.2.7. OpenLayers

OpenLayers es una biblioteca de JavaScript de código abierto bajo una derivación de la licencia BSD para mostrar mapas interactivos en los navegadores web. OpenLayers ofrece un API para acceder a diferentes fuentes de información cartográfica en la red: Web Map Services, Mapas comerciales, Web Features Services, distintos formatos vectoriales, mapas de OpenStreetMap, etc.

Inicialmente fue desarrollado por MetaCarta en junio del 2006. Desde noviembre del 2007 este proyecto forma parte de los proyectos de Open Source Geospatial Foundation. Actualmente el desarrollo y el soporte corre a cargo de la comunidad de colaboradores.

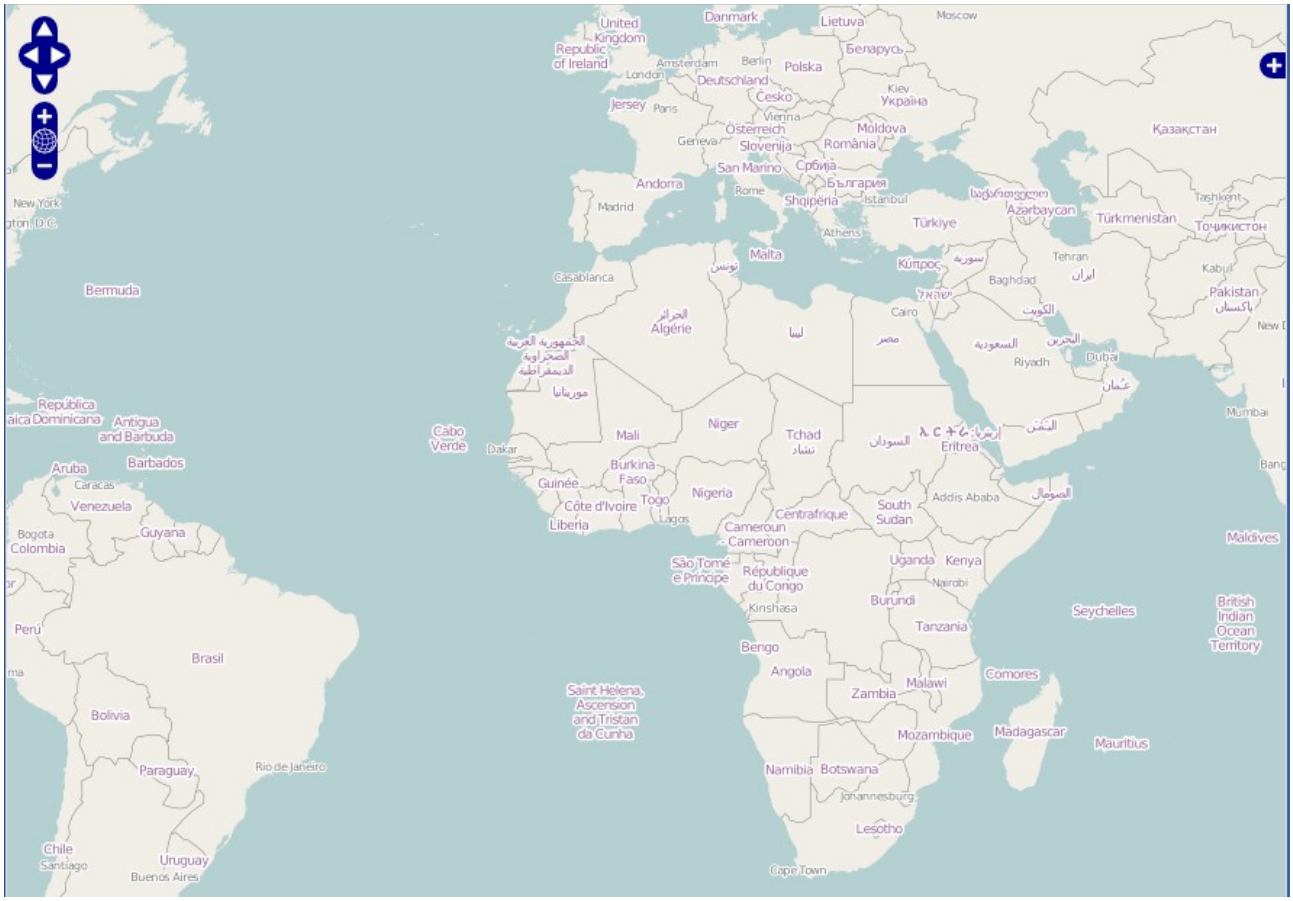


Figura 2.12. OpenLayers

Existe otra herramienta llamada ReadyMap, que es un nuevo conjunto de herramientas de código abierto en JavaScript para la presentación de mapas 3D basada OpenLayers. ReadyMap utiliza WebGL y el elemento Canvas de HTML5.

Capítulo 3: Planificación del Proyecto

En este capítulo se explica la planificación del proyecto. En él se detalla la metodología de desarrollo que se ha usado y porqué, el plan de trabajo llevado a cabo y su temporización y por último un presupuesto detallado del coste del proyecto.

3.1. Metodología de Desarrollo

Respecto a la metodología a utilizar se utilizarán las técnicas de desarrollo software propias del Proceso Unificado de Desarrollo (PUD).

3.1.1. Proceso Unificado de Desarrollo

El Proceso Unificado de Desarrollo dota de un marco de trabajo genérico que puede especificarse para una gran abanico de sistemas software con diferentes áreas de aplicación, para distintas configuraciones de organización y para cualquier tamaño de proyecto.

El Proceso Unificado de Desarrollo utiliza el Lenguaje Unificado de Modelado (UML) como parte esencial para el modelado del producto software en cada una de sus etapas.

El Proceso Unificado se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Al final de cada uno de ellos se obtiene una versión final del producto, que no sólo satisface ciertos casos de uso, sino que está lista para ser entregada y puesta en producción. En caso de que fuese necesario publicar otra versión, deberían repetirse los mismos pasos a lo largo de otro ciclo.

El mantra del Proceso Unificado de Desarrollo se basa en cuatro ideas claves:

- **Iterativo e incremental.** Está compuesto por cuatro fases denominadas Inicio, Elaboración, Construcción y Transición. Cada una de estas fases están compuestas por una serie de iteraciones que resulta en un incremento del producto al que va añadiendo funcionalidades en cada una de las iteraciones. Cada una de estas iteraciones pasa por cada etapa del ciclo de vida clásico del software (requisitos, análisis, diseño, implementación y pruebas) con lo que al final de cada iteración se tiene un producto usable y testado que será ampliado en iteraciones posteriores.

- **Dirigido por casos de uso.** Los casos de uso son una representación de los requisitos funcionales del software. El conjunto de todos ellos constituyen el modelo de casos de uso con el que se define la funcionalidad completa del sistema. Con ello se consigue definir lo que debe hacer el sistema y lo que un usuario puede hacer con él. Esta herramienta guía el proceso de desarrollo donde en cada iteración se toman un numero de casos de usos limitados y se van transformando en especificaciones de análisis y posteriormente de diseño para finalmente convertirse en código fuente dando lugar a una aplicación con tantas funcionalidades como casos de uso se hayan implementado hasta el momento.

- **Centrado en la arquitectura.** La arquitectura de un software comprenden un conjunto de diferentes vistas del sistema en construcción. Esta surge del problema que se pretende solucionar, de la plataforma donde debe funcionar el software, de los recursos reutilizables desarrollados anteriormente, de los sistemas heredados y de los requisitos no funcionales. Todo esto sirve de ayuda al arquitecto para fijar los objetivos como la capacidad de adaptación al cambio y la reutilización. La arquitectura y los casos de uso deben evolucionar en paralelo habiendo interacción entre ambos para que los casos de uso encajen en la arquitectura y sea posible la implementación de todos y cada uno de ellos.

- **Enfocado en los riesgos.** Se requiere que se identifiquen los riegos críticos del proyecto lo antes posible, durante las primeras etapas del ciclo de vida, para que en caso de que alguno comprometa el éxito del proyecto lo haga cuando la inversión sea mínima.

3.1.2. Fases del Proceso Unificado de Desarrollo

Como se ha comentado en el apartado anterior, cada ciclo se compone de varias fases, y dentro de cada una de ellas, los directores o los desarrolladores pueden descomponer adicionalmente el trabajo en iteraciones, con sus incrementos resultantes. Cada fase termina con un hito, determinado por la disponibilidad de un conjunto de artefactos, modelos o documentos.

Las iteraciones de cada fase se desarrollan a través de las actividades de identificación de requisitos, análisis, diseño, implementación, pruebas e integración.

Fase de Inicio

Suele ser la fase más corta del desarrollo. En esta fase se realizan las siguientes tareas:

- Desarrollar una descripción del producto final y presentar el análisis de negocio.
- Realizar una identificación inicial de riesgos.
- Establecer las principales funciones del sistema para los usuarios más importantes, la arquitectura a grandes rasgos y un plan de proyecto.

La fase de inicio termina con el hito de los objetivos del desarrollo.

Fase de Elaboración

Durante esta fase deberían capturarse la mayoría de requisitos del sistema, aunque los objetivos principales son tratar los riesgos ya identificados y establecer y validar la base de la arquitectura del sistema. Esta base se llevará a cabo a través de varias iteraciones, y servirá de punto de partida para la fase de construcción.

La fase de elaboración termina, por tanto, al alcanzar el hito de la arquitectura del sistema.

Fase de Construcción

Es la fase más larga del proyecto, y completa la implementación del sistema tomando como base la arquitectura obtenida durante la fase de elaboración. A partir de ella, las distintas funcionalidades son incluidas en distintas iteraciones, al final de cada una de las cuales se obtendrá una nueva versión ejecutable del producto.

Por tanto, esta fase concluye con el hito de obtención de una funcionalidad completa, que capacite al producto para funcionar en un entorno de producción.

Fase de Transición

En la fase final del proyecto se lleva a cabo el despliegue del producto en el entorno de los usuarios, lo que incluye la formación de éstos.

Durante esta fase se evoluciona desde la fase beta a una versión final, se resuelven incidencias en la implantación e integración, y si existen, se clasifican aquéllas que podrían justificar una nueva versión del producto.

Esta fase concluye con el hito de publicación del producto.

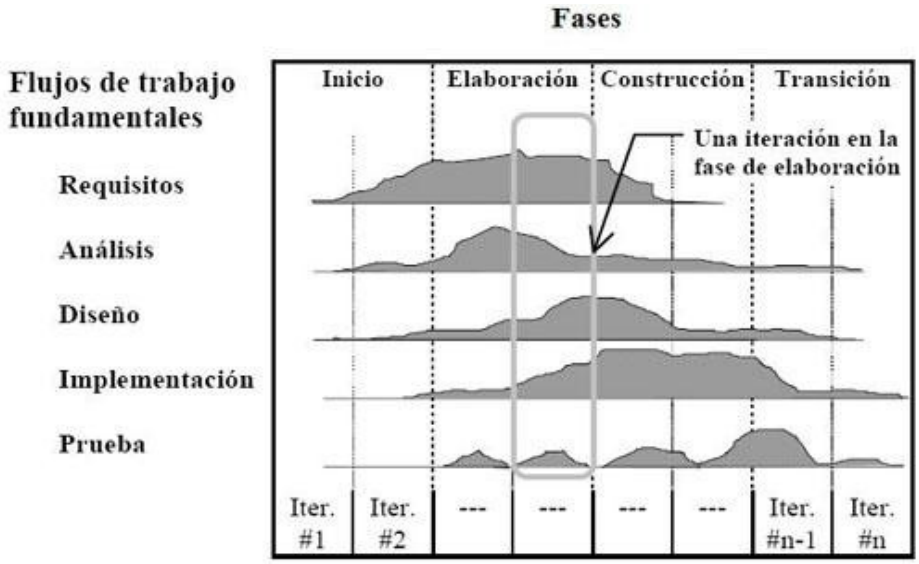


Figura 3.1. Fases del Proceso Unificado de Desarrollo.

3.1.3. Ventajas y Desventajas del Proceso Unificado de Desarrollo

Ventajas

- Se obtiene productos funcionales y testados en tiempos razonables.
- Se identifican los riesgos críticos pronto, evitando fracasos en fases tardías del proyecto.
- Es muy tolerante a cambios.
- Fácil adición de funcionalidades posteriores.
- Alta reutilización de componentes

Desventajas

- La evaluación de riesgos es compleja.
- El cliente debe ser capaz de describir a un gran nivel de detalle.
- Excesiva flexibilidad para algunos proyectos

3.2. Planificación Temporal

A continuación se explicará el plan de trabajo llevado a cabo durante el proyecto así como el desglose de las tareas realizadas en cada fase e iteración con el número de horas estimadas que se debían invertir en ellas.

3.2.1. Plan de Trabajo

Tal y como se define en el Proceso Unificado de Desarrollo, y explicado en el apartado anterior, el desarrollo del proyecto se dividirá en 4 fases. Cada una estas fases, a su vez se dividen en iteraciones de aproximadamente el mismo tiempo (alrededor de 100 horas/iteración).

Al final de cada una de estas iteraciones se planifica una reunión en la que participamos por parte de PLOCAN: el responsable de proyectos, Eduardo Quevedo, el responsable del Departamento de Informática y Telecomunicaciones, David Horat y el responsable del Observatorio, Eric Delory; y por parte de la Escuela de Ingeniería Informática de la Universidad de Las Palmas de Gran Canaria: el profesor y tutor del proyecto, Javier Sánchez y yo.

En cada una de estas reuniones se tratan y validan los hitos de la iteración que se acaba de finalizar y se establecen los siguientes para la próxima iteración.

La fase inicial está compuesta por dos iteraciones en las que en una primera se estudian las herramientas a utilizar en el proyecto. Y una segunda en la que se hace la mayor parte de recabado de información a través de entrevistas y cuestionarios, se define y negocia la lista de características y se redactan los documentos que establecerán las bases del proyecto.

La fase de elaboración se centrará en el análisis y el diseño de cada uno de los tres módulos por lo que está compuesta la aplicación. Se invertirá una iteración por cada módulo.

La fase de construcción también estará compuesta por tres iteraciones en donde se llevarán a cabo la implementación y prueba de cada uno de los tres módulos. Al final de esta fase se obtiene una versión beta de la aplicación.

Finalmente la fase de transición consta de una única iteración en donde se montará un entorno de pruebas y sobre el cuál se realizarán todas las pruebas beta por parte de la organización.

En el siguiente subapartado se detalla cada una de las actividades por iteración y fase.

3.2.2. Temporización

Fases/Actividades	Horas
Fase Inicial	
Iteración 1: Puesta en marcha del PFC	
• Valoración de las herramientas IVS-3D, Google Sketch-Up y Google Earth	20
• Estudio de KML y la API de Google Earth	40
• Estudio Python y Django	40
Iteración 2:	
• Redacción de cuestionario de entrevista	4
• Redacción del informe sobre el estado de ejecución del PFC	4
• Redacción del documento de descripción de los entregables del PFC	4
• Redacción del documento de requisitos y criterios de éxito del PFC	4
• Redacción del plan de gestión de calidad del PFC	4
• Redacción del plan de gestión de riesgos	4
• Redacción del plan de gestión de rr.hh. y comunicaciones	4
• Redacción del presupuesto	4
• Generar la lista de características inicial	8
• Generar el modelo de dominio inicial	8
• Generar el modelo de casos de uso inicial	16
• Desarrollo de un prototipo inicial	36
Fase de Elaboración	
Iteración 3: Módulo Cliente	
• Refinado del modelo de dominio	4
• Refinado de lista de características	4

• Refinado de casos de uso del Módulo Cliente	4
• Prototipo Cliente	8
• Diagramas de paquetes del modelo de análisis	15
• Diagrama de clases del Módulo Cliente - Análisis	15
• Diseño de la BD del Módulo Cliente	15
• Diagrama de despliegue	15
• Documentación	20
Iteración 4: Módulo Dispositivos	
• Refinado del modelo de dominio	4
• Refinado de lista de características	4
• Refinado de casos de uso del Módulo Dispositivos	4
• Prototipo Dispositivos	8
• Diagrama de Clases del Módulo Dispositivos - Análisis	20
• Diseño de la BD del Módulo Dispositivos	20
• Diagrama de paquetes del modelo de diseño	20
• Documentación	20
Iteración 5: Módulo Usuarios	
• Refinado del modelo de dominio	4
• Refinado de lista de características	4
• Refinado de casos de uso del Módulo Usuarios	4
• Prototipo Usuarios	8
• Diagrama de Clases del Módulo Usuarios - Análisis	20
• Diseño de la BD del Módulo Usuarios	20
• Diagrama de diseño arquitectónico	20
• Documentación	20

Fase de Construcción	
Iteración 6: Construcción Cliente	
• Refinado Diagramas de paquetes del modelo de análisis	2
• Refinado Diagramas de Clases del Módulo Cliente	2
• Refinado Diagramas de despliegue	2
• Implementación de las tablas de BD de Cliente	14
• Implementación del Módulo Cliente	64
• Prueba del Módulo Cliente	8
• Documentación	8
Iteración 7: Construcción Dispositivos	
• Refinado Diagramas de Clases del Módulo Dispositivos	2
• Refinado Diagramas de paquetes del modelo de diseño	2
• Implementación de las tablas de BD de Dispositivos	16
• Implementación del Módulo Dispositivos	64
• Prueba del Módulo Dispositivos	8
• Documentación	8
Iteración 8: Construcción Usuarios	
• Refinado Diagramas de Clases del Módulo Usuarios	2
• Refinado Diseño Arquitectónico	2
• Implementación de las tablas de BD de Usuarios	16
• Implementación del Módulo Usuarios	64
• Prueba del Módulo Usuarios	8
• Documentación	8

Iteración 9: Integración del sistema completo	
• Pruebas exhaustivas del software completo	40
• Versión Beta de la aplicación	10
• Generación del manual de instrucciones de la aplicación	30
• Documentación	20
Fase de Transición	
Iteración 10: Transición	
• Instalación y configuración del servidor final	20
• Volcado de la base de datos	8
• Instalación de la aplicación	12
• Pruebas Beta	40
• Documentación	20
Total horas	**1000**

3.3. Presupuesto

En este apartado se detalla la estimación de gastos correspondientes a los subapartados de recursos humanos, recursos hardware, recursos materiales y gastos de documentación del proyecto.

3.3.1. Recursos Humanos

Dado que la planificación temporal nos da el número de horas de trabajo previstas, la confección de los costos laborales es sencilla. Vamos a partir de la base de que las personas que trabajan en el proyecto lo hacen en calidad de empleados de una empresa.

Debemos tener en cuenta que el número de horas de trabajo efectivo de un trabajador ronda unas 1.575 horas laborables al año (descontando los días de vacaciones no laborables). Por otro lado

hay que distinguir lo que recibe el trabajador como salario bruto y lo que éste le cuesta a la empresa (teniendo en cuenta Seguridad Social, etc). Normalmente a la empresa, el trabajador le supone alrededor de 1,5 veces su nómina.

Como referencia, el alumno tiene un sueldo de 1.000 euros al mes con 2 pagas extras y el tutor (personal con más experiencia) cobra el doble (2.000 euros). Por tanto lo que cuesta al año cada uno a la empresa es :

Persona	Perfil	Nº Pagas	Neto/Mes	Coeficiente	Bruto/Año
Alumno	Analista Programador	14	1.000 €	1,5	21.000 €
Tutor	Jefe de Proyecto	14	2.000 €	1,5	42.000 €

Por tanto, si cada uno de ellos trabaja 1.575 horas al año, el coste/hora de cada uno es :

Persona	Perfil	Bruto/Año	Horas/Año	Coste / Hora
Alumno	Analista Programador	21.000 €	1.575	13,33 €/Hora
Tutor	Jefe de Proyecto	42.000 €	1.575	26,66 €/Hora

Finalmente los costes laborales se calcularían multiplicando el coste/hora de cada uno por el número de horas estimadas en la planificación.

Alumno: 13,33 * 1000 = 13.333 euros

Para calcular los gastos en el tutor suponemos que dedica aproximadamente 1,5 horas por cada semana de trabajo del alumno a jornada completa. Por lo tanto el número de horas que el tutor emplea en el proyecto es de :

Tutor: (1000 horas / 40 horas) * 1,5 horas ≈ 37,5 horas

Por lo tanto:

Tutor: 26,66 * 37,5 = 999,75 euros

Así que el resumen de los costes laborales resulta de la siguiente manera:

Persona	Perfil	Coste
Alumno	Analista Programador	13.333 €
Tutor	Jefe de Proyecto	999,75 €

Total 14.332,75 €

3.3.2. Recursos Hardware

Son los costes del material necesario para la ejecución del proyecto. Hay que tener en cuenta la vida útil del material y el grado de uso exclusivo para el proyecto. En base a estos criterios se establecen los siguientes costes.

Recurso	Precio	Porcentaje Uso	Porcentaje Vida Útil	Coste
Portatil Dell Vostro 3500	890 €	75%	35%	233,62 €
Monitor Acer X243HQ	239 €	75%	35%	62,74 €
Multifunción	180€	25%	35%	15,75 €
			Total	**312,11 €**

3.3.3. Recursos Materiales

Son los gastos generados por el uso cotidiano de las cosas, como el papel, tintas de impresoras y demás material de oficina.

Concepto	Coste
Material fungible	50 €

3.3.4. Gastos de Documentación

En el caso de los PFC se debe entregar una copia de la memoria del PFC en papel para la secretaría del centro y dos copias en formato digital, una para la secretaría y otra para la biblioteca. Hay que tener en cuenta el coste de la copia y encuadernación del ejemplar del documento final de PFC que hay que entregar. También hay que contar con las dos copias digitales.

Concepto	Cantidad	Precio	Coste
Página impresa	250	0,15 €	37,50 €
Encuadernación	1	6,00 €	6,00 €
CD´s	2	1,20 €	2,40 €
		Total	**45,90 €**

3.3.5. Resumen Presupuesto Total

Una vez calculado el coste de cada uno de los recursos utilizados en el proyecto, se le añade un 10% en concepto de costes indirectos para subsanar los gastos en conceptos no calculados como puede ser consumo eléctrico, consumo de agua y otros costes indirectos de difícil cálculo. Además con ello se cubre también los riesgos del proyecto.

Tipo	Concepto	Coste	Coste Total
Recursos Humanos	Analista Programador	13.333,00 €	**14.332,75 €**
	Jefe de Proyecto	999,75 €	
Recursos Hardware	Portátil Dell Vostro 3500	233,62 €	**312,11 €**
	Monitor Acer X243HQ	62,74 €	
	Multifunción	15,75 €	
Recursos Materiales	Material fungible	50,00 €	**50 €**
Gastos de Documentación	Página impresa	37,50 €	**45,90 €**
	Encuadernación	6,00 €	
	CD´s	2,40 €	
Costes Indirectos	Costes indirectos (10%)	1474,08 €	**1474,08 €**
		Total	**16.214,84 €**

Capítulo 4: Tecnologías y Herramientas

En el presente capítulo se enumerarán, por un lado, cada una de las tecnologías usadas a lo largo del proyecto, y por otro, los recursos tanto hardware como software de que se hicieron uso.

4.1. Tecnologías

Para la realización del proyecto se han utilizado gran cantidad de tecnologías, todas ellas bajo licencias de código abierto. Se han clasificado en grandes familias siendo agrupadas por su naturaleza y objetivos comunes: lenguajes de programación, frameworks, lenguajes de visualización y maquetado, sistemas de gestión de bases de datos, formatos de datos, sistemas de información geográfica, gráficas.

4.1.1. Lenguajes de Programación

Python es un lenguaje de programación de alto nivel multiparadigma que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa un fuerte tipado dinámico y es multiplataforma. Su filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible. Se ha elegido Python como lenguaje de backend.

JavaScript es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, en bases de datos locales al navegador. Este ha sido el lenguaje elegido como lenguaje del lado del cliente.

4.1.2. Frameworks de Desarrollo

Django es un framework web de código abierto escrito en Python que permite el desarrollo rápido y el diseño limpio y pragmático. Se basa en el paradigma MVC (Model View Controller), aunque con alguna pequeña particularidad adoptando el nombre de MTV (Model Template View) pero que a fin de cuentas se basa en el mismo principio.

Django pone énfasis en la reusabilidad, conectividad y extensibilidad de componentes, el desarrollo rápido y el principio "No te repitas" (DRY, del inglés Don't Repeat Yourself)

jQuery es un framework de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con AJAX a páginas web.

Sus principales características son: Selección de elementos DOM, interactividad y modificaciones del árbol DOM, soporte a eventos, manipulación de la hoja de estilos CSS, efectos y animaciones, animaciones personalizadas, soporte AJAX y existe un gran número de extensiones basadas en él.

4.1.3. Lenguajes de Visualización y Maquetado

HTML, siglas de HyperText Markup Language («lenguaje de marcado de hipertexto»), es el lenguaje con el que se escribe la estructura y la semántica del contenido de un documento Web. Permite describir la estructura y el contenido en forma de texto, además de complementar el texto con objetos tales como imágenes.

CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.

4.1.4. Sistema de Gestión de Bases de Datos

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. MySQL ofrece un amplio subconjunto de SQL, transacciones y claves foráneas, conectividad segura, replicación y búsqueda e indexación de campos de texto.

4.1.5. Formatos Estándar de Datos

El formato NetCDF (Network Common Data Format) fue creado por UNIDATA como formato estándar para que sea usado en algunos de sus softwares que ofrece a la comunidad científica. La característica de este formato es que contiene la suficiente información para poder saber qué clase de data se encuentra en el archivo (tipo de variable, unidades, dimensiones, institución que la creo, etc) a diferencia de otros formatos que necesitan de un archivo adicional para su correcta interpretación.

SensorML especifica los modelos y la codificación XML que proporcionan un marco dentro del cual pueden ser definidas las características geométricas y dinámicas, y observación de los sensores y sistemas de sensores.

KML es un lenguaje XML centrado en la visualización geográfica, incluyendo la anotación de mapas e imágenes. La visualización geográfica incluye no sólo la presentación de los datos gráficos en el mundo, sino también el control de la navegación del usuario.

4.1.6. Sistema de Información Geográfica

El plugin de Google Earth es un complemento para navegadores web similar a un Sistema de Información Geográfica (SIG). Permite visualizar imágenes en 3D del planeta, combinando imágenes de satélite, mapas y el motor de búsqueda de Google que permite ver imágenes a escala de un lugar específico del planeta sobre un globo terráqueo en 3D desde un navegador web.

4.1.7. Gráficas

Google Chart es una herramienta por la cual se nos permite crear, vía petición http a la API, unas gráficas que podemos insertar en un documento web. Para ello se debe generar una URL pasando una serie de parámetros en la misma y que como respuesta se obtiene una gráfica. Estas gráficas pueden ser de diferentes tipos y además se le puede añadir control, dinamismo y un amplio conjunto de configuraciones.

4.2. Herramientas

En los siguientes subapartados se enumeran las herramientas que han sido necesarias para la realización del proyecto, tanto hardware como software.

4.2.1. Hardware

Como equipo de trabajo para llevar a cabo el proyecto se ha utilizado un ordenador portátil y para una mayor comodidad se le ha añadido un segundo monitor, teclado y ratón externos. La características de cada uno ellos son las siguientes:

- Ordenador portátil Dell Vostro 3500
  - Procesador: Intel Core i5 M460 2,53GHz
  - Memoria RAM: 4 GB DDR3
  - Gráficos: NVIDIA GeForce 310M
  - Disco Duro: Toshiba SATA 320GB
- Monitor Acer X243HQ de 24"
- Teclado y ratón Logitech

4.2.2. Software

Con un soporte físico como el citado en el subapartado anterior, pero sin un software que lo complemente no se podría haber hecho nada relativo al proyecto que nos atañe. Se ha usado software muy variado y de diferentes índoles. A continuación se enumeran:

- **Sistema Operativo:** Windows 7 Professional y Ubuntu 12.10
- **Editor de Código:** Sublime Text 2
- **Procesador de Texto:** Libre Office
- **Editor de Imágenes:** Gimp
- **Modelado UML:** StartUML
- **Control de Versiones:** SVN y TortoiseSVN
- **Editor Modelos 3D :** Google SketchUp
- **Administración BD:** MySQL Workbench

Capítulo 5: Desarrollo del Proyecto

En el desarrollo de software, la primera decisión que se ha de tomar es la metodología de desarrollo que se va a utilizar para llevar a cabo un proyecto. Esta metodología será la que marcará los pasos a seguir durante la construcción del proyecto.

El presente capítulo documenta todo el proceso de creación del software desde la captura de los requisitos hasta las pruebas finales de la aplicación. El capítulo está organizado por apartados donde cada apartado coincide con cada uno de los pasos marcados por la metodología elegida (requisitos del sistema, requisitos del software, análisis, diseño, implementación y pruebas). Cada uno de los apartados estará documentado a través de diagramas, imágenes y textos explicativos.

5.1. Requisitos del Sistemas

La primera etapa de un proyecto es ser capaz de entender las necesidades y problemas que se pretenden resolver con la aplicación. Para ello es necesario conocer también el dominio en el que será aplicada. Esto es lo que se obtiene con los requisitos del sistema. Esta información queda representada por una lista de características y modelo del dominio.

La lista de características enumeran los requisitos candidatos. La misión principal de los requisitos es guiar el desarrollo hacia un sistema correcto en el que se hayan ya identificado y priorizado las funcionalidades deseadas y se hayan detectado los riesgos de manera que no comprometa el proyecto en fases posteriores.

Por otro lado, el modelo de dominio ayuda a comprender el contexto del sistema. Describe los conceptos importantes del contexto como los objetos del dominio y las interrelaciones existentes entre ellos.

5.1.1. Lista de características

A continuación se detalla una lista de características donde se definirán las distintas operaciones que debe permitir la aplicación. En ella se definen tanto las que se van a implementar en esta versión del software como las que se pueden implementar en versiones posteriores. La tabla de característica ha sido subdividida en los siguientes campos:

- **Código.** Es el identificador de la característica. Se especifica como:

  LC-[Grupo]-[Número], donde LC hace referencia a lista de características.

- **Nombre de la característica.**

- **Descripción.** Pequeña explicación de la característica.

- **Prioridad.** Se asigna una prioridad para resaltar las características más importantes para determinar que características se implementarán primero. Las prioridades que se van a utilizar son: *alta, media, baja.*

- **Estado.** Cada característica tiene un estado asociado. Los posibles estados son:

  - *Aceptado.* La característica se desarrollará en esta versión del producto.

  - *Propuesto.* La característica está planteada pero se planificará para una futura versión del producto.

- **Riesgo.** Cada característica puede tener asociado un riesgo que representa la dificultad para conseguir implementarla correctamente. Utilizamos tres niveles de riesgo: *crítico, significativo, rutinario* .

- **Coste.** Representa al coste que conllevaría realizar dicha característica. Los costes que se utilizarán son: *alto, medio, bajo.*

Para que la lista de características quede mejor estructurada y así facilitar su lectura y comprensión se clasifican las operaciones que se requieren en el sistema en cinco grandes grupos:

- **Cliente (A).** En esta categoría irán agrupadas todas las operaciones pertenecientes al cliente de la aplicación.

- **Gestión de usuarios (B).** Las funciones contenidas en esta categoría son todas las correspondientes a la gestión de usuarios.

- **Gestión de dispositivos (C).** Categoría donde se incluyen todas las funciones de administración de dispositivos científicos. Las operaciones son las de creación, edición, asignación de modelos, altas y bajas, etc.

- **Obtención de datos (D).** Son las operaciones relacionadas con la obtención y extracción de datos científicos recabados por los distintos dispositivos de toma de datos. En ella se incluyen todas las conexiones mediante los distintos protocolos con las fuentes de datos.

- **Animación gráfica (E).** Operaciones orientadas a las funciones lúdicas de la aplicación como creación de entornos multimedia de conocimiento del medio.

5.1.1.1. Cliente

LC-A-1	Ver mapa 3D
Descripción	Parte pública de la aplicación donde se mostrará un mapa 3D tipo Google Earth en un entorno web
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Medio

LC-A-2	Mover en mapa 3D
Descripción	Desplazarse en el mapa 3D de forma que se pueda girar el globo terrestre virtual hacia la izquierda, hacia la derecha, hacia arriba y hacia abajo.
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-A-3	Hacer zoom en mapa 3D
Descripción	Acercarse o alejarse a una posición el mapa 3D
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-A-4	Ver mundo submarino en mapa 3D
Descripción	Poder mostrar no sólo la superficie terrestre sino además poder mostrar el fondo marino y los elementos que se encuentren bajo el nivel del mar
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Medio

LC-A-5	Ver dispositivos activos
Descripción	Presentar un menú tipo lista jerárquica donde se muestren todo los dispositivos que se encuentren activos en el momento y por lo tanto se puedan localizar en el mapa 3D
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Medio

LC-A-6	Ver dispositivos en mapa 3D
Descripción	Visualización del dispositivo en su posición actual en el mapa 3D
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Medio

LC-A-7	Localizar en mapa 3D el dispositivo seleccionado en la lista
Descripción	Al seleccionar un dispositivo de la lista de dispositivos que el mapa 3D se sitúe en una vista donde se vea claramente el dispositivo colocado en su ubicación actual.
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Medio

LC-A-8	Ver información del dispositivo
Descripción	Al seleccionar el dispositivo se mostrará la información asociada a este dispositivo en forma de burbuja o similar.
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-A-9	Ver lectura de datos del dispositivo
Descripción	Parte de la información mostrada del dispositivo en la burbuja son los datos capturados por éste mediante tablas de datos o gráficas asociadas a éstas.
Estado	Aprobado
Prioridad	Media
Riesgo	Significativo
Coste	Alto

LC-A-10	Ver vídeo tomado por el dispositivo
Descripción	En la información mostrada en la burbuja se mostrará vídeos capturados por el dispositivo
Estado	Propuesto
Prioridad	Baja
Riesgo	Ordinario
Coste	Alto

LC-A-11	Insertar imágenes reales captadas por dispositivos en el fondo marino del mapa 3D
Descripción	Superponer imágenes reales captadas por el dispositivo al fondo marino virtual
Estado	Propuesto
Prioridad	Baja
Riesgo	Crítico
Coste	Alto

5.1.1.2. Usuarios

LC-B-1	Crear usuario
Descripción	Zona privada de la aplicación donde el administrador crea un nuevo usuario y le añade la información asociada a éste
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-B-2	Editar usuario
Descripción	Zona privada de la aplicación donde el administrador modifica parte de la información de un usuario
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-B-3	Eliminar usuario
Descripción	Zona privada de la aplicación donde el administrador elimina un usuario del sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-B-4	Asignar permisos
Descripción	Zona privada de la aplicación donde el administrador le asigna los permisos de uso a un usuario
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-B-5	Cambiar contraseña
Descripción	Zona privada de la aplicación donde un usuario modifica su antigua contraseña por una nueva
Estado	Aprobado
Prioridad	Media
Riesgo	Ordinario
Coste	Bajo

LC-B-6	Recordar contraseña
Descripción	Zona pública de la aplicación donde un usuario solicita que se le reenvíe su contraseña al correo electrónico asociado a su cuenta de usuario.
Estado	Aprobado
Prioridad	Media
Riesgo	Ordinario
Coste	Bajo

LC-B-7	Login
Descripción	Zona pública de la aplicación donde un usuario registrado introduce sus credenciales de acceso para poder acceder a la parte privada para la que tiene permisos mediante una sesión de usuario
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-B-8	Logout
Descripción	Zona privada de la aplicación donde el usuario solicita el cierre de su sesión de usuario
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-B-9	Controlar acceso a usuarios anónimos
Descripción	Zona privada de la aplicación donde el administrador hace que la parte pública de la aplicación sea o deje de ser visible para usuarios no registrados
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Medio

LC-B-10	Logs de actividades
Descripción	Registro de las actividades realizadas por cada usuario en el sistema
Estado	Propuesto
Prioridad	Baja
Riesgo	Ordinario
Coste	Medio

LC-B-11	Registrarse como usuario
Descripción	Zona pública de la aplicación donde un usuario anónimo introduce sus datos para solicitar crear una cuenta de usuario
Estado	Propuesto
Prioridad	Baja
Riesgo	Ordinario
Coste	Bajo

LC-B-12	Validación de usuario
Descripción	Zona privada de la aplicación donde el administrador valida un registro de usuario y confirma la creación de la cuenta de usuario
Estado	Propuesto
Prioridad	Baja
Riesgo	Ordinario
Coste	Bajo

5.1.1.3. Dispositivos

LC-C-1	Crear tipo de dispositivo
Descripción	Zona privada de la aplicación donde el editor crea un nuevo tipo de dispositivo
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-2	Eliminar tipo de dispositivo
Descripción	Zona privada de la aplicación donde el editor elimina un tipo de dispositivo existente en el sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-3	Editar tipo de dispositivo
Descripción	Zona privada de la aplicación donde el editor modifica la información asociada a un tipo de dispositivo existente en el sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-4	Crear dispositivo
Descripción	Zona privada de la aplicación donde el editor crea un nuevo dispositivo añadiéndole la información asociada a éste
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-5	Dar de alta a dispositivo
Descripción	Zona privada de la aplicación donde el editor activa el dispositivo de modo que sea visible en el mapa 3D
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-6	Asignar modelo a dispositivo
Descripción	Zona privada de la aplicación donde el editor le asigna un modelo CAD al dispositivo
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-7	Dar de baja a dispositivo
Descripción	Zona privada de la aplicación donde el editor desactiva un dispositivo y por lo tanto deja de ser visible en el mapa 3D
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-8	Eliminar dispositivo
Descripción	Zona privada de la aplicación donde el editor elimina un dispositivo del sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-9	Editar dispositivo
Descripción	Zona privada de la aplicación donde el editor modifica la información asociada a un dispositivo
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-10	Introducir modelo
Descripción	Zona privada de la aplicación donde el editor introduce un nuevo modelo CAD en el sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-11	Eliminar modelo
Descripción	Zona privada de la aplicación donde el editor elimina un modelo CAD existente en el sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

LC-C-12	Editar información del modelo
Descripción	Zona privada de la aplicación donde el editor modifica la información asociada a un modelo CAD existente en el sistema
Estado	Aprobado
Prioridad	Alta
Riesgo	Ordinario
Coste	Bajo

5.1.1.4. Obtención de datos

LC-D-1	Extracción de datos de sensores mediante NetCDF
Descripción	Conexión al servidor de datos mediante protocolo NETCDF para la extracción de datos históricos capturados por sensores
Estado	Aprobado
Prioridad	Alta
Riesgo	Crítico
Coste	Alto

LC-D-2	Muestra de metadatos de sensor a partir de fichero SensorML
Descripción	Muestra de datos de sensores a partir de un fichero SensorML.
Estado	Aprobado
Prioridad	Alta
Riesgo	Significativo
Coste	Alto

LC-D-3	Generar gráficas navegables a partir de datos de sensores
Descripción	Creación de gráficas navegables a partir de una serie de datos obtenidos de un sensor.
Estado	Aprobado
Prioridad	Media
Riesgo	Significativo
Coste	Alto

LC-D-4	Conexión a los sensores en tiempo real
Descripción	Comunicación entre la aplicación y un sensor obteniendo la información que está capturando en tiempo real
Estado	Propuesto
Prioridad	Baja
Riesgo	Crítico
Coste	Alto

LC-D-5	Extracción de datos de sensores mediante OGC-SOS-O&M
Descripción	Conexión al servidor de datos mediante protocolo OGC-SOS-O&M para la extracción de datos históricos capturados por sensores
Estado	Propuesto
Prioridad	Baja
Riesgo	Crítico
Coste	Alto

LC-D-6	Crear eventos
Descripción	Zona privada de la aplicación donde el editor crea un nuevo evento en el cual se establece una serie de condiciones que se deben cumplir para que el evento genere un aviso de que las condiciones establecidas se están dando en ese momento
Estado	Propuesto
Prioridad	Baja
Riesgo	Significativo
Coste	Alto

LC-D-7	Editar eventos
Descripción	Zona privada de la aplicación donde el editor modifica las condiciones y la información asociadas a un evento
Estado	Propuesto
Prioridad	Baja
Riesgo	Significativo
Coste	Alto

LC-D-8	Eliminar eventos
Descripción	Zona privada de la aplicación donde el editor elimina un evento existente en el sistema
Estado	Propuesto
Prioridad	Baja
Riesgo	Significativo
Coste	Bajo

LC-D-9	Detectar condiciones de eventos
Descripción	Proceso en segundo plano donde se comprueba que se den una serie de condiciones para lanzar una aviso
Estado	Propuesto
Prioridad	Baja
Riesgo	Significativo
Coste	Alto

LC-D-10	Visualización de eventos
Descripción	Muestra de una alarma informando que las condiciones de un evento se están cumpliendo en ese momento
Estado	Propuesto
Prioridad	Baja
Riesgo	Significativo
Coste	Alto

5.1.1.5. Animación Gráfica

LC-E-1	Integración de contenidos lúdicos basados en Infinity 3D
Descripción	Enlazar aplicación con contenidos lúdicos basados en Infinity 3D
Estado	Propuesto
Prioridad	Bajo
Riesgo	Crítico
Coste	Alto

LC-E-2	Muestra de mundo virtual mediante Unity 3D
Descripción	Enlazar aplicación con mundo virtual usando Unity 3D
Estado	Propuesto
Prioridad	Bajo
Riesgo	Crítico
Coste	Alto

5.1.2. Modelo del dominio

Mediante el modelo del dominio se pretende modelar el contexto en el que se va a introducir el sistema. En él se describe mediante diagramas de clases el funcionamiento actual de las actividades de PLOCAN a las que se quiere dar apoyo tecnológico con el sistema a desarrollar.

5.1.2.1. Instrumentación Oceanográfica y Atmosférica

PLOCAN cuenta con una serie de dispositivos de diferentes tipos, los cuales son capaces de capturar datos de tipo oceanográficos y atmosféricos tanto del mundo submarino como de la superficie marina.

Estos dispositivos, dependiendo del tipo, pueden estar equipados con varios tipos de sensores y cámaras, tanto fotográficas como de vídeo.

Todos estos datos, incluyendo imágenes, son almacenadas en bases de datos administradas por el administrador. Estos datos son de especial interés para personal científico tanto de PLOCAN como del resto del mundo. Los datos pueden ser mostrados en tablas o incluso en gráficas navegables en el tiempo.

La forma de acceso a estos datos por parte de los científicos son vía dos protocolos:

- OPENDAP-NetCDF
- OGC-SOS-O&M

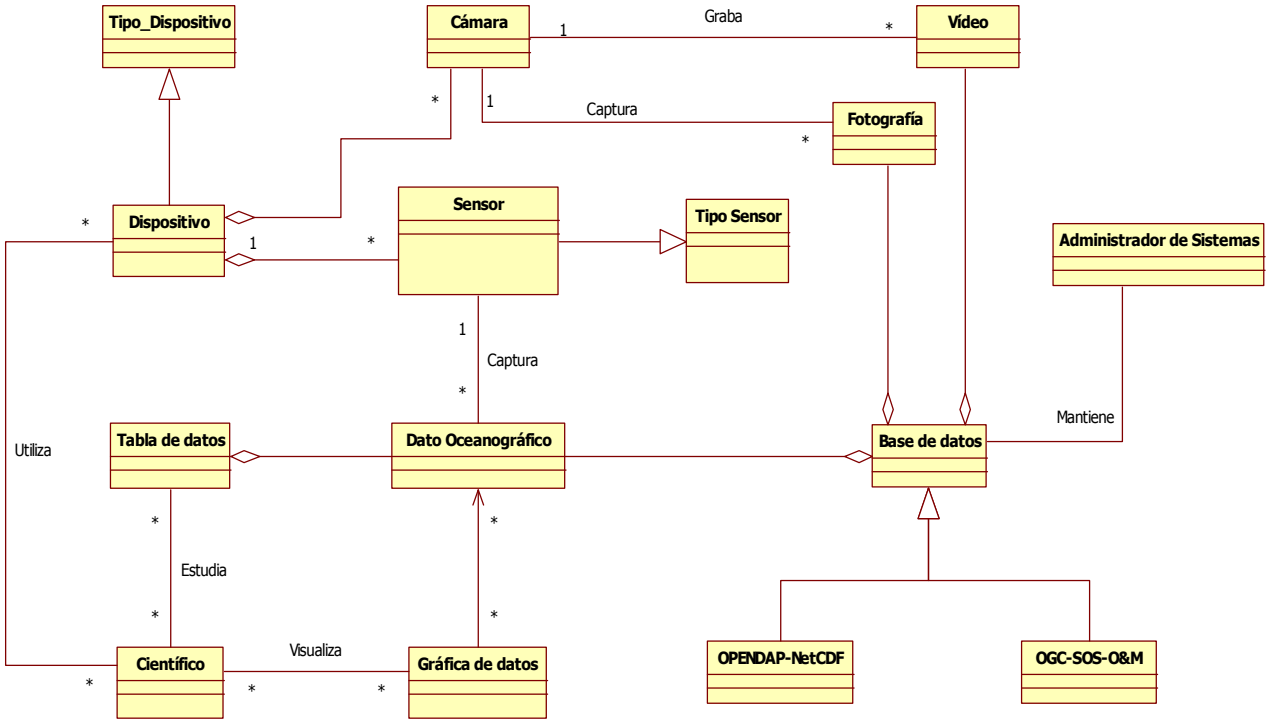


Figura 5.1. Instrumentación Oceanográfica

5.1.2.2. Tipos de dispositivos

Actualmente PLOCAN trabaja con varios tipos de dispositivos de captura de datos.

Los gliders son vehículos submarinos autónomos no tripulados desarrollados para llevar a cabo mediciones de parámetros físico-químicos de la columna de agua tanto en zonas costeras como oceánicas hasta profundidades de 1000 m.

Tanto los vehículos submarinos ROVs (remote operated vehicle) como los gliders son dirigidos por un operador desde tierra mediante control remoto.

Aparte cuentan con dos tipos de boyas: las boyas de deriva, que son boyas libres que se mueven a través del impulso del viento, las olas y las corrientes marinas y van tomando medidas por cualquier punto por el que van pasando; y las boyas ancladas, que como su propio nombre indica se encuentran ancladas al fondo marino de forma de que toma medidas en un mismo punto del océano de forma continua y a diferentes profundidades gracias a los sensores instalados en el anclaje de la boya.

Y por último cuentan, aunque no es un dispositivo como tal, con las tortugas marinas. A las tortugas se les equipa con unos transmisores de forma de que se pueden estudiar sus movimientos migratorios y su relación con los parámetros ambientales marinos.

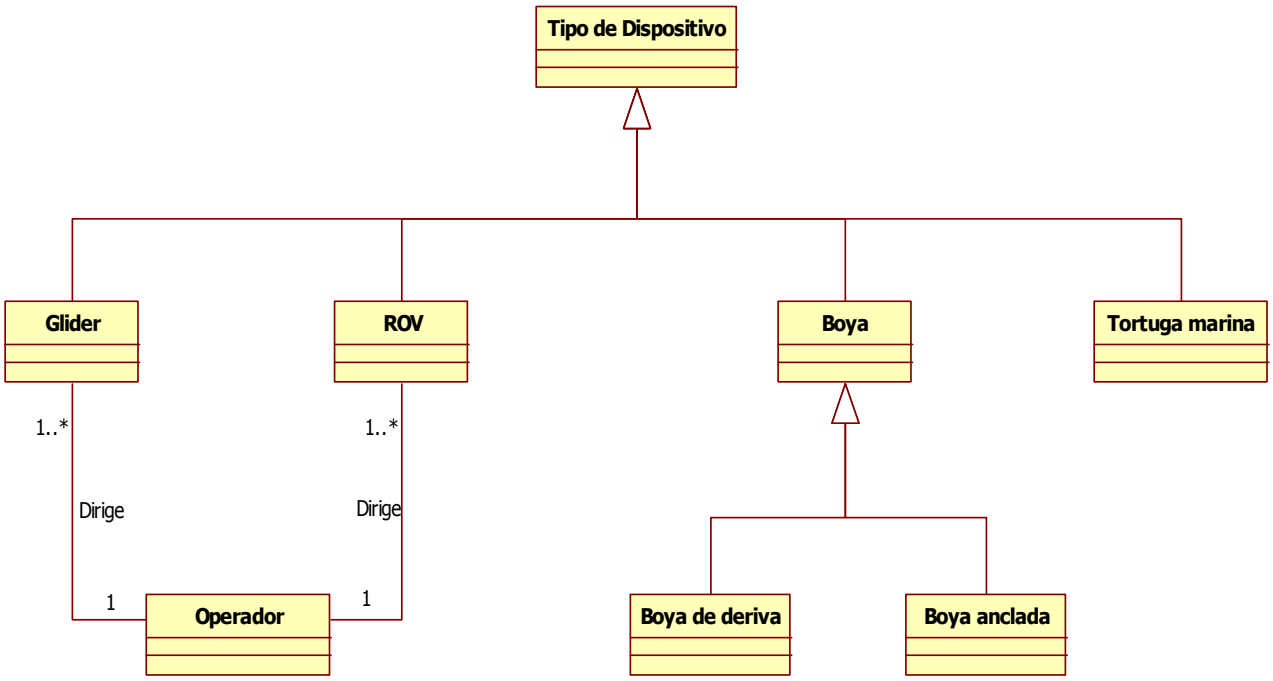


Figura 5.2. Plataformas de observación

5.1.2.3. Equipamiento de dispositivos

Como se ha explicado anteriormente, los dispositivos pueden ser de diferentes tipos, y dependiendo de sus funciones pueden estar equipados con distintas herramientas. En el siguiente gráfico se muestra la posible instrumentación de un dispositivo.

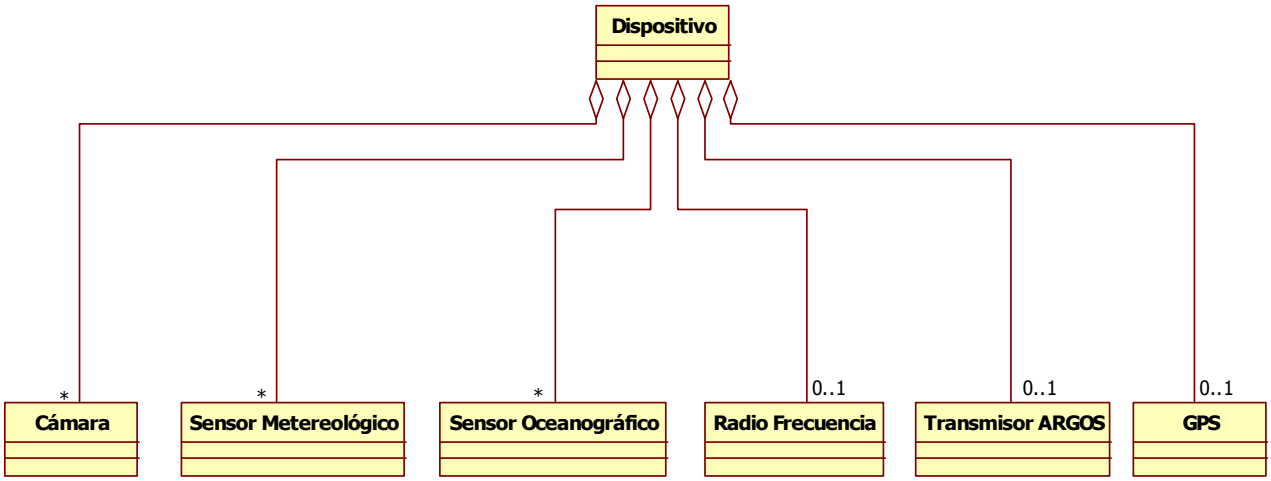


Figura 5.3. Equipamiento de dispositivos

5.2. Requisitos del Software

La especificación de requisitos del software es una descripción intensiva del comportamiento del sistema. A partir de la lista de características del apartado anterior, se obtiene una representación en forma de casos de uso.

El modelo de casos de uso define la interacción entre los distintos usuarios, o actores a partir de ahora, y el sistema. Los casos de uso son pequeñas unidades de la funcionalidad completa del sistema. Con los casos de uso se representarán los requisitos funcionales del sistema.

Cada caso de uso es definido en detalle con la especificación del caso de uso. Esta especificación es una secuencia precisa de acciones necesarias para realizar un caso de uso.

Por lo tanto, a lo largo de este apartado se habrá que definir los actores del sistema, el modelo de casos de uso para cada paquete o módulo del sistema y la especificación de cada uno de ellos.

5.2.1. Actores del Sistema

El siguiente diagrama muestra la jerarquía de usuarios o actores que interactuarán con el sistema. Existe un usuario genérico del cual heredan todos los usuarios representando la funcionalidad común a todos ellos.

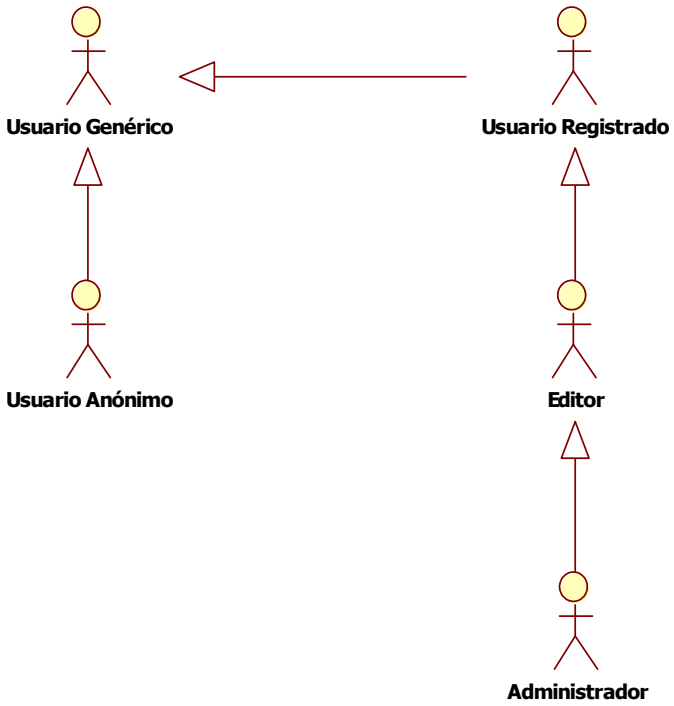


Figura 5.4. Actores del sistema

- **Usuario anónimo.** Será el usuario no registrado en el sistema. Este usuario tendrá acceso únicamente a la parte del cliente, y siempre que el administrador no cierre el acceso para este tipo de usuario.
- **Usuario registrado.** Son los usuarios que están registrados en el sistema y que inician su sesión para hacer uso de la aplicación. Estos usuarios tienen acceso a la misma parte de la aplicación que el usuario anónimo, con la diferencia de que el usuario registrado sí puede acceder al cliente cuando el administrador declara privada a la aplicación.
- **Editor.** Este tipo de usuario es el encargado de administrar la información científica de la aplicación. Tiene acceso a la interfaz de administración donde se mantienen dispositivos, modelos y cualquier otro artefacto científico integrado en la aplicación. Además hereda la funcionalidad disponible para el usuario registrado.
- **Administrador.** El perfil del súper usuario. Es el usuario que tiene acceso total a todas las partes de la aplicación. Tiene acceso al sistema de administración donde puede acceder a todas las secciones de administración. Además de las funciones del editor, también es el encargado de la gestión de usuarios y de sus permisos, y de limitar el acceso a la aplicación haciéndola privada en los momentos puntuales que se deseen.

5.2.2. Modelo de Casos de Uso

Los casos de uso han sido clasificados en subsistemas lógicos de forma que los casos de uso quedan organizados por la naturaleza de su funcionalidad. Se han divido en tres paquetes siguiendo la lógica llevada a cabo al organizar la lista de características.

Un primer subsistema lógico es el del "Cliente" donde se agrupan todos los casos de uso correspondientes al funcionamiento del cliente de la aplicación. Este paquete tiene dependencia con el resto de paquetes ya que hace uso de todos los demás para su correcto funcionamiento.

El paquete "Usuarios" engloba los casos de uso relacionados con las cuentas de usuarios. En él se describen las tareas de administración sobre las cuentas de usuario y la creación y cierres de sesiones de usuarios.

El paquete "Dispositivos" agrupa las funcionalidades sobre dispositivos, tipos de dispositivos y modelos. Con él se satisfacen las necesidades sobre la administración de dichos artefactos.

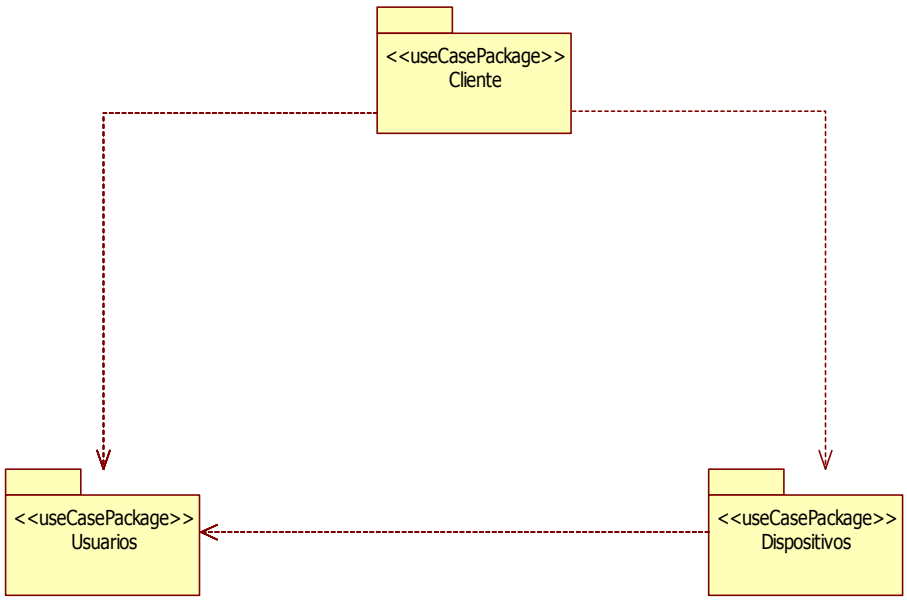


Figura 5.5. Paquetes de casos de uso.

A continuación se definen los casos de uso que se van a implementar. Los casos de uso estarán divididos en las categorías que se han definido anteriormente. En cada categoría se mostrarán los casos de uso mediante diagramas de casos de uso y posteriormente se especificará cada caso de uso mediante una tabla descriptiva.

5.2.2.1. Cliente

Este subsistema lógico engloba los casos de uso del cliente de la aplicación. Todos los usuarios del sistema tienen acceso a esta sección del software.

El cliente representa la zona, a priori, "pública" de la aplicación. En ella estará contenido el mapa 3D, los menús, y la visualización de todos los dispositivos y los datos capturados por sus sensores.

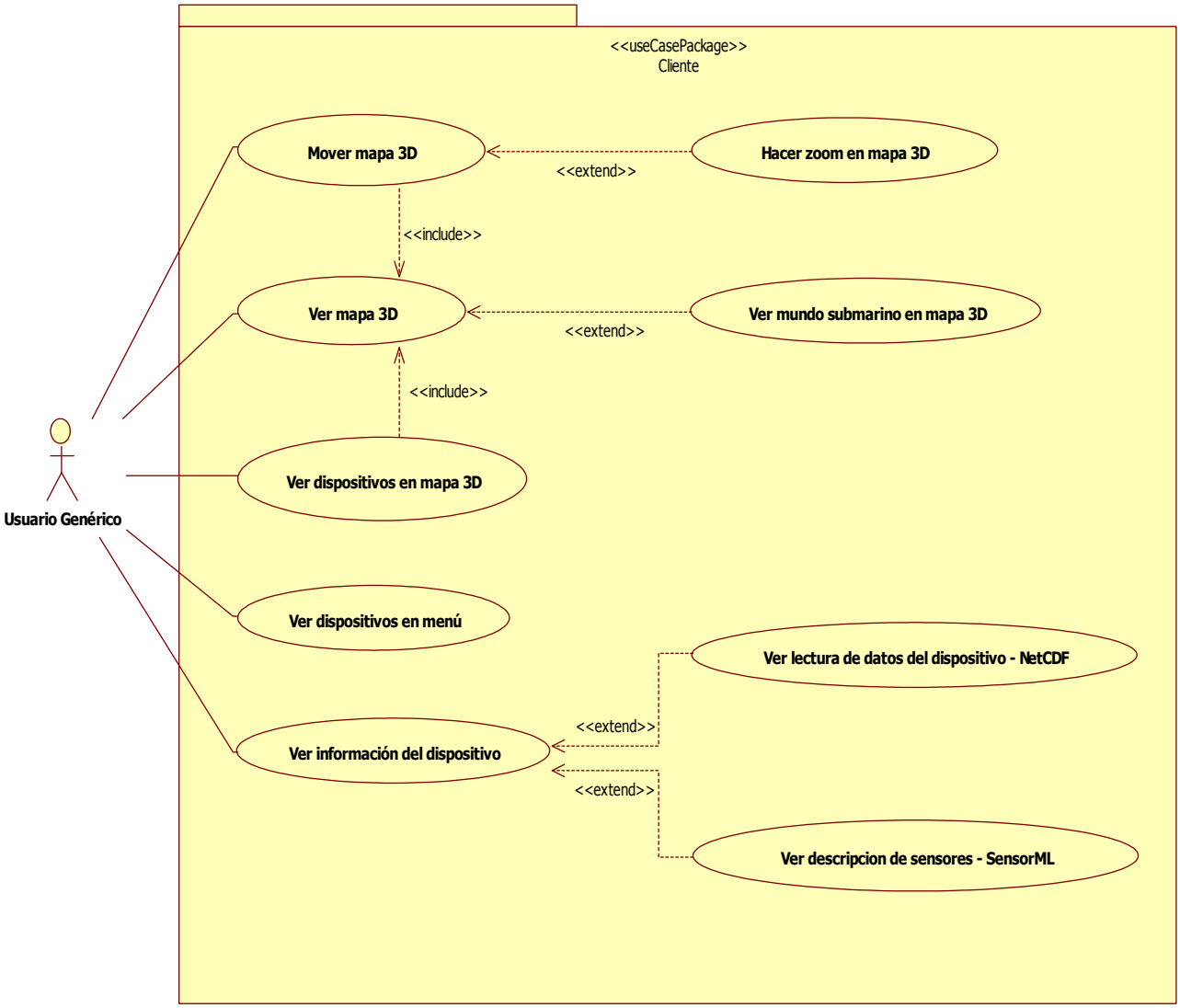


Figura 5.6. Casos de uso cliente.

5.2.2.2. Usuarios

El subsistema lógico Usuarios engloba los casos de uso relacionados con la gestión de usuarios. El usuario con perfil de administrador tendrá las tareas de crear, modificar y eliminar usuarios, además de asignarle los permisos a éstos. También el administrador tendrá la posibilidad de cerrar el acceso a la aplicación a los usuarios anónimos, con lo que cuando esta opción esté activada, sólo los usuarios registrados podrán hacer uso del cliente de la aplicación.

En este subsistema también se incluyen el uso de estas cuentas de usuarios dando la posibilidad a los usuarios registrados de iniciar una sesión de usuario, finalizarla y cambiar su contraseña.

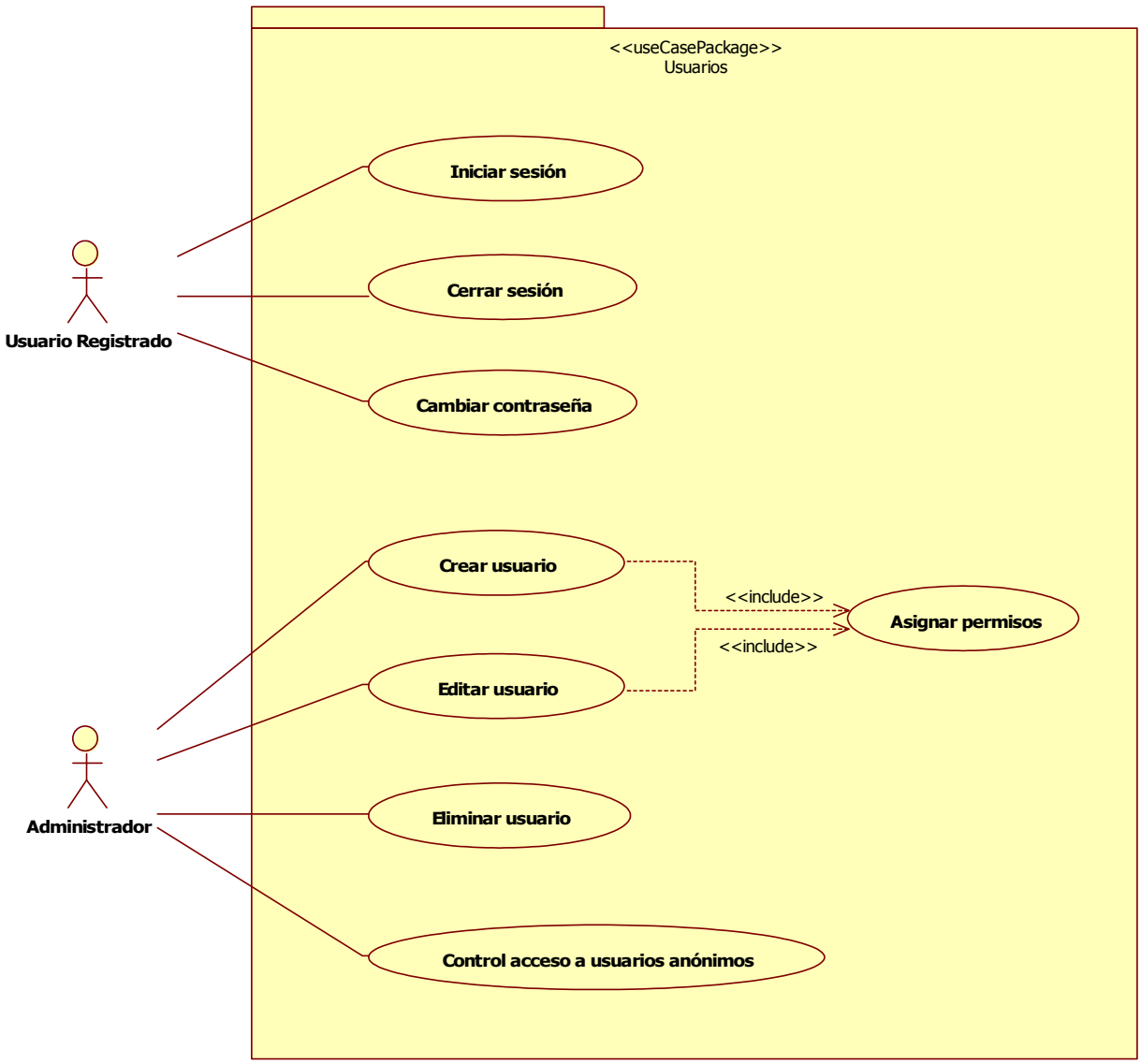


Figura 5.7. Casos de uso usuario

5.2.2.3. Dispositivos

El subsistema lógico “Dispositivos” engloba todos los casos de uso relacionado con la gestión y administración de los dispositivos y todos los demás artefactos asociados a éstos.

El actor que está directamente relacionado con este subsistema la figura del editor. El editor es el encargado de crear un dispositivo, editarlo, eliminarlo, añadirle sensores, asignarle modelo 3D, asignarle un tipo y darle de alta o de baja en la aplicación.

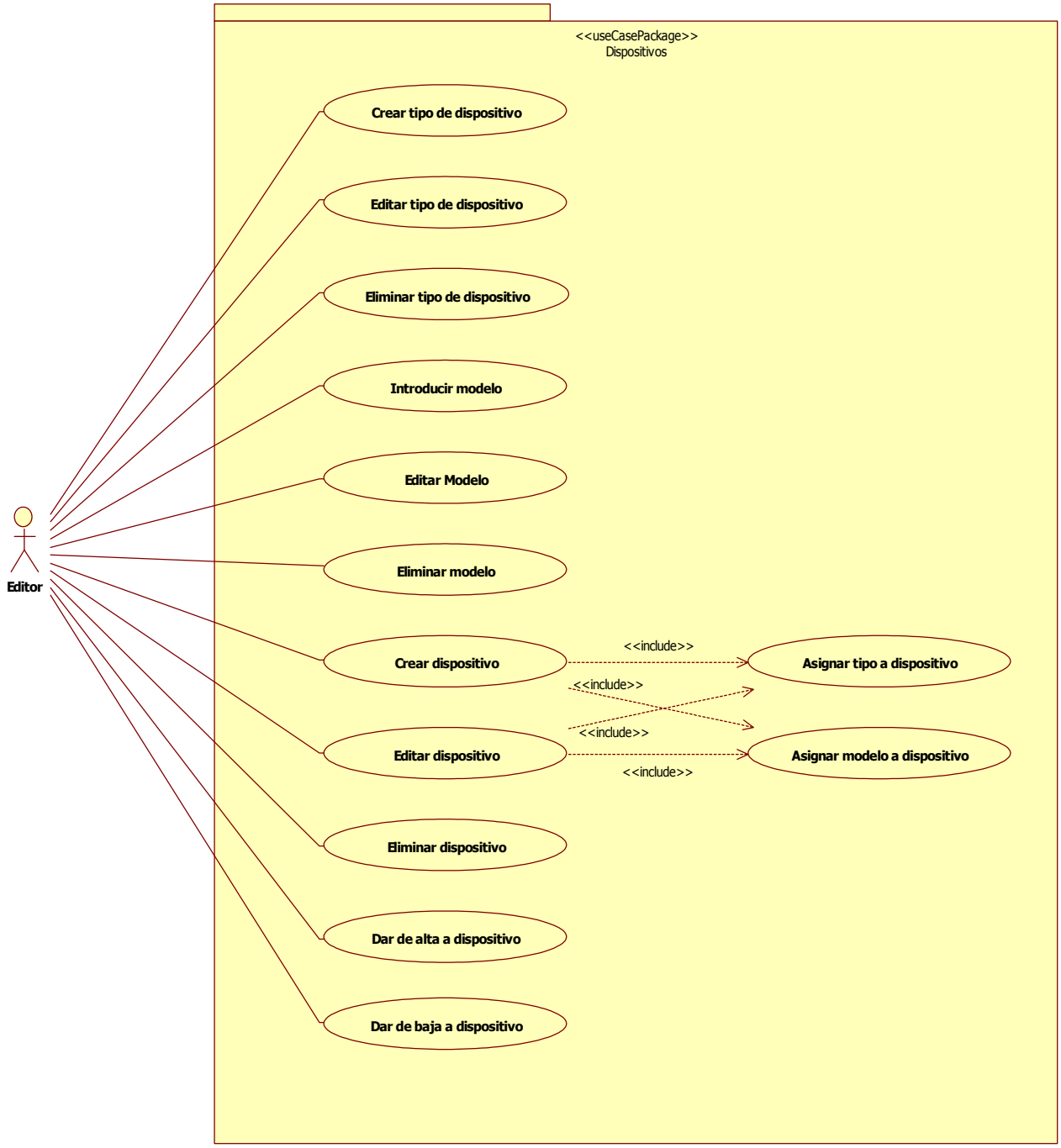


Figura 5.8. Casos de uso dispositivos.

5.2.3. Especificación de Casos de Uso

Con los diagramas de casos de uso se puede obtener una idea global de las operaciones que cada usuario puede realizar pero no nos da una información completa de en qué condiciones se pueden ejecutar o el flujo de acciones que se tienen que llevar a cabo.

Para solventar ésto, se hace uso de las tablas de especificación de casos de uso donde se recoge la siguiente información:

- **Código.** Es el identificador del caso de uso. Se especifica como:

  CU-[Grupo]-[Número], donde CU hace referencia a caso de uso.

- **Nombre del caso de uso.**
- **Descripción.** Pequeña explicación del caso de uso.
- **Subsistema.** Subsistema al que pertenece.
- **Actor principal.** Tipo de u*suario que ejecuta la acción. Es uno de los actores del sistema.*
- **Precondiciones.** Condiciones que deben cumplirse para poder iniciar la realización.
- **Flujo de ejecución.** Flujo de acciones que deben ejecutarse durante la realización.
- **Camino alternativo.** Flujo de acciones que se llevan a cabo si no se cumple la precondición.
- **Postcondiciones.** Condiciones cumplidas tras finalizar la realización.

5.2.3.1. Cliente

CU-A-1	Ver mapa 3D
Descripción	Acción mediante la cual cualquier usuario obtiene el mapa 3D de la tierra en la vista por defecto
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	
Flujo de ejecución	1. El usuario accede a la dirección de la aplicación. 2. Se muestra en la ventana del navegador el mapa 3D
Camino alternativo	
Postcondiciones	Mapa 3D activo

CU-A-2	Ver mundo submarino en mapa 3D
Descripción	Acción mediante la cual cualquier usuario obtiene la vista bajo el nivel del mar del mapa 3D de la tierra en la posición actual
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Mapa 3D activo
Flujo de ejecución	**1.** Avanzar hasta estar bajo el nivel del mar **2.** Se muestra la imagen de fondo marino de las coordenadas seleccionadas
Camino alternativo	2.1. Si no existe mar en la posición seleccionada no se hace nada
Postcondiciones	Mapa 3D activo

CU-A-3	Mover mapa 3D
Descripción	Acción mediante la cual cualquier usuario obtiene la vista del mapa 3D de la posición seleccionada
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Mapa 3D activo
Flujo de ejecución	**1.** Hacer girar el mapa 3D hasta la posición deseada **2.** Se muestra la imagen del mapa de la localización seleccionada
Camino alternativo	
Postcondiciones	Mapa 3D activo

CU-A-4	Hacer zoom en mapa 3D
Descripción	Acción mediante la cual cualquier usuario obtiene la vista del mapa 3D de la posición actual pero aumentada o disminuida
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Mapa 3D activo
Flujo de ejecución	**1.** Hacer avanzar o retroceder el zoom hasta la vista deseada **2.** Se muestra la imagen del mapa de la localización seleccionada con un nuevo zoom
Camino alternativo	2.1. Si el zoom está al límite no se hace nada.
Postcondiciones	Mapa 3D activo

CU-A-5	Ver dispositivos en el mapa 3D
Descripción	Acción mediante la cual cualquier usuario obtiene la vista del mapa 3D de la posición seleccionada donde se encuentra el dispositivo seleccionado
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Mapa 3D activo
Flujo de ejecución	**1.** Hacer girar el mapa 3D hasta la posición deseada **2.** Se hace doble click sobre el icono del dispositivo en el mapa **3.** Se muestra la imagen del dispositivo situado en el mapa 3D
Camino alternativo	
Postcondiciones	Mapa 3D activo

CU-A-6	Ver dispositivos en el menú
Descripción	Existe un menú donde el usuario puede ir accediendo a la lista jerárquica de dispositivos navegando por dicho menú
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	
Flujo de ejecución	**1.** Seleccionar en el menú la familia del dispositivo **2.** Seleccionar el dispositivo **3.** Mostrar en el mapa 3D el dispositivo seleccionado
Camino alternativo	
Postcondiciones	

CU-A-7	Ver información del dispositivo
Descripción	Acción mediante la cual cualquier usuario obtiene información sobre el dispositivo seleccionado
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Mapa 3D activo
Flujo de ejecución	**1.** Hacer click sobre el icono del dispositivo en el mapa **2.** Se muestra la información del dispositivo en ventana modal sobre el mapa 3D
Camino alternativo	
Postcondiciones	Ventana modal activa

CU-A-8	Ver lectura de datos del dispositivo
Descripción	Acción con la que cualquier usuario obtiene la lectura de datos de los sensores de un dispositivo. Esta información está almacenada mediante NetCDF.
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Ventana modal activa
Flujo de ejecución	1. Acceder a la sección de lectura de datos 2. Se muestra un formulario donde el usuario selecciona las magnitudes a mostrar. 3. Se conecta a la base de datos NetCDF del dispositivo y se obtiene los datos de sus sensores 4. La información se formatea y se muestra mediante gráficas
Camino alternativo	
Postcondiciones	Ventana modal activa y gráfica navegable

CU-A-9	Obtención de descripción de sensores - SensorML
Descripción	Acción con la que cualquier usuario obtiene la descripción de los sensores de un dispositivo. Esta información está almacenada en formato SensorML
Subsistema	Cliente
Actor principal	Usuario genérico
Precondiciones	Burbuja de información activa
Flujo de ejecución	1. Acceder a la sección de lectura de datos 2. Se muestra un listado de sensores del dipositivo. 3. Se selecciona el sensor del dispositivo 4. Se conecta al servidor de ficheros SensorML y obtiene la información del sensor seleccionado 5. La información se formatea y se muestra
Camino alternativo	
Postcondiciones	Ventana modal activa

5.2.3.2. Usuarios

CU-B-1	Crear usuario
Descripción	Acción mediante la cual el administrador crea una nueva cuenta de usuario
Subsistema	Usuarios
Actor principal	Administrador
Precondiciones	Sesión de administrador activa
Flujo de ejecución	**1.** Acceder a crear usuario **2.** Introducir los datos de usuario **3.** Asignar permisos **4.** Se da de alta al usuario en el sistema
Camino alternativo	4.1. Si el usuario ya existe se cancela la operación y se muestra un mensaje al usuario volviendo al paso 2.
Postcondiciones	

CU-B-2	Editar usuario
Descripción	Acción mediante la cual el administrador del sistema edita parte o toda la información de una cuenta de usuario
Subsistema	Usuarios
Actor principal	Administrador
Precondiciones	Sesión de administrador activa
Flujo de ejecución	**1.** Acceder a editar usuario **2.** Modificar los datos de usuario y/o permisos **3.** Se actualiza la información del usuario en el sistema
Camino alternativo	
Postcondiciones	

CU-B-3	Eliminar usuario
Descripción	Acción mediante la cual el administrador del sistema elimina una cuenta de usuario
Subsistema	Usuarios
Actor principal	Administrador
Precondiciones	Sesión de administrador activa
Flujo de ejecución	**1.** Seleccionar el usuario que se quiere eliminar **2.** Ordenar eliminar **3.** Se elimina el usuario en el sistema
Camino alternativo	
Postcondiciones	

CU-B-4	Asignar permisos a usuarios
Descripción	Acción mediante la cual a un usuario registrado se le asigna permisos sobre dispositivos individuales.
Subsistema	Usuarios
Actor principal	Administrador
Precondiciones	Sesión de administrador activa
Flujo de ejecución	**1.** Acceder a editar usuario **2.** Modificar los permisos del usuario **3.** Se actualizan los permisos del usuario
Camino alternativo	
Postcondiciones	

CU-B-5	Controlar acceso a usuarios anónimos
Descripción	Acción mediante la cual el administrador abre o cierra el acceso a todo usuario no registrado y con sesión iniciada al cliente de la aplicación
Subsistema	Usuarios
Actor principal	Administrador
Precondiciones	Sesión de administrador activa
Flujo de ejecución	**1.** Acceder a cerrar acceso a usuarios anónimos **2.** Se cierra el acceso al cliente a los usuarios anónimos
Camino alternativo	
Postcondiciones	

CU-B-6	Iniciar sesión
Descripción	Acción mediante la cual un usuario registrado inicia su sesión de usuario
Subsistema	Usuarios
Actor principal	Usuario registrado
Precondiciones	Sesión de usuario no iniciada
Flujo de ejecución	**1.** Introducir nombre de usuario y contraseña **2.** El sistema comprueba que los credenciales son correctos **3.** Se accede al sistema y se crea la sesión
Camino alternativo	1.1. Si el usuario no rellena todos los campo se devuelve un error para informarle 2.1. Si los credenciales no son correctos se redirige a la pantalla de login informando del error volviendo al paso 1
Postcondiciones	Sesión de usuario iniciada

CU-B-7	Cerrar sesión
Descripción	Acción mediante la cual un usuario registrado finaliza su sesión de usuario
Subsistema	Usuarios
Actor principal	Usuario registrado
Precondiciones	Sesión de usuario iniciada
Flujo de ejecución	**1.** Acceder a cerrar sesión **2.** Se finaliza la sesión de usuario
Camino alternativo	
Postcondiciones	Sesión de usuario no iniciada

CU-B-8	Cambiar contraseña
Descripción	Acción mediante la cual un usuario registrado cambia su contraseña de acceso
Subsistema	Usuarios
Actor principal	Usuario registrado
Precondiciones	Sesión de usuario iniciada
Flujo de ejecución	**1.** Acceder a cambiar contraseña **2.** Introducir la contraseña antigua y la nueva repetida **3.** Se cambia la contraseña del usuario
Camino alternativo	2.1. Si la contraseña antigua es incorrecta o las dos repeticiones de la nueva no coinciden se muestra un error al usuario
Postcondiciones	

5.2.3.3. Dispositivos

CU-C-1	Crear dispositivo
Descripción	Acción mediante la cual el editor del sistema crea una nuevo dispositivo, le asigna su tipo, su modelo 3D y le añade los sensores por los que está compuesto
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Acceder a crear dispositivos **2.** Introducir los datos del dispositivo **3.** Asignar tipo **4.** Asignar modelo 3D **5.** Introducir los datos de cada sensor perteneciente al dispositivo **6.** Se da de alta al dispositivo en el sistema
Camino alternativo	2.1. Si no se introducen todos los datos requeridos se muestra un error al usuario
Postcondiciones	

CU-C-2	Editar dispositivo
Descripción	Acción mediante la cual el editor modifica parte o toda la información de un dispositivo
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Acceder a editar dispositivo **2.** Modificar los datos del dispositivo, tipo y/o modelo 3D **3.** Se actualiza la información del dispositivo en el sistema
Camino alternativo	
Postcondiciones	

CU-C-3	Eliminar dispositivo
Descripción	Acción mediante la cual el editor elimina un dispositivo del sistema
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Seleccionar el dispositivo que se quiere eliminar **2.** Ordenar eliminar **3.** Se elimina el dispositivo del sistema
Camino alternativo	
Postcondiciones	

CU-C-4	Crear tipo de dispositivo
Descripción	Acción mediante la cual el editor crea un tipo de dispositivo que pasa a identificarse como familia de dispositivo
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Acceder a crear tipo de dispositivo **2.** Introducir los datos del dispositivo **3.** Se da de alta al tipo de dispositivo en el sistema
Camino alternativo	2.1. Si no se introducen todos los datos requeridos se muestra un error al usuario
Postcondiciones	

CU-C-5	Editar tipo de dispositivo
Descripción	Acción mediante la cual el editor modifica parte o toda la información de un dispositivo
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Acceder a editar tipo dispositivo **2.** Modificar los datos del tipo de dispositivo **3.** Se actualiza la información del tipo de dispositivo en el sistema
Camino alternativo	
Postcondiciones	

CU-C-6	Eliminar tipo de dispositivo
Descripción	Acción mediante la cual el editor elimina un tipo de dispositivo del sistema
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Seleccionar el tipo de dispositivo que se quiere eliminar **2.** Ordenar eliminar **3.** Se elimina el tipo de dispositivo del sistema
Camino alternativo	3.1. Mientras exista algún dispositivo con el tipo de dispositivo que se quiere eliminar, no se podrá eliminar el tipo avisando al usuario de ello.
Postcondiciones	

CU-C-7	Introducir modelo
Descripción	Acción mediante la cual un editor añade al sistema un modelo 3D
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Acceder a insertar modelo **2.** Introducir los datos del modelo **3.** Se selecciona el fichero .collada **4.** Se da de alta al modelo en el sistema
Camino alternativo	2.1. Si no se introducen todos los datos requeridos se muestra un error al usuario
Postcondiciones	

CU-C-8	Editar modelo
Descripción	Acción mediante la cual el editor modifica parte o toda la información de un modelo 3D
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Acceder a editar modelo **2.** Modificar los datos del modelo **3.** Se actualiza la información del modelo en el sistema
Camino alternativo	
Postcondiciones	

CU-C-9	Eliminar modelo
Descripción	Acción mediante la cual el editor elimina un tipo de dispositivo del sistema
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa
Flujo de ejecución	**1.** Seleccionar el modelo que se quiere eliminar **2.** Ordenar eliminar **3.** Se elimina el modelo del sistema
Camino alternativo	3.1. Mientras exista algún dispositivo con el modelo que se quiere eliminar, no se podrá eliminar el modelo avisando al usuario de ello.
Postcondiciones	

CU-C-10	Dar de alta/ baja a dispositivo
Descripción	Acción mediante la cual el editor decide que un dispositivo existente sea visible en la aplicación
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa, dispositivo no visible
Flujo de ejecución	**1.** Acceder a dar de alta a dispositivo **2.** El dispositivo es visible en la aplicación
Camino alternativo	
Postcondiciones	Dispositivo visible

CU-C-11	Dar de baja a dispositivo
Descripción	Acción mediante la cual un usuario registrado cambia su contraseña de acceso
Subsistema	Dispositivos
Actor principal	Editor
Precondiciones	Sesión de editor activa, dispositivo visible
Flujo de ejecución	1. Acceder a dar de baja a dispositivo 2. El dispositivo no es visible en la aplicación
Camino alternativo	
Postcondiciones	Dispositivo no visible

5.3. Análisis

El objetivo principal del presente apartado es comprender el problema y comenzar a definir un modelo de lo que se pretende construir independientemente de la tecnología y lenguaje de programación que se vaya a utilizar.

Con el análisis se pasará del modelo de casos de uso al modelo de análisis a través de las trazas y seguidamente se documentará con diagramas de colaboración, en el que queda definido como interaccionan las distintas entidades.

El modelo de análisis ayuda a refinar los requisitos y reflexionar sobre los aspectos internos del sistema. En esta etapa se identifican los paquetes de análisis y las clases que los componen. Según se va avanzando en el modelo de análisis estos paquetes se van refinando y manteniendo.

A diferencia con el modelo de casos de uso, que servía para capturar la funcionalidad del sistema, el modelo de análisis sirve para dar forma a la arquitectura para soportar dichas funcionalidades y a su vez ayuda a identificar las clases de análisis que posteriormente se convertirán en clases de diseño.

Por lo tanto, el apartado estará formado por el subapartado de arquitectura de paquetes de análisis, donde estarán definidos los paquetes y las relaciones entre ellos, y el subapartado de realización de casos de uso donde se expresará mediante diagramas de colaboración y diagramas de clases las relaciones y acciones necesarias para la realización de cada caso de uso.

5.3.1. Arquitectura de paquetes

La siguiente figura representa la organización lógica de los paquetes de análisis que forman el sistema. En ella se muestran los distintos paquetes que componen el modelo de análisis y la relación entre ellos.

El paquete Cliente contiene todas las operaciones relacionadas con el cliente de la aplicación y depende del resto de paquetes. Además contiene las funciones de interrelación y comunicación, mediante los distintos protocolos del cliente con los dispositivos para obtener los datos o lecturas realizados por éstos y ser mostrados.

El paquete Usuarios engloba todas las acciones de administración y gestión de cuentas de usuario. El paquete Dispositivos es el encargado de la administración y gestión de dispositivos, sensores, modelos y tipos de dispositivos.

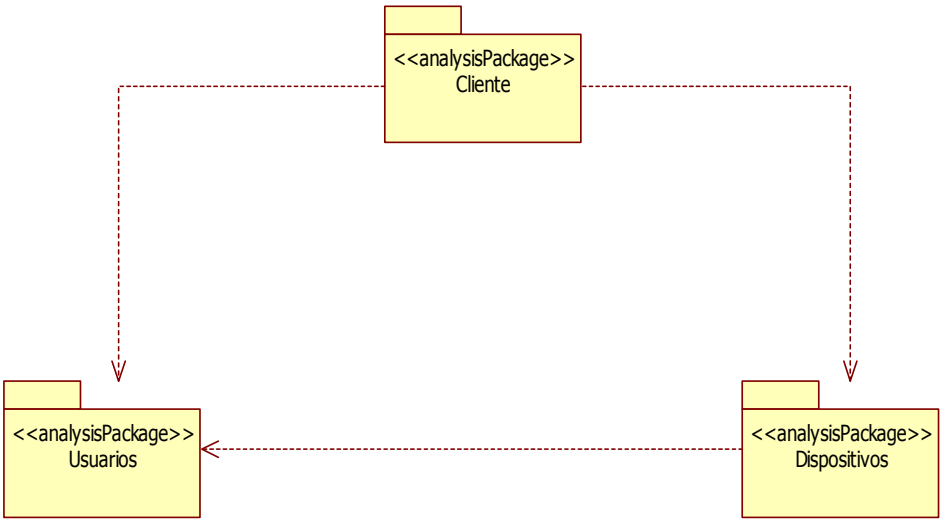


Figura 5.9. Paquetes de análisis.

5.3.2. Realización de Casos Uso

A continuación se hará el análisis de cada caso de uso. Con ello se obtiene cada realización "caso de uso - análisis" con lo que a partir de cada caso de uso se genera su realización en el modelo de análisis.

5.3.2.1. Cliente

La siguiente imagen representa el paquete de análisis Cliente que contiene todas las realizaciones de casos de uso del cliente de la aplicación.

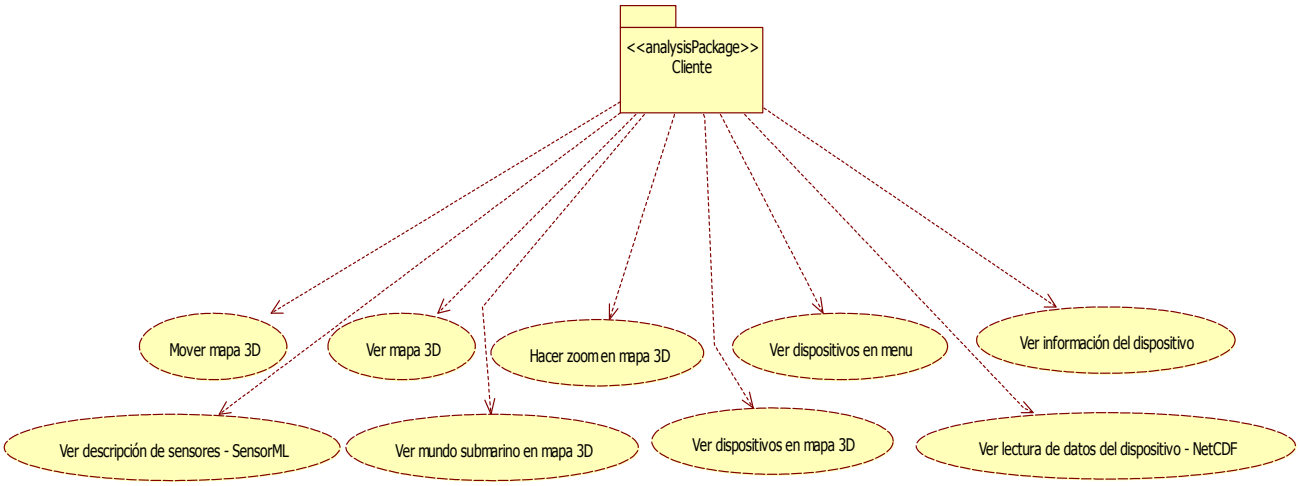


Figura 5.10. Realización casos de uso cliente

A continuación se define el diagrama de clases del paquete de análisis Cliente, donde se muestra la relación entre los actores del sistema y la interconexión entre cada una de las clases.

Las clases las podemos englobar en tres grandes grupos: clases de control, interfaces y entidades.

- Las clases de control se encargan de la gestión y manipulación de los datos.
- Las interfaces proporcionan las funciones de relación entre un usuario y el sistema.
- Las entidades contienen la información.

El paquete cliente hace uso de las siguientes clases agrupadas en base a la definición anterior.

Control

- **Controlador Mapa 3D.** Gestiona todas las acciones que se realizan sobre el mapa 3D de la aplicación.
- **Controlador Dispositivos.** Gestiona la información y realiza las acciones sobre dispositivos, sensores, modelos o tipos.
- **Controlador Menú.** Lleva a cabo las acciones sobre el menú desplegable.

Interfaz

- **IU Mapa 3D.** Representa al mapa 3D en sí mismo y permite al usuario interactuar con él.
- **IU Dispositivo.** Permite al usuario obtener información y datos de los distintos dispositivos que contiene el sistema.
- **IU Menú.** Representa al menú desplegable.

Entidad

- **Dispositivo.** Contiene los dispositivos presentes en el sistema.

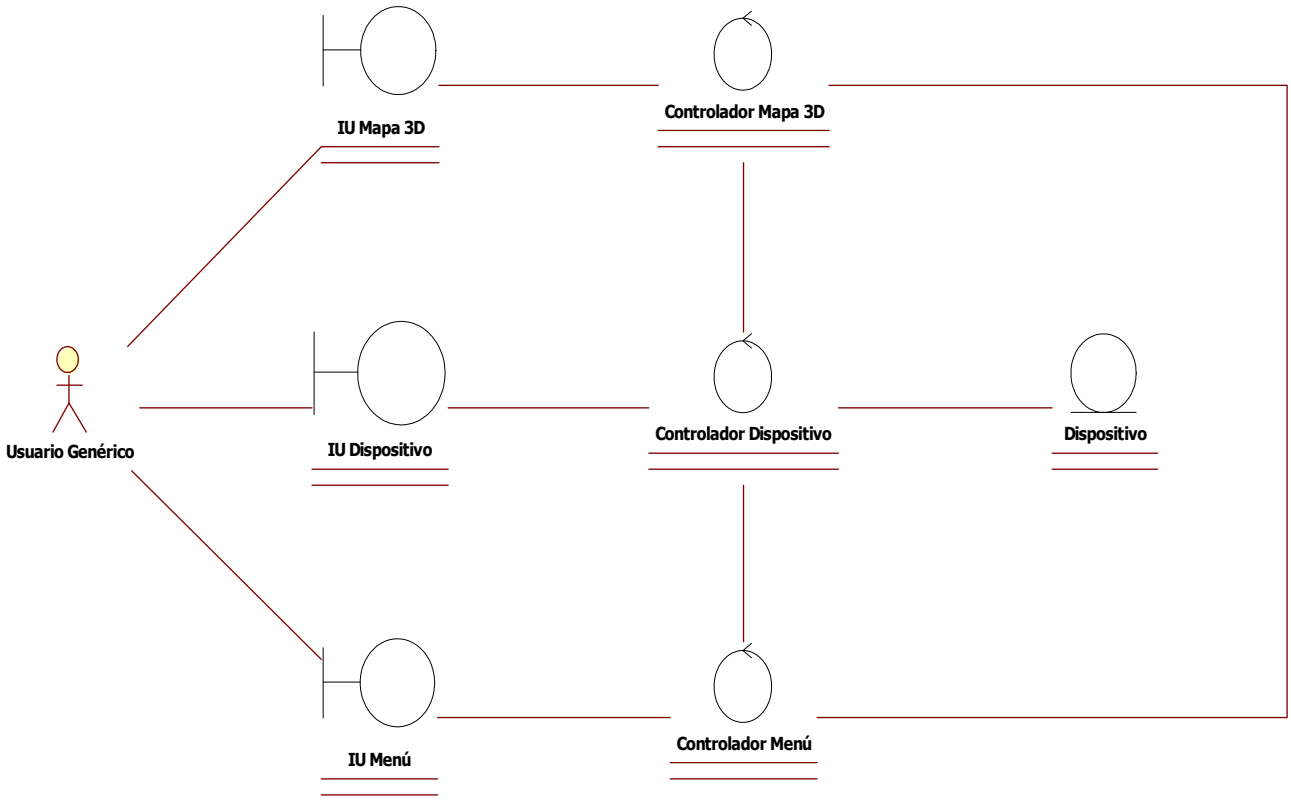


Figura 5.11. Diagrama colaboración cliente.

Ver mapa 3D

Como se ha comentado anteriormente, cada caso de uso tendrá su representación en el modelo de análisis a través de una relación de traza.

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Mapa 3D	IU Mapa 3D	

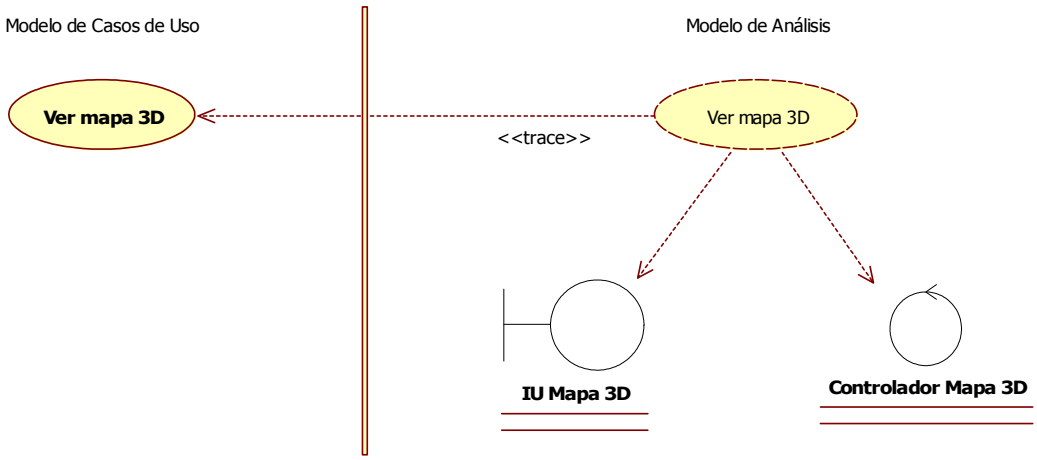


Figura 5.12. Traza (Caso de uso – Análisis) ver mapa 3D.

Este caso de uso se inicia al acceder a la URL del cliente de la aplicación. El controlador recoge esa petición y genera la interfaz con el mapa en el que se muestra al usuario. Cualquier usuario tiene permisos para esta acción y por lo tanto se representa mediante el usuario genérico.

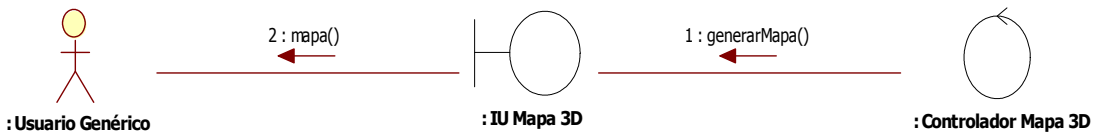


Figura 5.13. Diagrama colaboración Ver Mapa 3D.

Ver mundo submarino en mapa 3D

Este caso de uso es una particularidad del anterior, y lo que hace es permitir a un usuario situarse en el mapa 3D en un nivel negativo o bajo el agua.

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Mapa 3D	IU Mapa 3D	

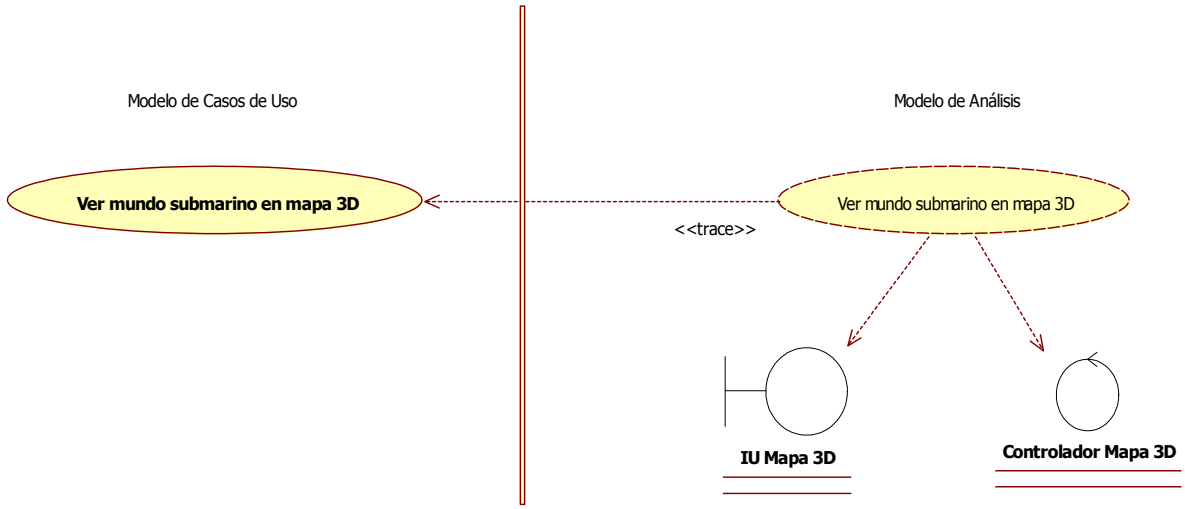


Figura 5.14. Traza (Caso de uso – Análisis) ver mundo submarino en mapa 3D.

Una vez un usuario genérico tiene acceso al mapa 3D, usa la interfaz de usuario para colocarse en un punto submarino. El controlador recibe la nueva posición y genera la vista del mapa 3D en ese punto mostrando el fondo marino correspondiente.

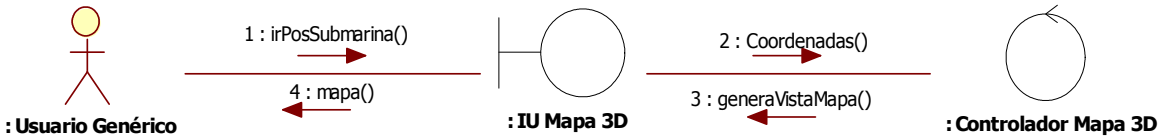


Figura 5.15. Diagrama colaboración ver mundo submarino en mapa 3D.

Mover mapa 3D

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Mapa 3D	IU Mapa 3D	

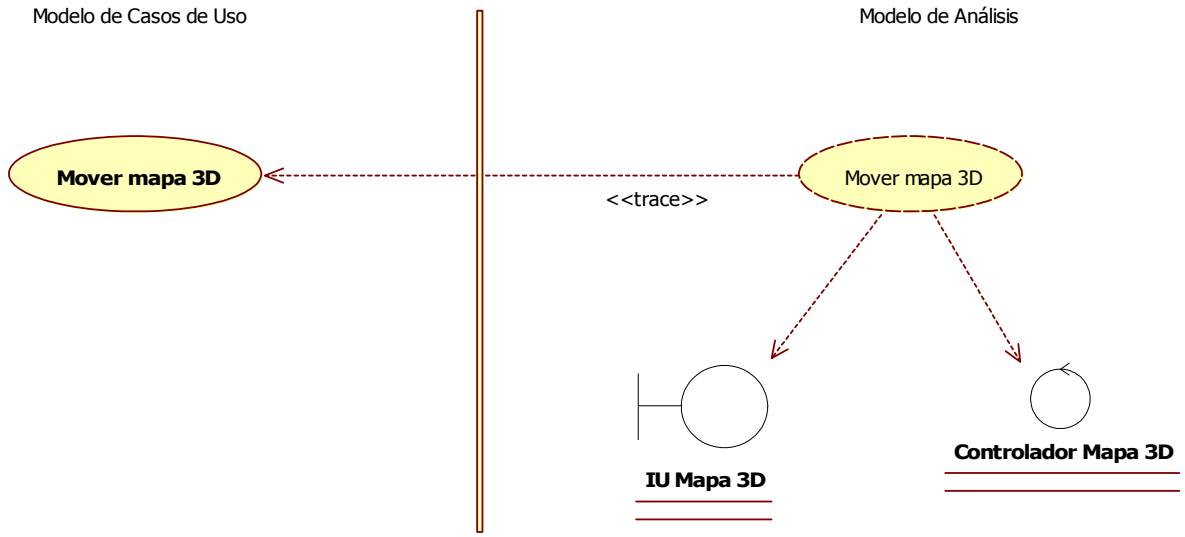


Figura 5.16. Traza (Caso de uso – Análisis) mover mapa 3D.

El caso de uso es iniciado por un usuario genérico. El usuario interactúa con la interfaz de usuario para que se muestre una nueva localización en el mapa 3D. Para ello especifica unas nuevas coordenadas implícitamente. El controlador recibe esa petición, la procesa y genera una nueva vista del mapa centrada en esa posición a través de la interfaz de usuario.

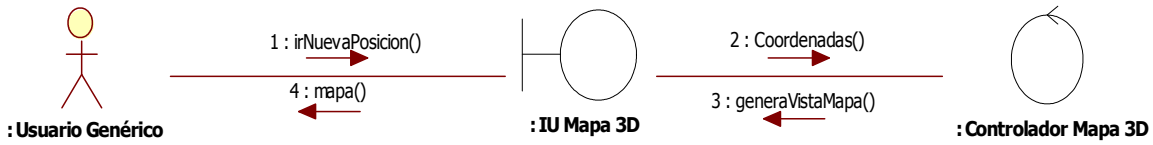


Figura 5.17. Diagrama colaboración mover mapa 3D.

Hacer zoom en el mapa 3D

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Mapa 3D	IU Mapa 3D	

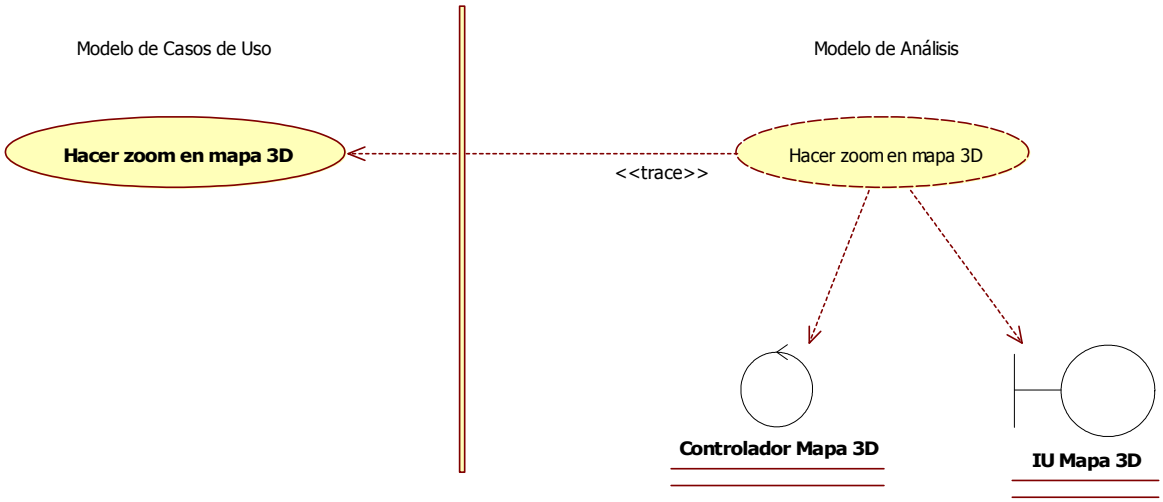


Figura 5.18. Traza (Caso de uso – Análisis) hacer zoom en mapa 3D.

Un usuario genérico, una vez está situado en la localización deseada, quiere obtener una vista mas cercana o más general. Para ello se hace uso del zoom. El usuario genera el zoom mediante movimientos de scroll. El controlador recibe la petición y genera una nueva vista que es mostrada al usuario a través de la interfaz de usuario.

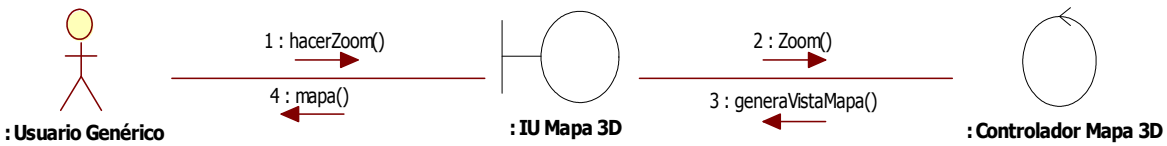


Figura 5.19. Diagrama colaboración hacer zoom en mapa 3D.

Ver dispositivos en el mapa 3D

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Mapa 3D	IU Mapa 3D	Dispositivo
Controlador Dispositivo		

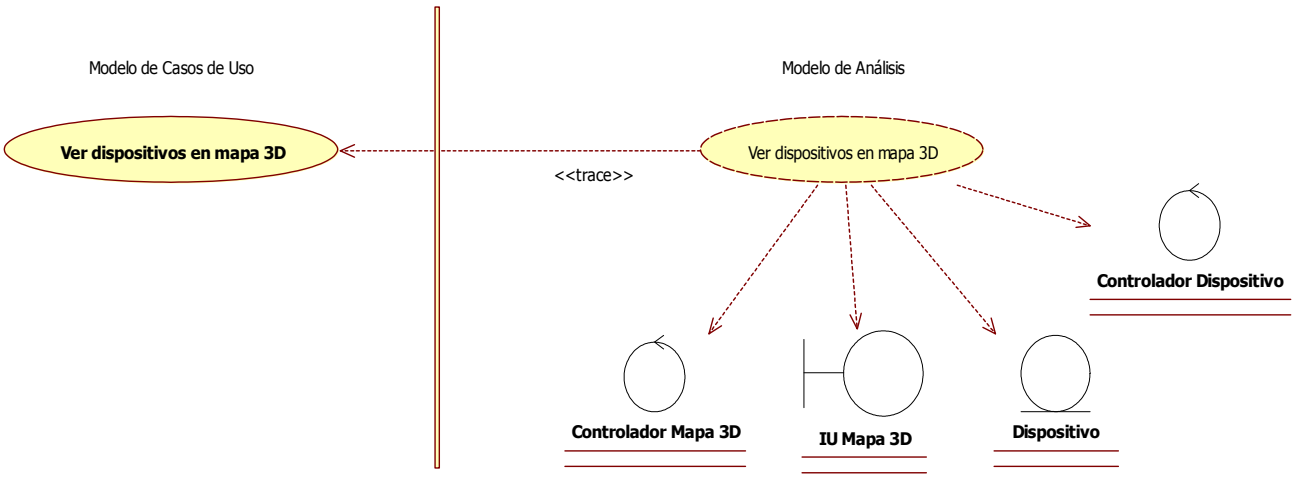


Figura 5.20. Traza (Caso de uso – Análisis) ver dispositivos en mapa 3D.

Un usuario genérico inicia el caso de uso al colocarse sobre una localización en el mapa 3D. El controlador del mapa 3D captura el evento. En ese momento comunica con Controlador Dispositivo solicitándole los dispositivos situados en el rango de coordenadas que mostrará el mapa. Controlador Dispositivo recibe la petición y obtiene de Dispositivo todos los dispositivos que cumplen la condición y le es devuelto a Controlador Mapa 3D. En ese momento genera la vista del mapa en la posición seleccionada incluyendo los dispositivos y se le muestra al usuario.

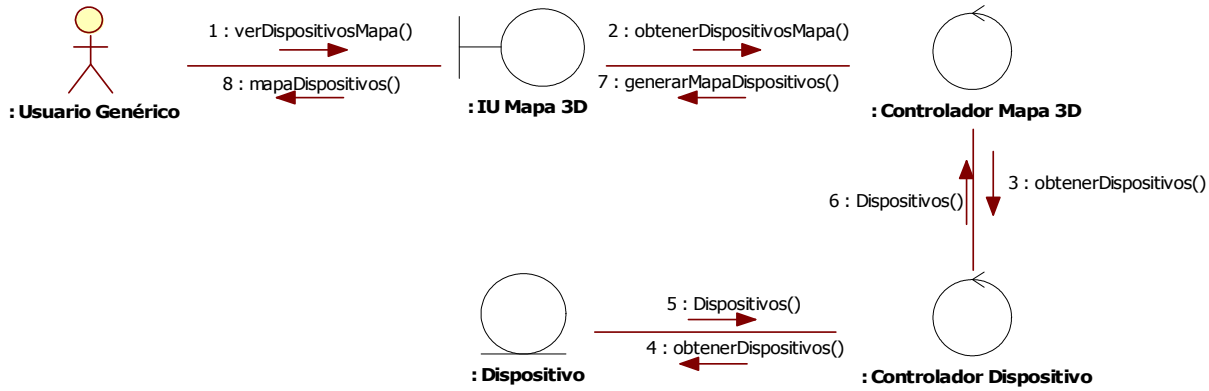


Figura 5.21. Diagrama colaboración ver dispositivos en mapa 3D.

Ver dispositivos en el menú

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Mapa 3D	IU Mapa 3D	Dispositivo
Controlador Dispositivo	IU Menú	
Controlador Menú		

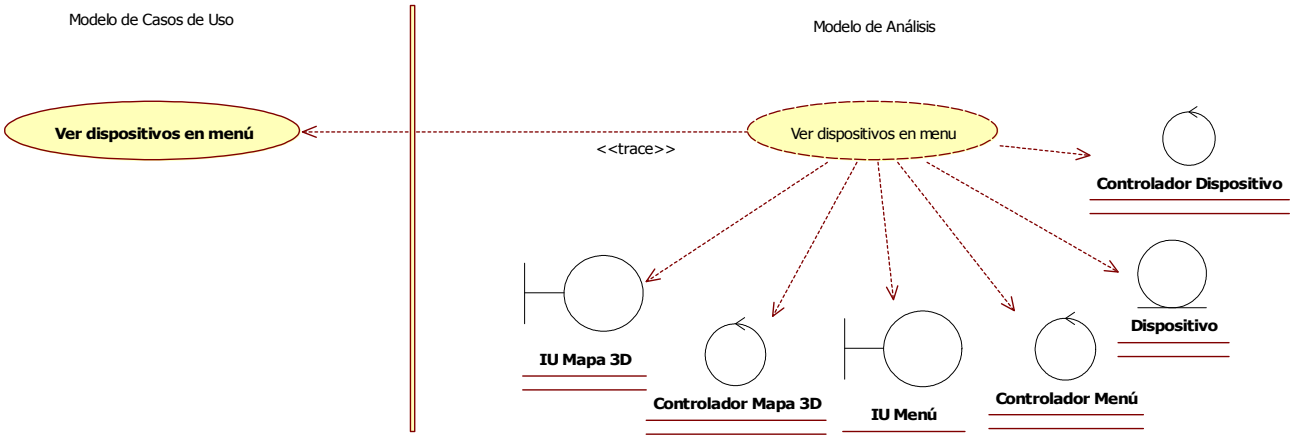


Figura 5.22. Traza (Caso de uso – Análisis) ver dispositivos en menú.

El controlador del menú solicita a Controlador Dispositivo la lista de todos los dispositivos que componen el sistema. Éste lo obtiene de Dispositivo y se lo devuelve a Controlador Menú. El controlador genera un menú desplegable con todos los dispositivos agrupados por tipo y se le muestra al usuario. Cuando un usuario hace click sobre un dispositivo el controlador del menú informa al controlador del mapa y éste genera la vista del mapa en la posición del dispositivo y se le muestra al usuario.

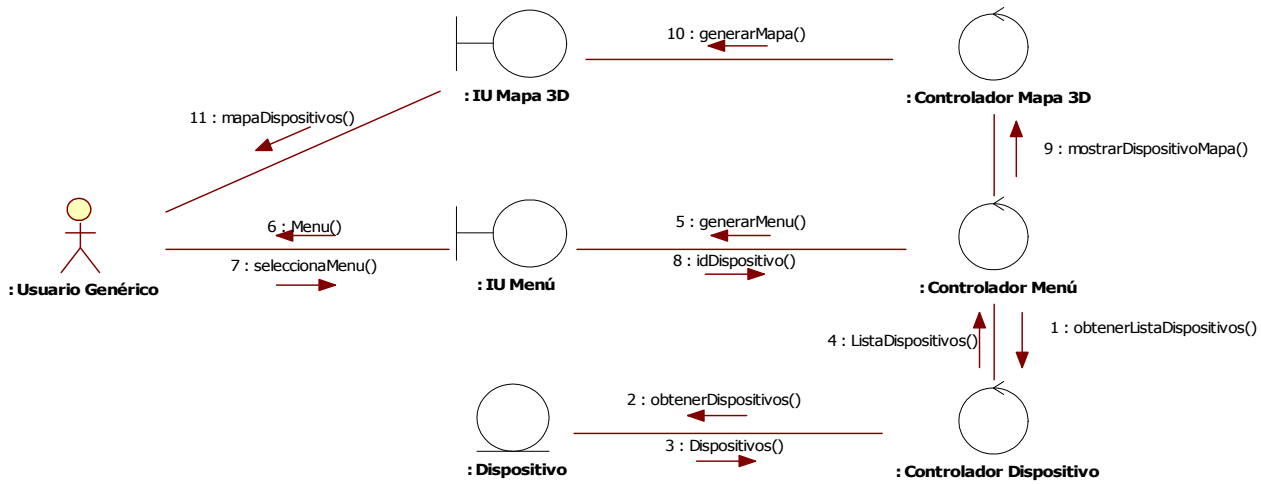


Figura 5.23. Diagrama colaboración ver dispositivos en menú.

Ver información del dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo

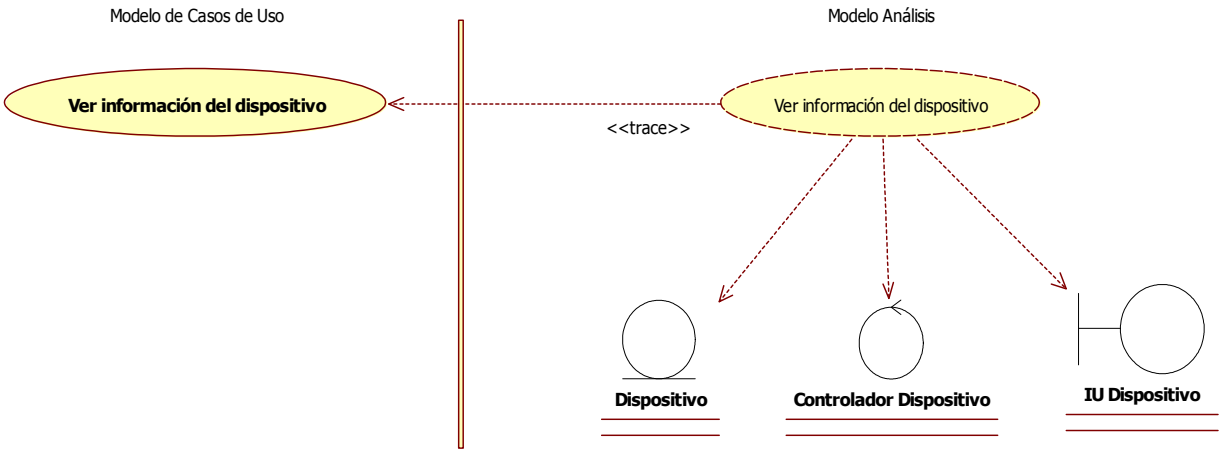


Figura 5.24. Traza (Caso de uso – Análisis) ver información del dispositivo.

Este caso de uso se inicia cuando un usuario hace doble click sobre un dispositivo en el mapa. En ese momento Controlador Dispositivo recibe el identificador del dispositivo y solicita la información perteneciente a éste a Dispositivo. Una vez recibe dicha información genera una burbuja donde se le muestra al usuario a través de IU Dispositivo.

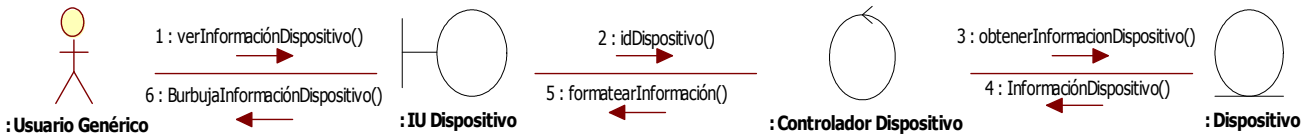


Figura 5.25. Diagrama colaboración ver información del dispositivo.

Ver lectura de datos del dispositivo

El siguiente caso de uso es una extensión del anterior. Una vez se muestra la información de un dispositivo al usuario mediante la burbuja, el usuario accede a ver las lecturas de los sensores del dispositivo.

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo

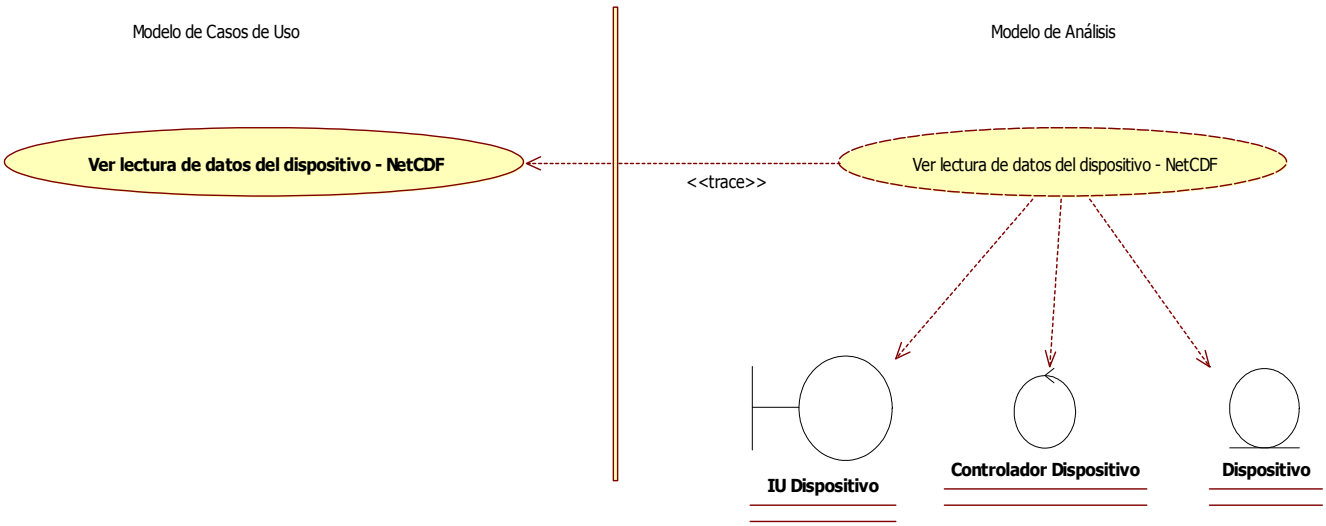


Figura 5.26. Traza (Caso de uso – Análisis) ver lectura de datos del dispositivo.

El usuario solicita ver la lectura de los sensores del dispositivo. El controlador solicita la información a Dispositivo y una vez la recibe formatea los datos y se los muestra al usuario.

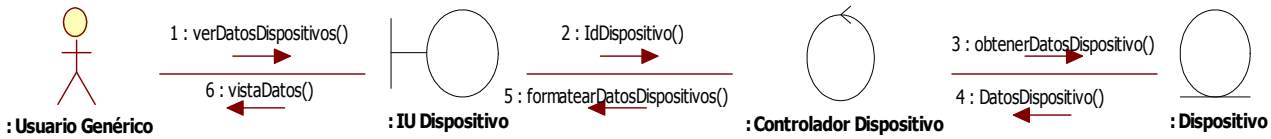


Figura 5.27. Diagrama colaboración ver lectura de datos del dispositivo.

Ver descripción de sensor – SensorML

Cuando la información de un dispositivo es mostrada al usuario mediante la burbuja, el usuario accede a ver la descripción de los sensores presentes en el dispositivo.

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo

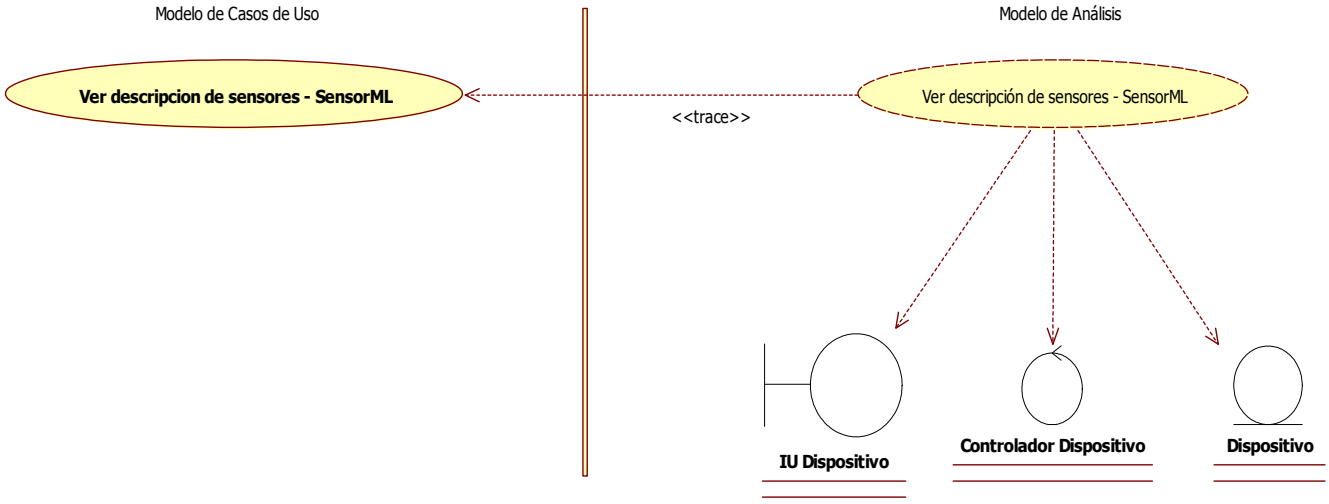


Figura 5.28. Traza (Caso de uso – Análisis) ver descripción del sensor.

El usuario solicita ver la descripción del sensor. El controlador solicita la información a Dispositivo y una vez la recibe formatea los datos y se los muestra al usuario.

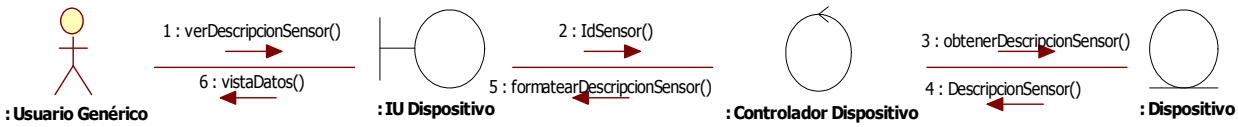


Figura 5.29. Diagrama colaboración ver descripción del sensor.

5.3.2.2. Usuarios

La siguiente imagen muestra cada una de las colaboraciones que contiene el paquete de análisis Usuarios. Este paquete tiene una relación de traza con el paquete de casos de uso con el mismo nombre.

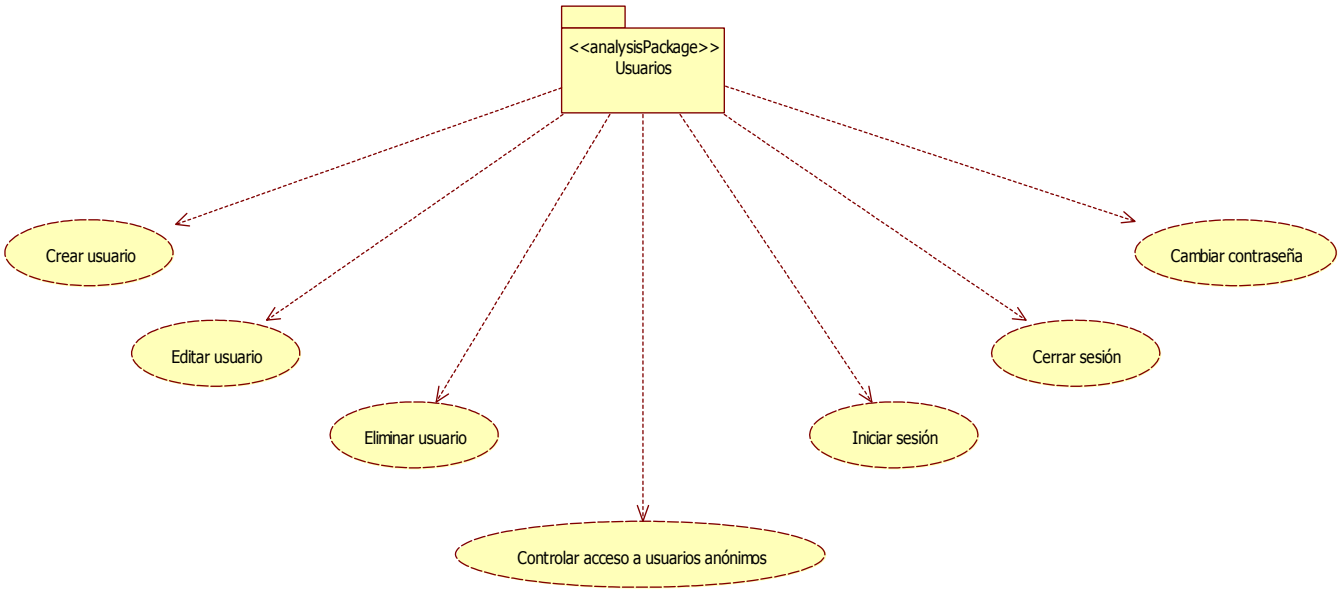


Figura 5.30. Realización casos de uso usuarios.

A continuación se muestra el diagrama de clases donde se identifican las clases que participan en dicho paquete y las relaciones entre ellas. Además también se muestran los perfiles de usuarios que harán uso de ellas.

En este caso se diferencian dos de los perfiles de usuarios que ya se ha explicado en el capítulo "Captura de requisitos". Un primero será el perfil de administrador, y es quien tendrá acceso a las secciones de administración de cuentas de usuarios. Y por otro lado está el perfil de usuario registrado que hará uso de las sesiones y de su propia cuenta de usuario.

Control

- **Controlador Acceso.** Gestiona el grado de privacidad del cliente de la aplicación.
- **Controlador Usuario.** Gestiona la información y realiza las acciones oportunas sobre las cuentas de usuario.
- **Controlador Sesion.** Crea y destruye sesiones de usuario.

Interfaz

- **IU Administración.** Interfaz de usuario de la parte administrativa de la aplicación.
- **IU Usuario.** Permite al usuario introducir o editar la información referente a las cuentas de usuario.
- **IU Cambio Contraseña.** Permite a un usuario registrado modificar su contraseña.
- **IU Sesion.** Zona de la interfaz donde el usuario introduce sus credenciales de acceso para acceder al sistema como usuario registrado.

Entidad

- **Acceso.** Contiene el estado actual de la privacidad de la aplicación.
- **Usuario.** Contiene la información referente a las cuentas de usuario.
- **Sesion.** Contiene la información de las sesiones de usuario.

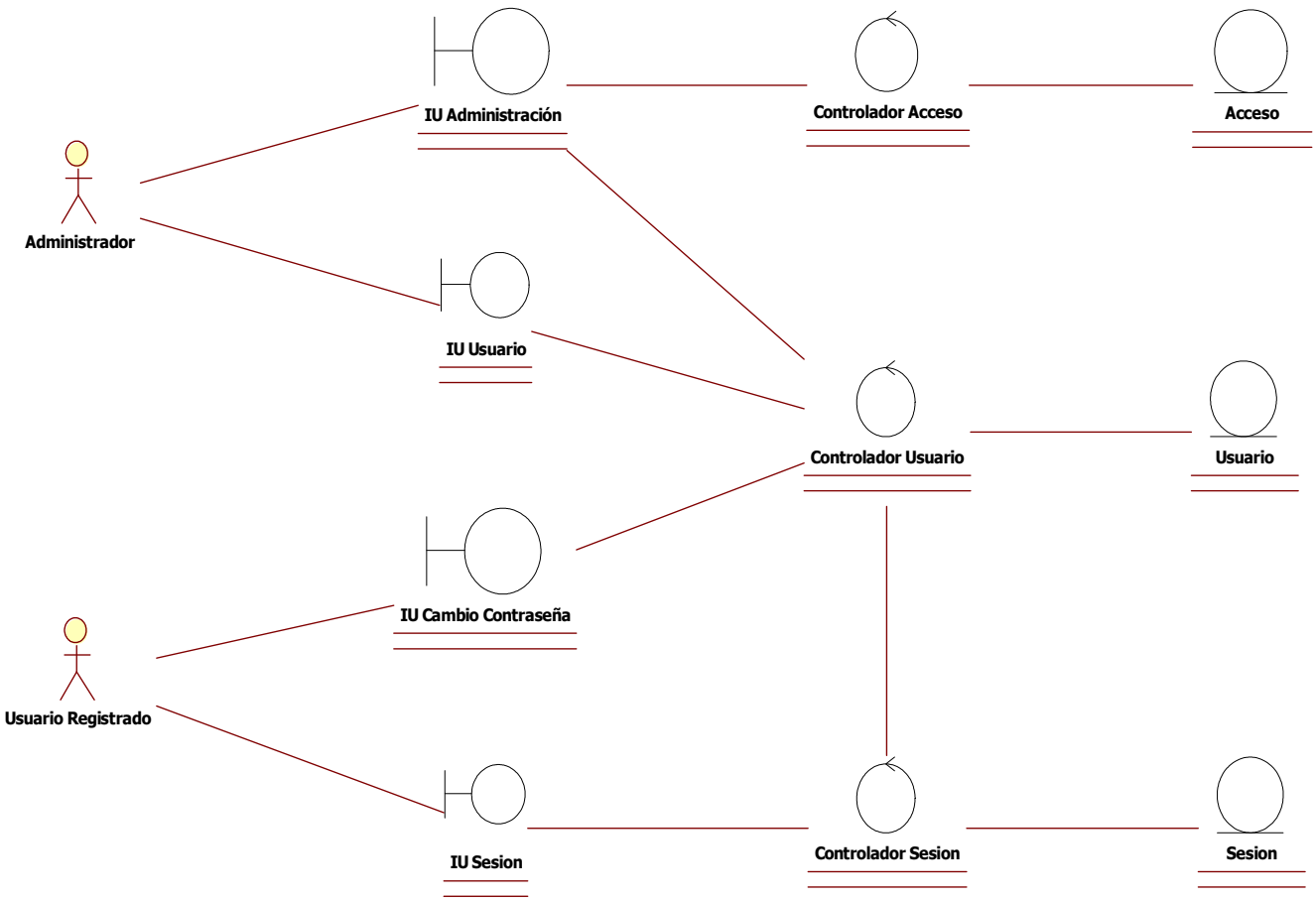


Figura 5.31. Diagrama de colaboración de usuarios.

Crear usuario

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Usuario	IU Administración	Usuario
	IU Usuario	

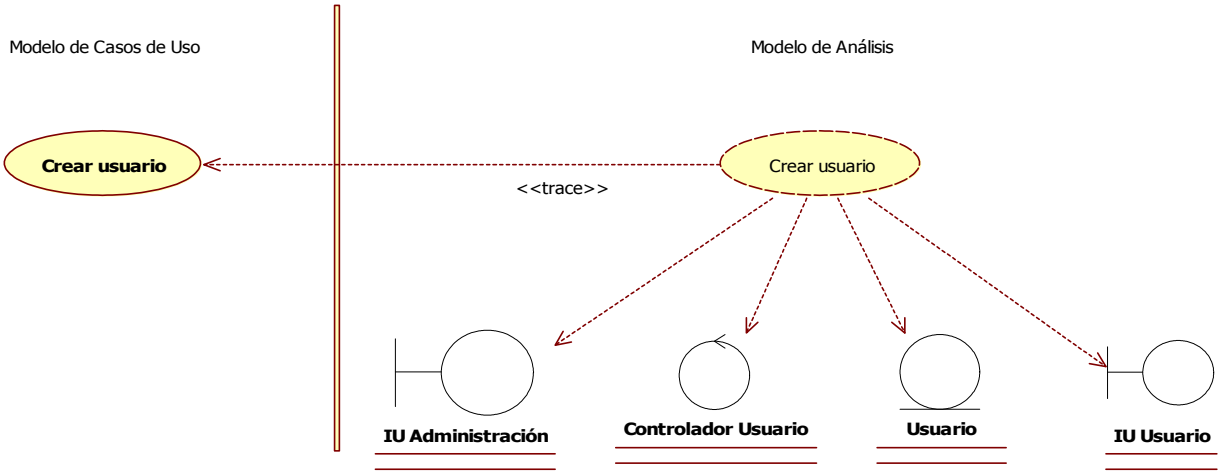


Figura 5.32. Traza (Caso de uso – Análisis) crear usuario.

Este caso de uso comienza cuando el administrador solicita crear una nueva cuenta de usuario a través de la interfaz de administración. El Controlador Usuario genera un formulario donde el propio administrador introduce los datos de la cuenta de usuario. Estos datos llegan al controlador que lo almacena en Usuario.

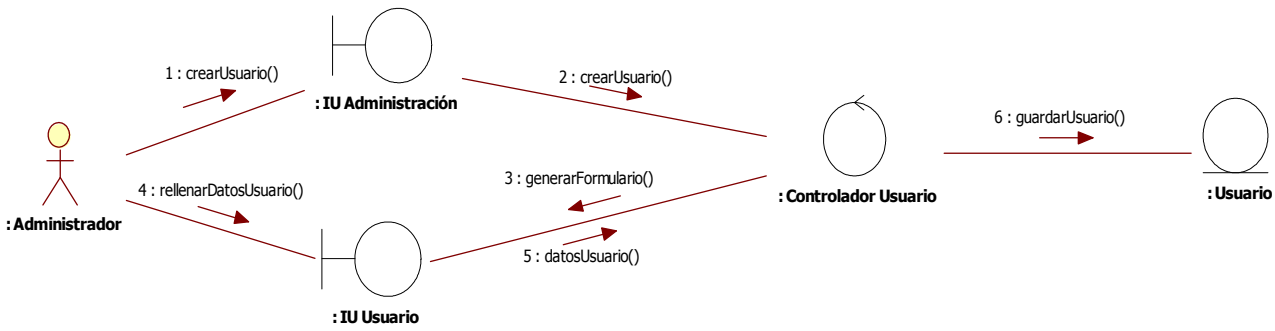


Figura 5.33. Diagrama colaboración crear usuario.

Editar usuario

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Usuario	IU Administración	Usuario
	IU Usuario	

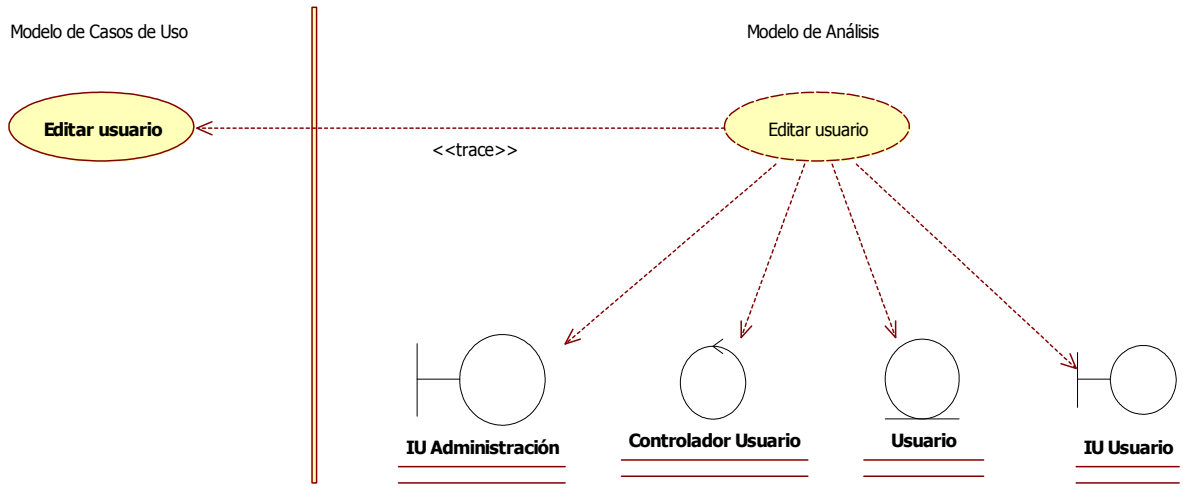


Figura 5.34. Traza (Caso de uso – Análisis) editar usuario.

Este caso de uso es muy similar al anterior, con la diferencia de que los datos de la cuenta de usuario son insertados en el formulario.

Se inicia cuando el administrador selecciona un usuario para editarlo en la interfaz de administración. En ese momento el controlador recupera los datos de la cuenta de la entidad Usuario. Con los datos genera un formulario que le es mostrado al administrador. Éste modifica los datos que considera oportuno y le da a guardar. El controlador recibe los datos y los almacena nuevamente en Usuario.

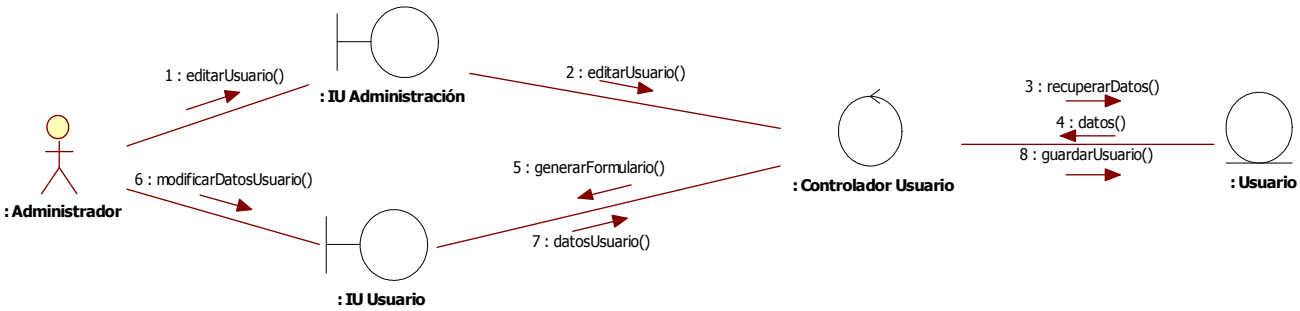


Figura 5.35. Diagrama colaboración editar usuario.

Eliminar usuario

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Usuario	IU Usuario	Usuario

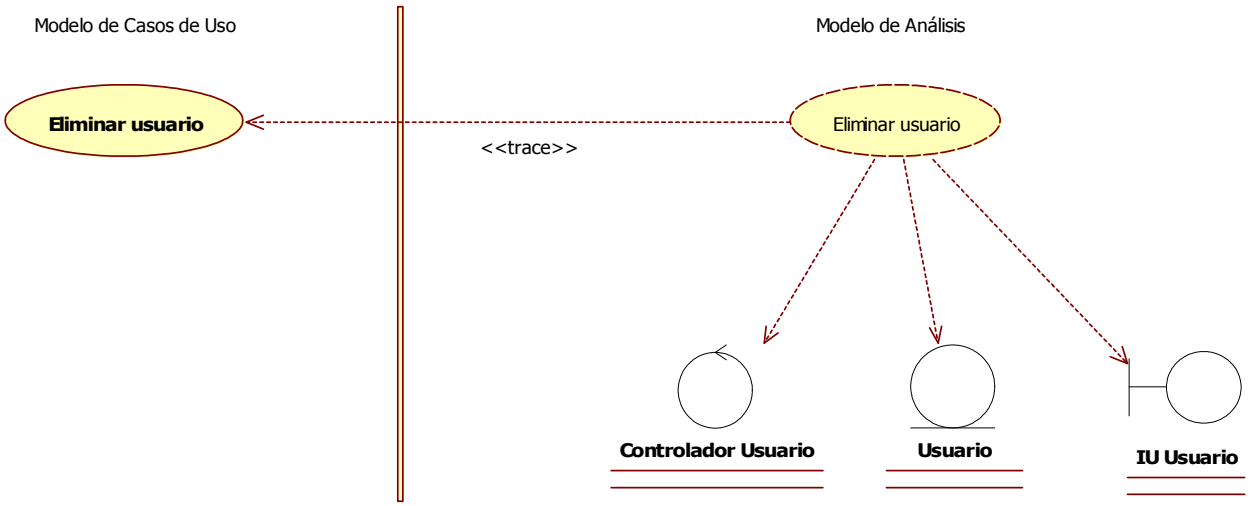


Figura 5.36. Traza (Caso de uso – Análisis) eliminar usuario.

El administrador selecciona un usuario que desea eliminar. Cuando la petición le llega al controlador éste ejecuta la acción y elimina la cuenta de usuario de la entidad Usuario.

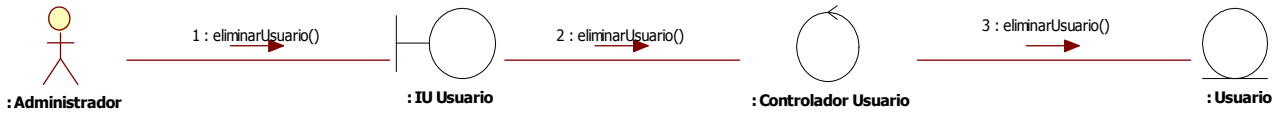


Figura 5.37. Diagrama colaboración eliminar usuario.

Controlar acceso a usuarios anónimos

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Acceso	IU Administración	Acceso

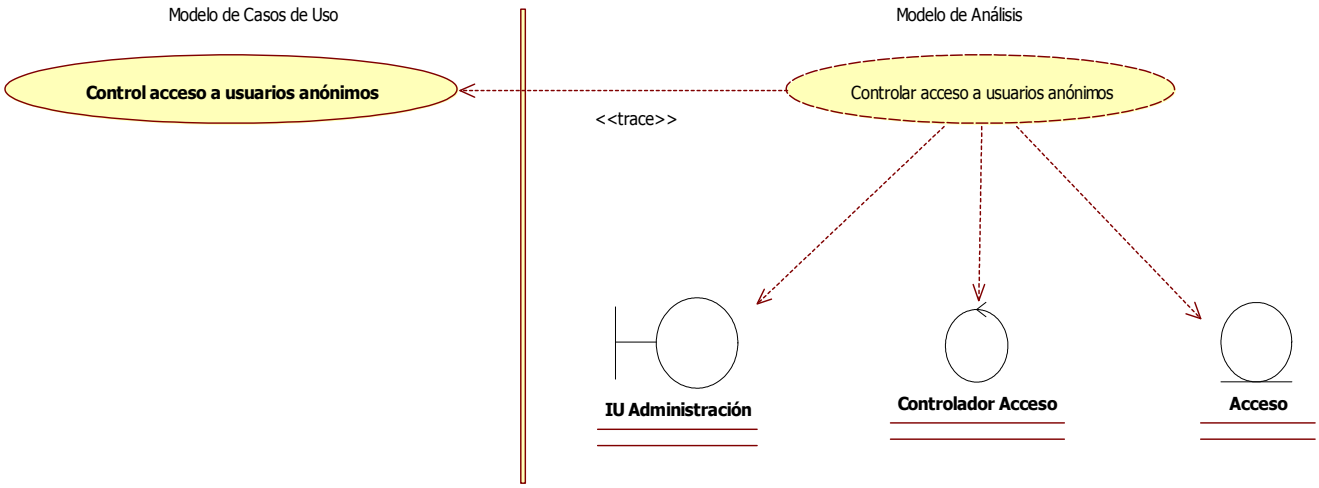


Figura 5.38. Traza (Caso de uso – Análisis) controlar acceso a usuarios anónimos.

Este caso de uso se da cuando el administrador quiere modificar el grado de privacidad de la aplicación. Con ello se pretende controlar que en ciertos momentos, sólo usuarios registrados puedan acceder al cliente de la aplicación.

El caso de uso comienza cuando el administrador modifica el valor del acceso en la interfaz de administración. En ese momento el controlador recibe el valor y lo almacena en la entidad Acceso.

Figura 5.39. Diagrama colaboración controlar acceso a usuarios anónimos.

Iniciar sesión

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Usuario	IU Sesion	Usuario
Controlador Sesion		Sesion

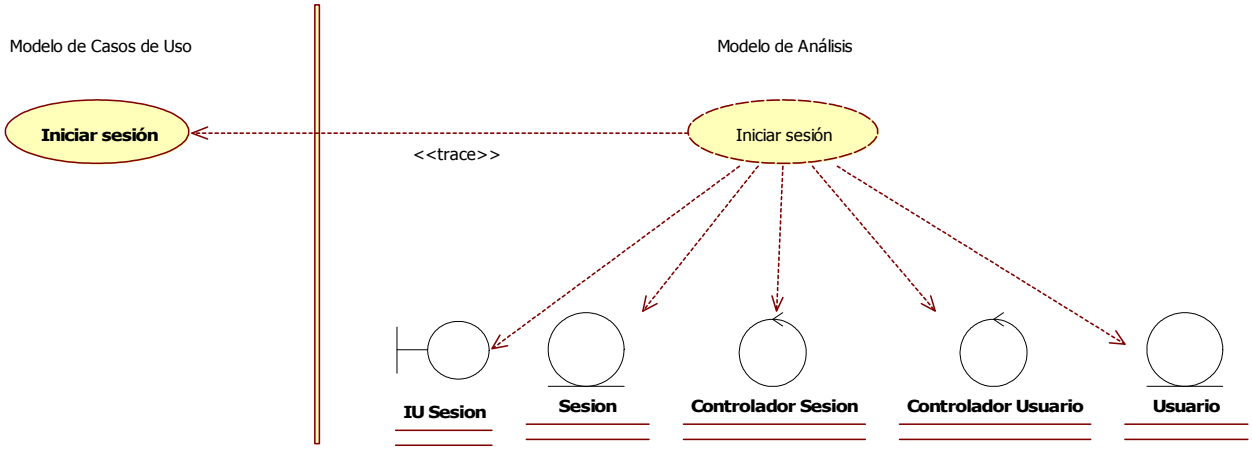


Figura 5.40. Traza (Caso de uso – Análisis) iniciar sesión.

El caso de uso se inicia cuando un usuario registrado intenta acceder al sistema con su cuenta de usuario. El usuario introduce los credenciales de acceso en la interfaz de usuario. El controlador Sesion recibe los credenciales y solicita a Controlador Usuario la contraseña codificada perteneciente al usuario recibido. Éste los recupera desde Usuario y se los devuelve a Controlador Sesión. En ese momento compara si coinciden las dos contraseñas codificadas y si es así crea una sesión de usuario.

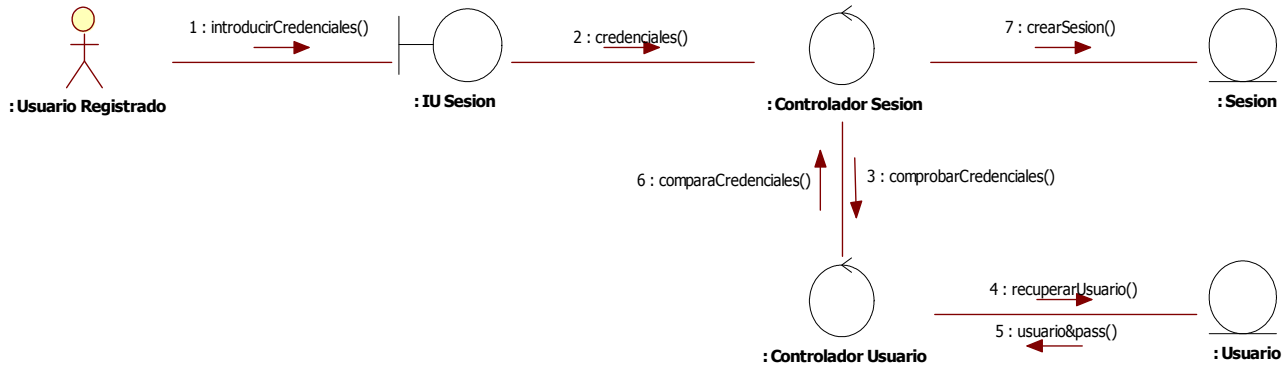


Figura 5.41. Diagrama colaboración iniciar sesión.

Cerrar sesión

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Sesion	IU Sesion	Sesion

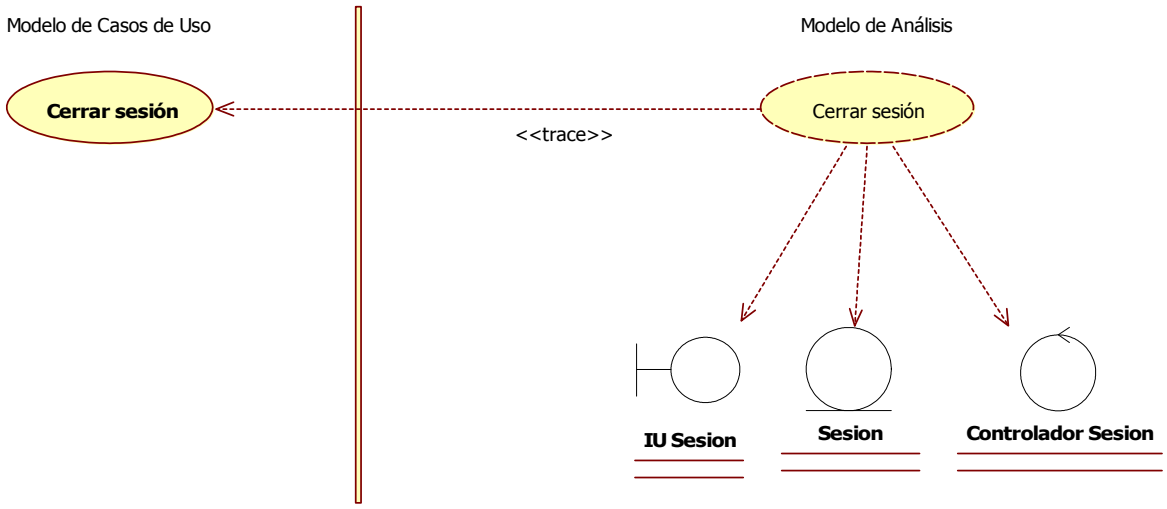


Figura 5.42. Traza (Caso de uso – Análisis) cerrar sesión.

Cuando un usuario registrado desea abandonar el sistema se inicia este caso de uso. En ese momento el usuario solicita cerrar la sesión. Entonces el controlador elimina la sesión de la entidad Sesion. A partir de ese momento el usuario se convierte en un usuario anónimo.

Figura 5.43. Diagrama colaboración cerrar sesión.

Cambiar contraseña

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Usuario	IU Cambio Contraseña	Usuario
	IU Usuario	

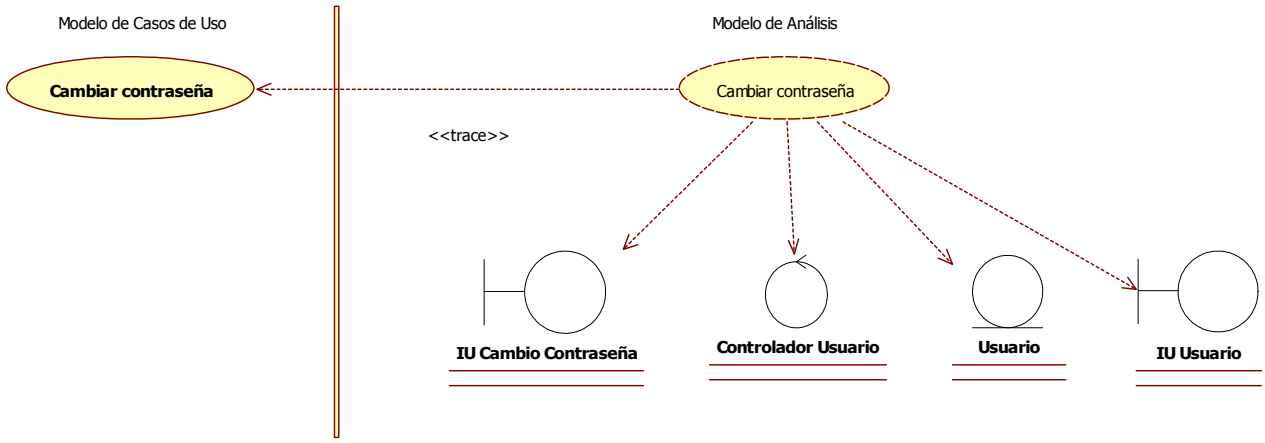


Figura 5.44. Traza (Caso de uso – Análisis) cambiar contraseña.

Un usuario registrado y que ha accedido al sistema solicita cambiar su contraseña. El controlador genera un formulario para que el usuario introduzca la nueva contraseña y la contraseña anterior. El usuario rellena el formulario y el controlador lo recibe. Cuando comprueba que la nueva contraseña es válida la almacena en Usuario.

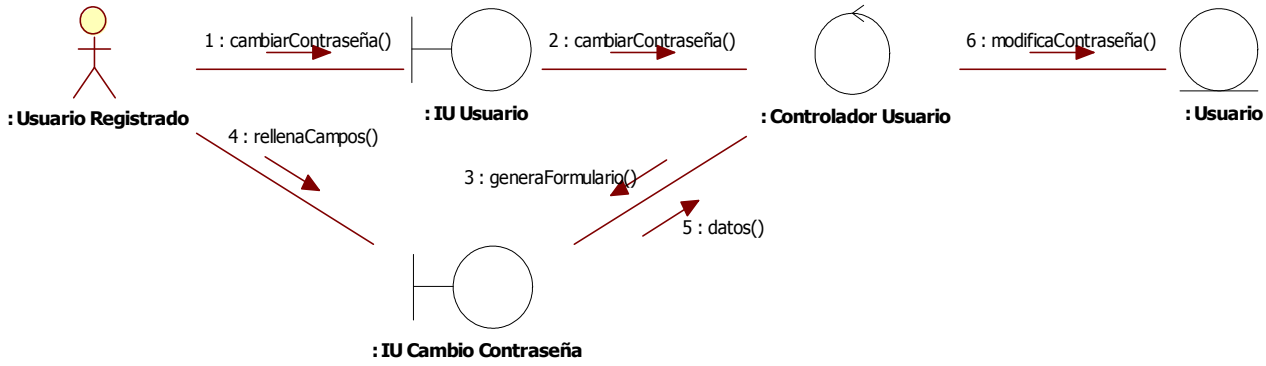


Figura 5.45. Diagrama colaboración cambiar contraseña.

5.3.2.3. Dispositivos

La siguiente imagen se corresponde con el paquete de análisis Dispositivos que mantiene una relación de traza con el paquete de casos de uso con el mismo nombre. En él se incluyen las colaboraciones de casos de uso que se muestran a continuación.

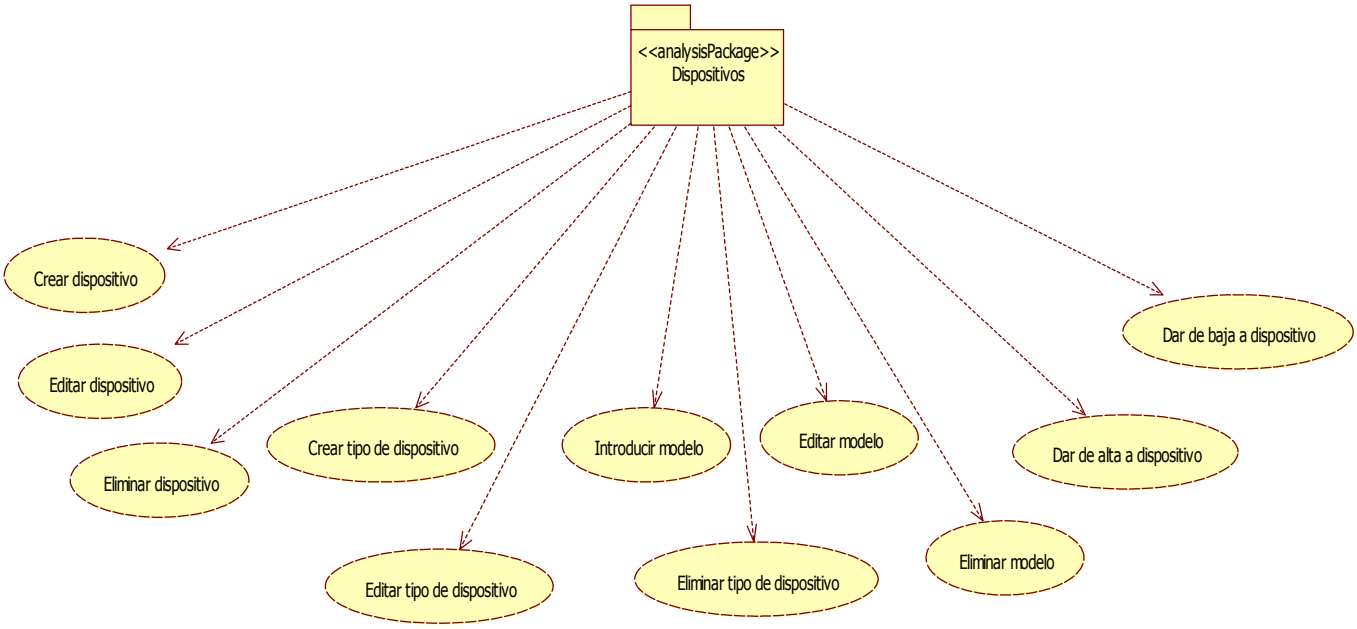


Figura 5.46. Realización casos de uso dispositivos.

En el siguiente diagrama de clases, al igual que se ha hecho en los apartados anteriores, se muestra las clases participantes en estos casos de uso y la relación entre ellas.

Principalmente, el actor que interviene en estos casos de uso es el perfil de usuario Editor. Él es el encargado de gestionar y mantener la información relativa a los dispositivos presentes en el sistema.

Control

- **Controlador Dispositivo.** Se encarga de gestionar la información relativa a los dispositivos y ejecutar las acciones sobre ella.
- **Controlador ModeloDispositivo.** Es el encargado de gestionar la información relativa a la representación gráfica de un dispositivo en el mapa 3D.
- **Controlador TipoDispositivo.** Se encarga de la gestión de los diferentes tipos de dispositivos y de las acciones sobre ellos.

Interfaz

- **IU Administración.** Interfaz de usuario de la parte administrativa de la aplicación.
- **IU Dispositivo.** Permite al editor introducir y editar la información relativa a los dispositivos.
- **IU ModeloDispositivo.** Permite al editor introducir y editar la información relativa a la representación gráfica de los dispositivos.
- **IU TipoDispositivo.** Permite al editor introducir y editar la información relativa a los tipos de dispositivos.

Entidad

- **Dispositivo.** Contiene la información referente a los dispositivos.
- **ModeloDispositivo.** Contiene la información referente a la representación gráfica de los dispositivos.
- **TipoDispositivo.** Contiene la información referente a los tipos de dispositivos.

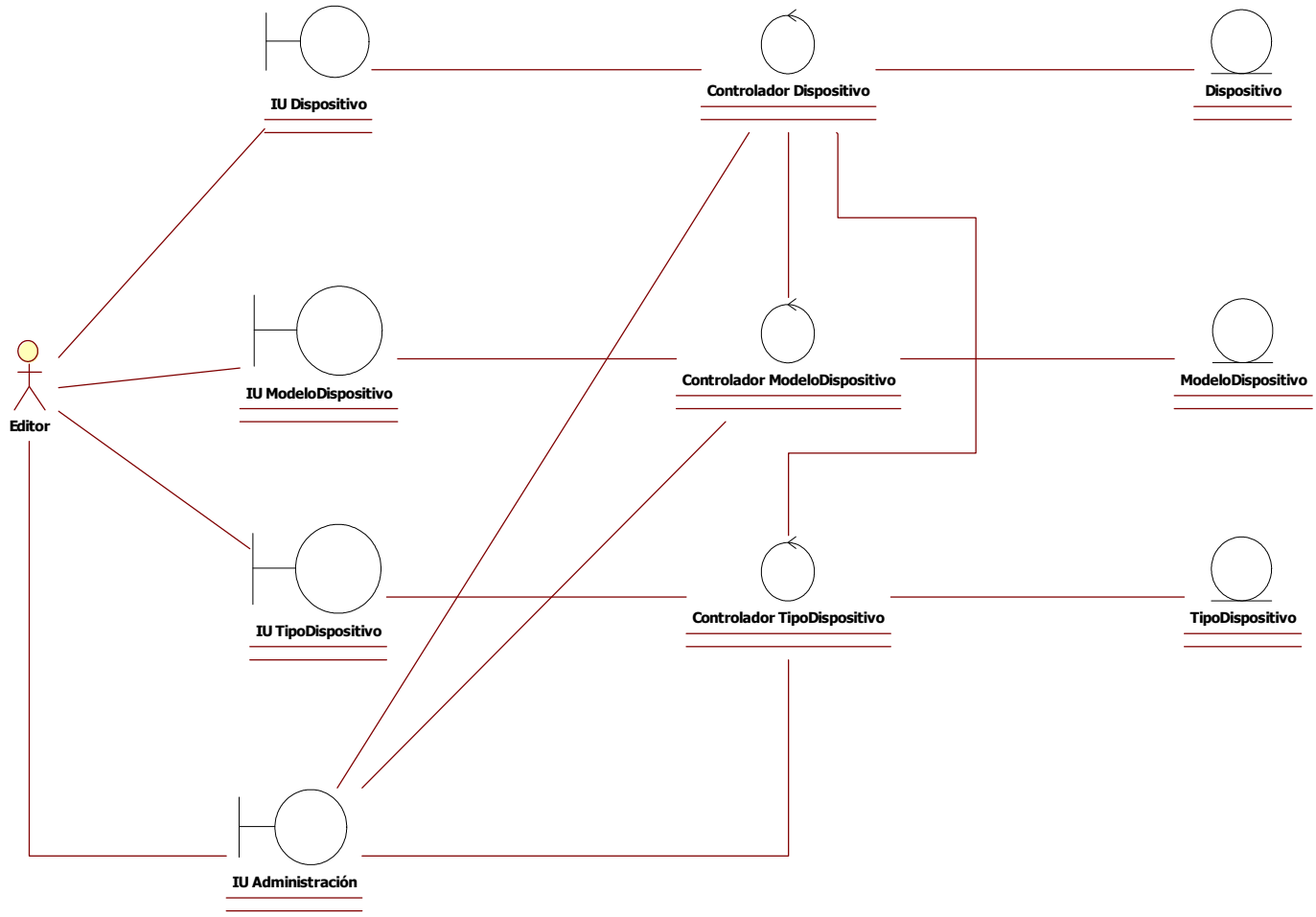


Figura 5.47. Diagrama de colaboración dispositivos.

Crear dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo
Controlador TipoDispositivo	IU Administración	TipoDispositivo
Controlador ModeloDispositivo		ModeloDispositivo

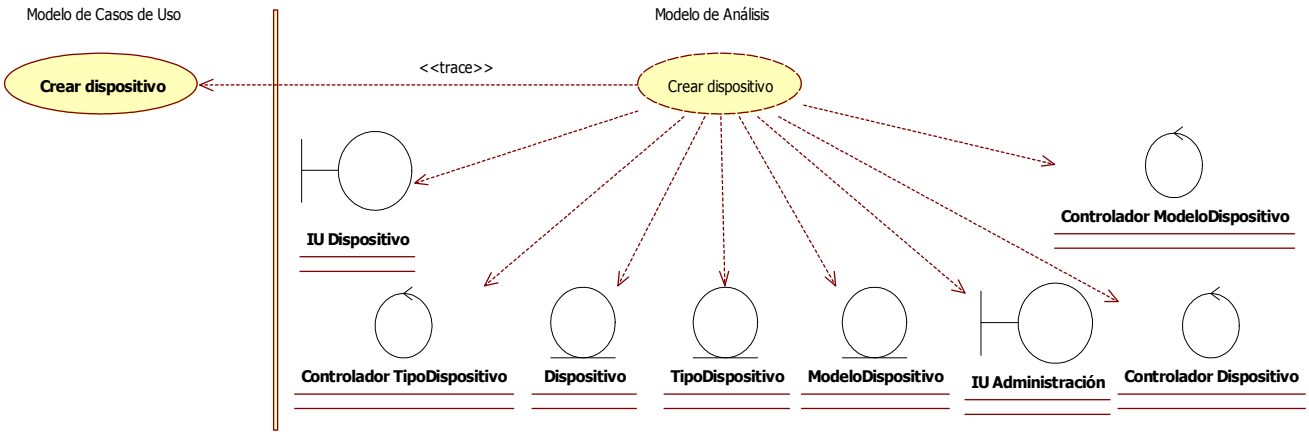


Figura 5.48. Traza (Caso de uso – Análisis) crear dispositivo.

El editor solicita crear un nuevo dispositivo. Controlador Dispositivo solicita los tipos de dispositivos existentes en el sistema a Controlador TipoDispositivo y hace lo propio con los modelos 3D a Controlador ModeloDispositivo. Cada uno accede a su identidad para recuperar la información y devolvérsela al controlador de dispositivos. Cuando recibe la información genera un formulario que será rellenado por el editor con los datos del dispositivo. Estos datos le llegan al controlador que los almacena en la entidad Dispositivo.

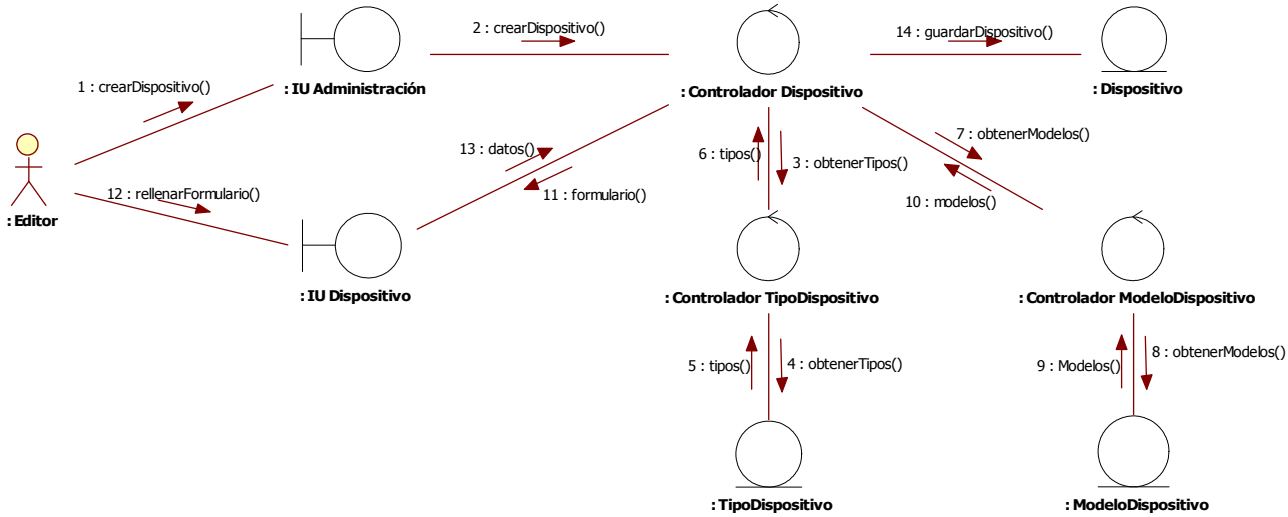


Figura 5.49. Diagrama colaboración crear dispositivo.

Editar dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo
Controlador TipoDispositivo	IU Administración	TipoDispositivo
Controlador ModeloDispositivo		ModeloDispositivo

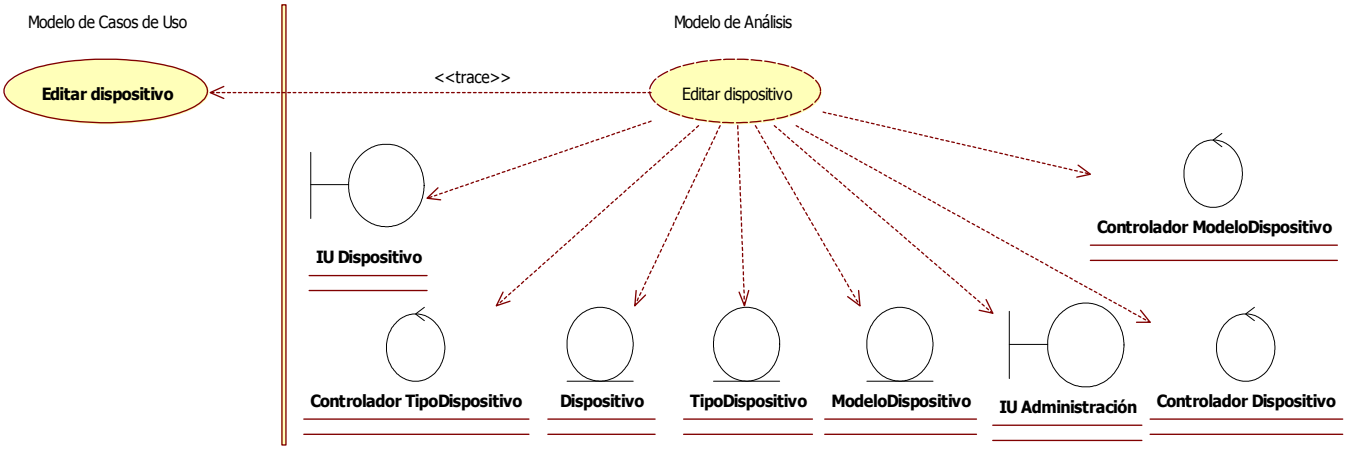


Figura 5.50. Traza (Caso de uso – Análisis) editar dispositivo.

El funcionamiento de este caso de uso es prácticamente el mismo que en el anterior con la salvedad que Controlador Dispositivo, antes de generar el formulario, accede a la entidad Dispositivo para recuperar la información del dispositivo seleccionado. Una vez tiene la información le devuelve el formulario completo al editor, que modificará la información que considere oportuna y le será devuelto al mismo controlador que lo almacena nuevamente en la entidad Dispositivo.

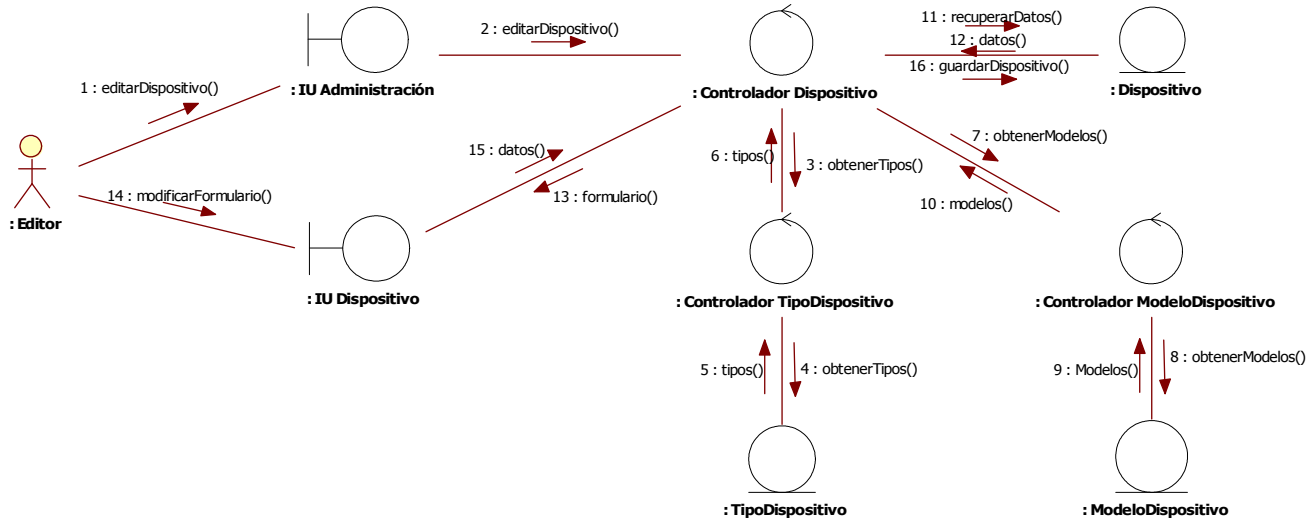


Figura 5.51. Diagrama colaboración editar dispositivo.

Eliminar dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo

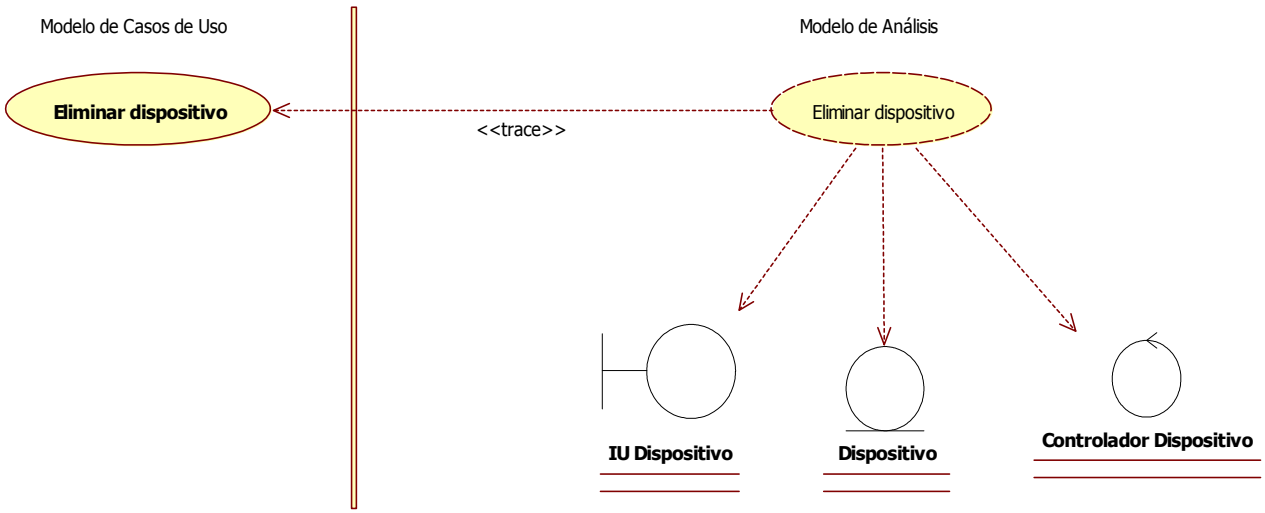


Figura 5.52. Traza (Caso de uso – Análisis) eliminar dispositivo.

Este caso de uso representa cuando un editor desea eliminar un dispositivo. El caso de uso comienza con la petición por parte del usuario de borrar un dispositivo. Cuando llega la petición al controlador éste accede a la entidad Dispositivo y lo elimina.

Figura 5.53. Diagrama colaboración eliminar dispositivo.

Crear tipo de dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador TipoDispositivo	IU TipoDispositivo	TipoDispositivo
	IU Administración	

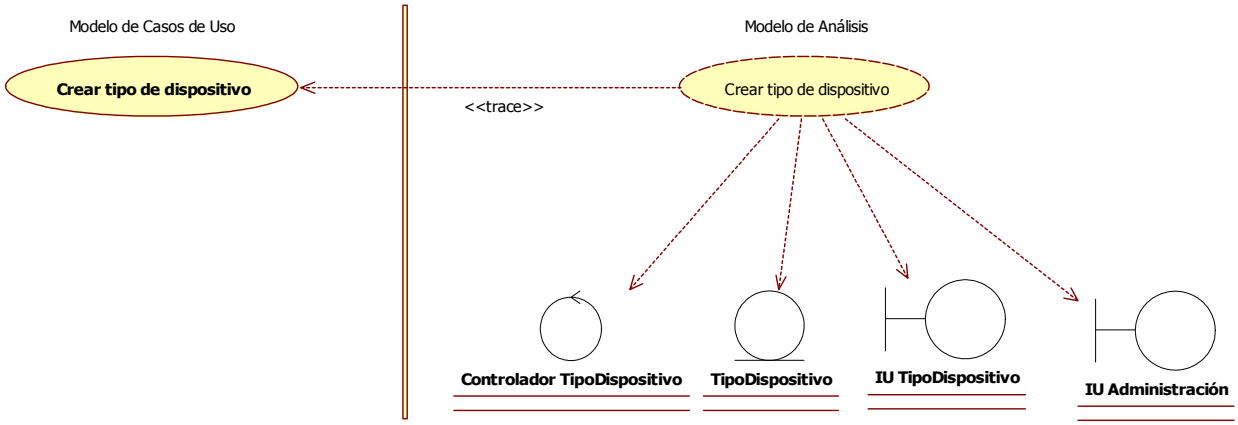


Figura 5.54. Traza (Caso de uso – Análisis) crear tipo de dispositivo.

El editor solicita crear un tipo de dispositivo. En ese momento el controlador captura la petición y genera un formulario vacío. El editor introduce los datos del tipo de dispositivo que desea crear y es enviado al controlador. Éste último, finalmente lo almacena en la entidad TipoDispositivo.

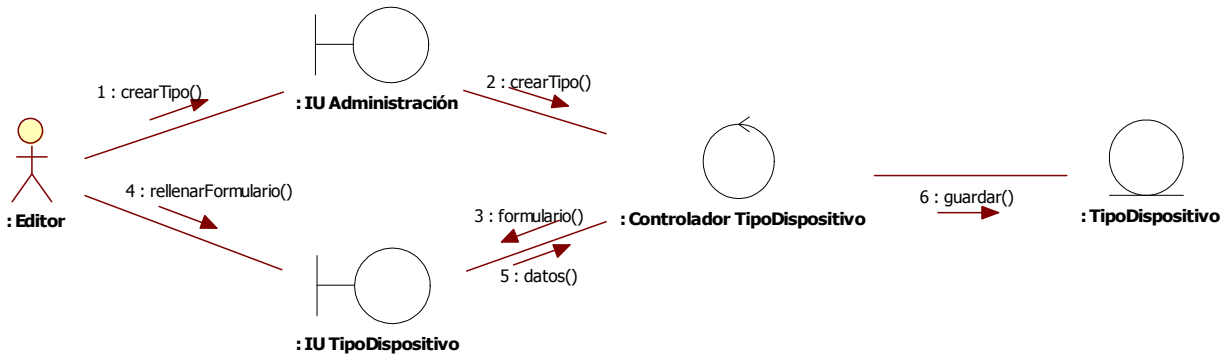


Figura 5.55. Diagrama colaboración crear tipo de dispositivo.

Editar tipo de dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador TipoDispositivo	IU TipoDispositivo	TipoDispositivo
	IU Administración	

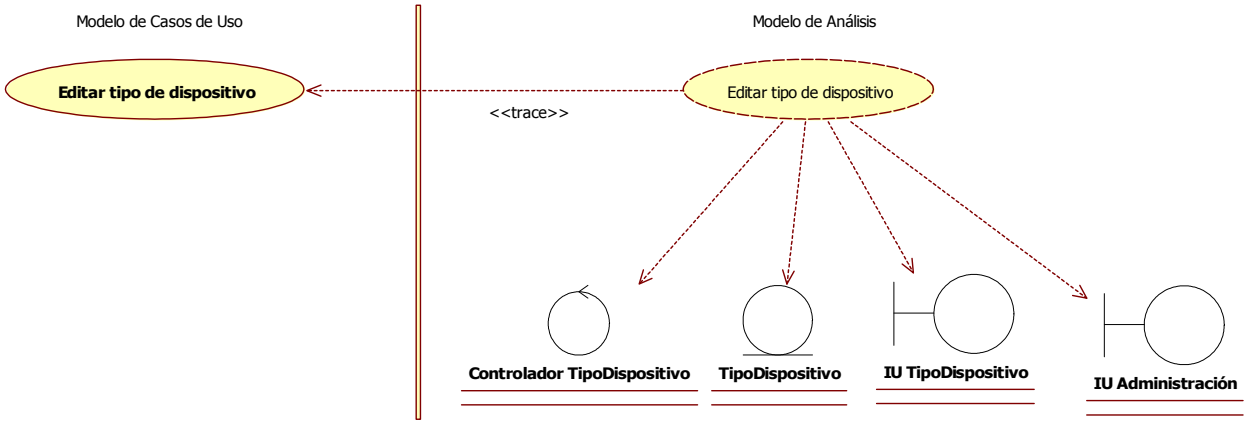


Figura 5.56. Traza (Caso de uso – Análisis) editar tipo de dispositivo.

El funcionamiento de este caso de uso es, en gran medida, igual al de crear tipo de dispositivo. El editor selecciona el dispositivo que desea modificar. El controlador accede a la entidad para recuperar los datos del dispositivo seleccionado. Posteriormente genera un formulario incluyendo los datos recuperados y se le muestra al usuario. El usuario modifica la información que considera necesaria y se envía todos los datos de vuelta al controlador. Por último, éste lo almacena nuevamente en la entidad TipoDispositivo.

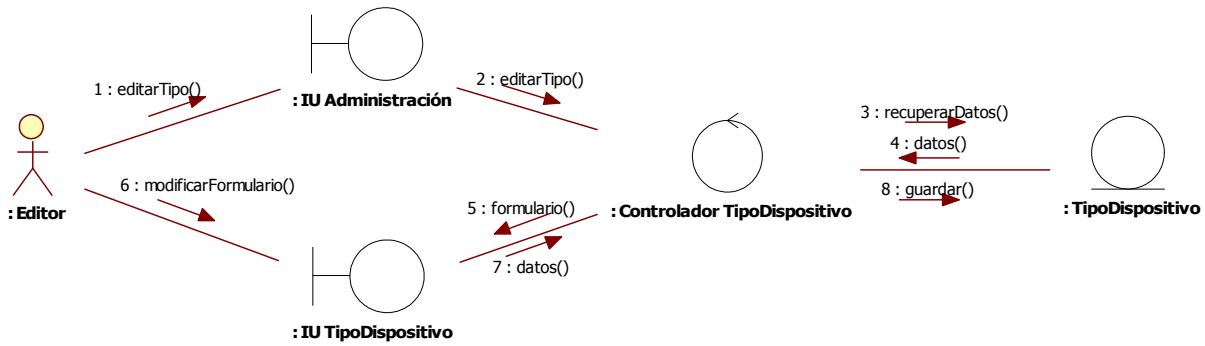


Figura 5.57. Diagrama colaboración editar tipo de dispositivo.

Eliminar tipo de dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador TipoDispositivo	IU TipoDispositivo	TipoDispositivo

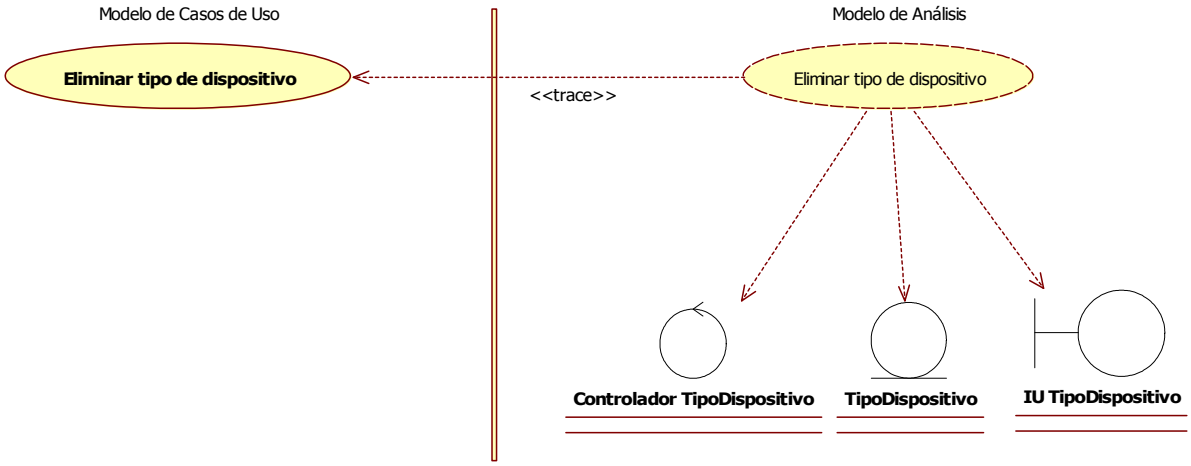


Figura 5.58. Traza (Caso de uso – Análisis) eliminar tipo de dispositivo.

El editor selecciona un dispositivo para ser eliminado. El controlador recibe esa petición. Entonces, Controlador TipoDispositivo accede a la entidad Tipo Dispositivo y elimina el dispositivo seleccionado.

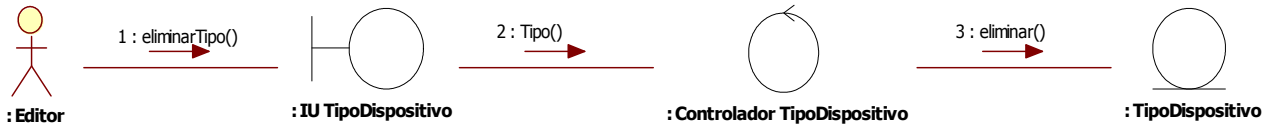


Figura 5.59. Diagrama colaboración eliminar tipo de dispositivo.

Introducir modelo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador ModeloDispositivo	IU ModeloDispositivo	ModeloDispositivo
	IU Administración	

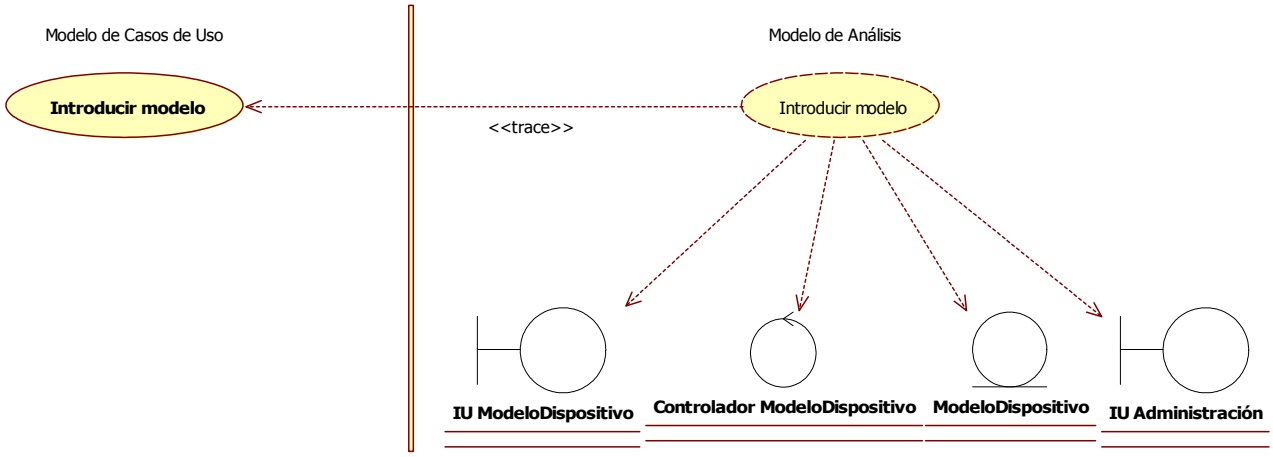


Figura 5.60. Traza (Caso de uso – Análisis) introducir modelo.

El editor, mediante la interfaz de administración, solicita introducir un modelo 3D de dispositivo en el sistema. El controlador recibe la solicitud y genera un formulario donde el editor insertará la información y la representación del dispositivo. Una vez llega esa información a Controlador ModeloDispositivo, éste lo almacena en la entidad ModeloDispositivo.

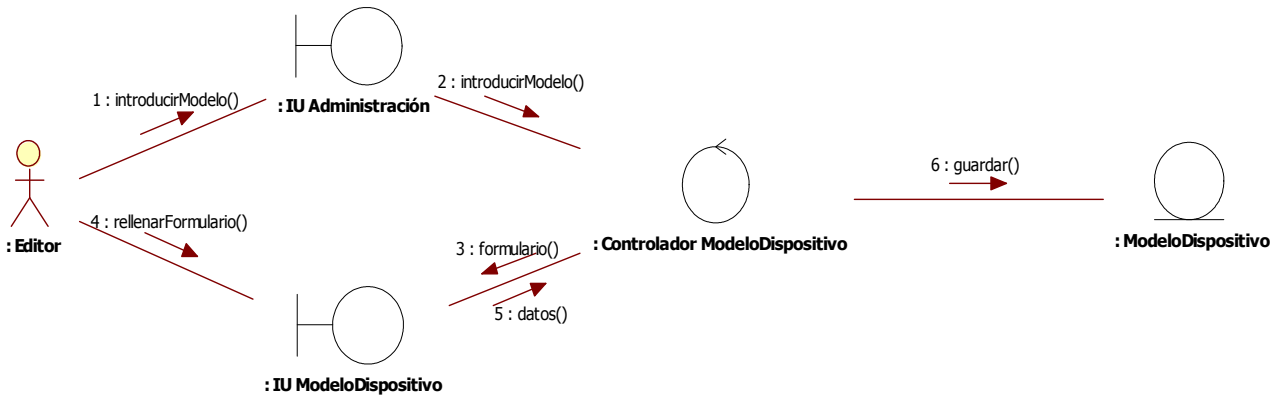


Figura 5.61. Diagrama colaboración introducir modelo.

Editar modelo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador ModeloDispositivo	IU ModeloDispositivo	ModeloDispositivo
	IU Administración	

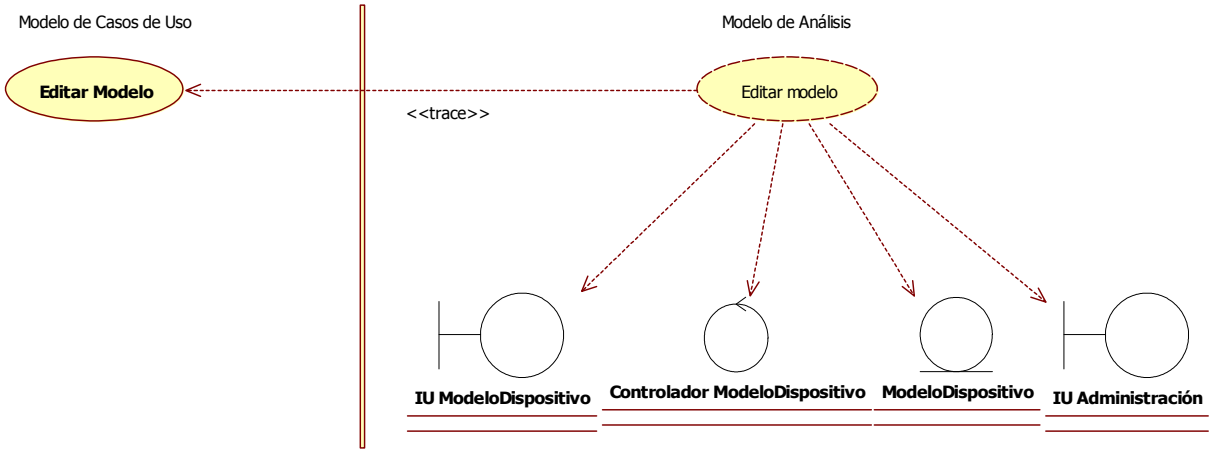


Figura 5.62. Traza (Caso de uso – Análisis) editar modelo.

Cuando un editor quiere modificar un modelo 3D de dispositivo del sistema lo selecciona y esta petición le llega a Controlador ModeloDispositivo. El controlador recupera la información del modelo seleccionado en la entidad ModeloDispositivo, genera un formulario incluyendo los datos recuperados y se le muestra al usuario. El usuario modifica la información necesaria y se envía de nuevo los datos al controlador, que a su vez lo almacena nuevamente en la entidad contenedora.

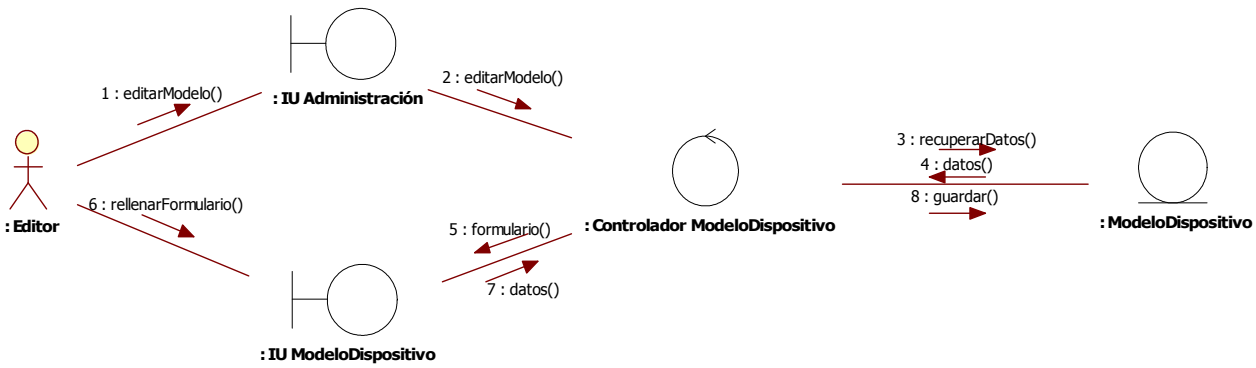


Figura 5.63. Diagrama colaboración editar modelo.

Eliminar modelo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador ModeloDispositivo	IU ModeloDispositivo	ModeloDispositivo

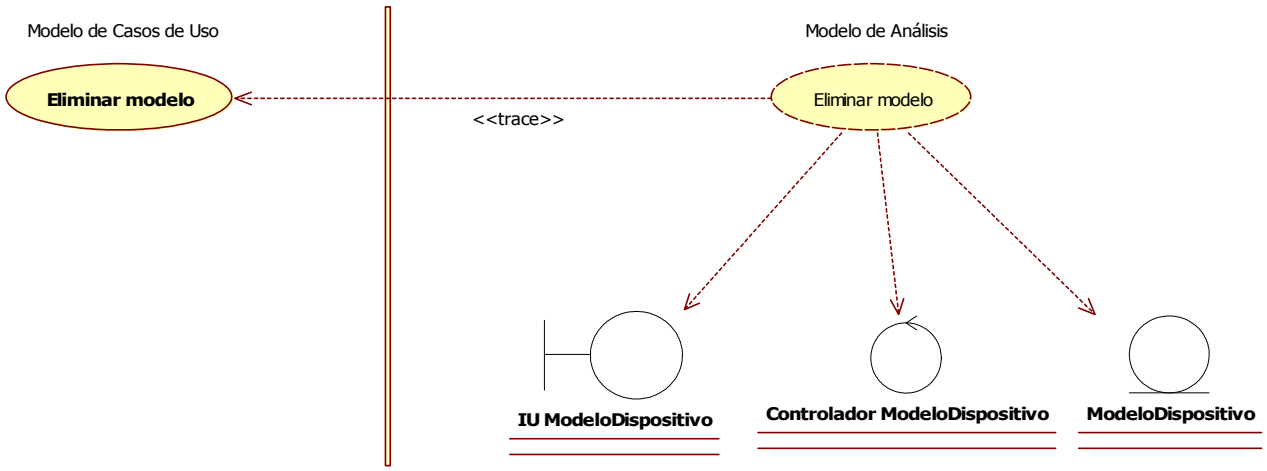


Figura 5.64. Traza (Caso de uso – Análisis) eliminar modelo.

El caso de uso comienza cuando un editor solicita eliminar un Modelo de representación 3D de dispositivo. En ese momento, Controlador ModeloDispositivo accede a la entidad ModeloDispositivo y ejecuta la petición eliminándolo del sistema.

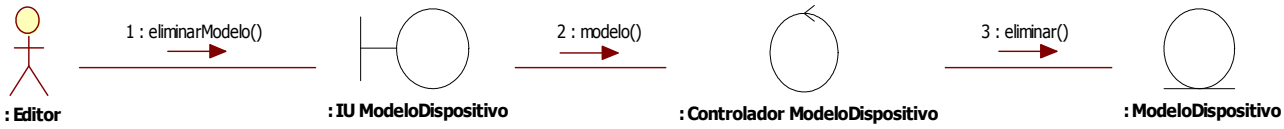


Figura 5.65. Diagrama colaboración eliminar modelo.

Dar de alta a dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo

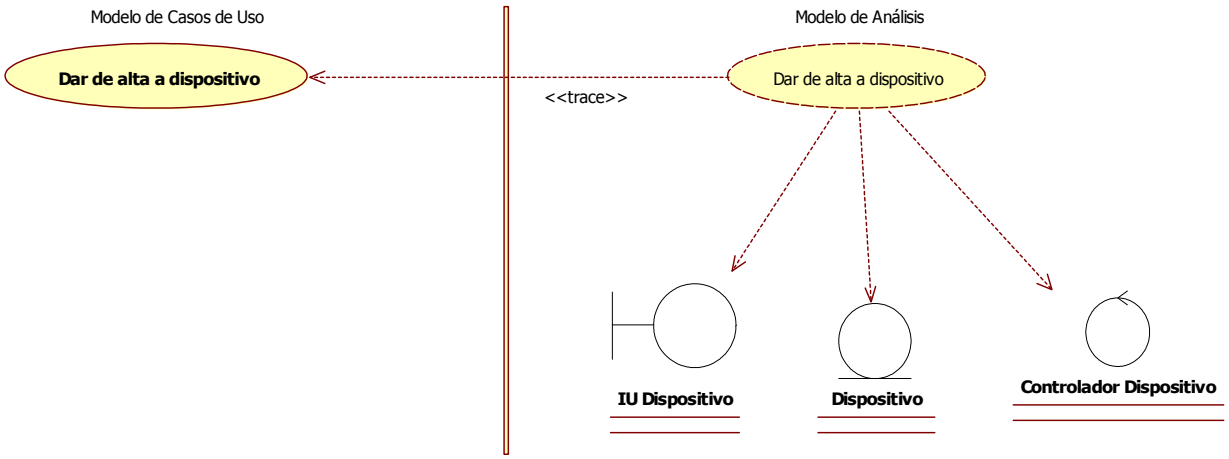


Figura 5.66. Traza (Caso de uso – Análisis) dar de alta dispositivo.

Aunque un dispositivo exista en el sistema puede darse el caso en el que a los editores de la aplicación les interese que no sea visible para los usuarios. Este caso de uso representa el momento en que el editor decide hacer visible un dispositivo que no lo era. Para ello, el editor selecciona un dispositivo y lo marca como activo. En ese momento el controlador accede a la entidad Dispositivo y actualiza esa información para ese dispositivo.

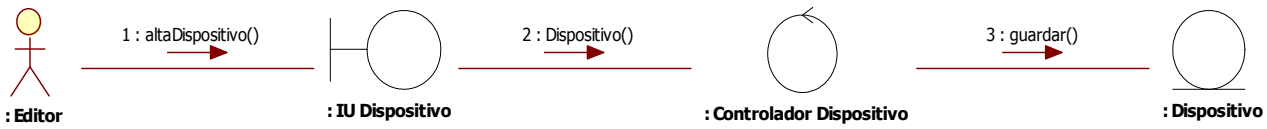


Figura 5.67. Diagrama colaboración dar de alta dispositivo.

Dar de baja a dispositivo

Clases participantes		
Control	**Interfaz**	**Entidad**
Controlador Dispositivo	IU Dispositivo	Dispositivo

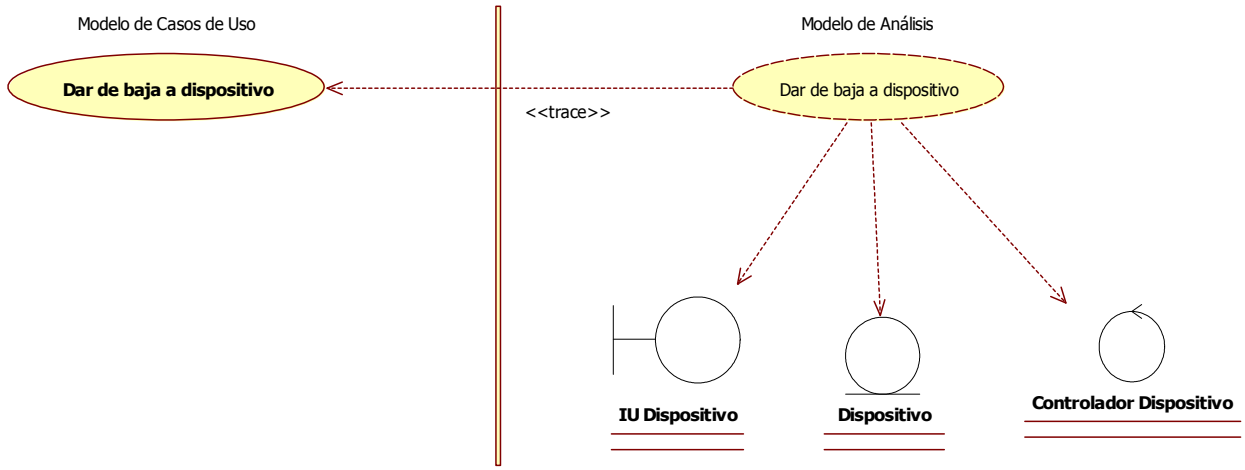


Figura 5.68. Traza (Caso de uso – Análisis) dar de baja dispositivo.

Este es el caso de uso inverso al anterior. En esta ocasión, el caso de uso se inicia cuando un editor decide que un dispositivo del sistema deje de ser visible. Una razón de esto puede ser que el dispositivo se haya extraído de su posición para ser reparado, que el dispositivo se haya roto, se haya perdido la comunicación con él, o simplemente no interesa que sea visible en un momento dado.

Entonces el caso de uso comienza con el editor marcando el dispositivo como inactivo. En ese momento el controlador accede a la entidad Dispositivo y actualiza el valor de estado a inactivo para el dispositivo seleccionado.

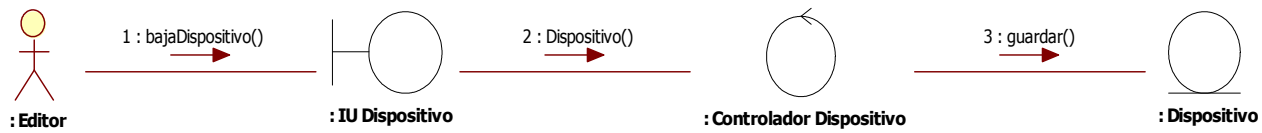


Figura 5.69. Diagrama colaboración dar de baja dispositivo.

5.4. Diseño

A continuación se desarrolla el modelo diseño de proyecto. En este apartado se pasará del modelo de análisis, que proporciona una comprensión detallada de los requisitos, al modelo de diseño, que ofrece una representación coherente y detallada del programa donde se ha tenido en cuenta aspectos técnicos de la construcción del proyecto. En el diseño se requiere de una comprensión profunda de las restricciones ligadas a los lenguajes de programación, librerías o tecnologías de interfaz de usuario. El presente apartado estará divido en los siguientes subapartados:

- **Diseño arquitectónico.** Se definirá la arquitectura del software desde la capa de más bajo nivel a la más específica.
- **Diagramas de clases y diagramas de secuencia.** Se documentará mediante diagramas de clases la composición de cada uno de los módulos a implementar y se documentará cada realización de casos de uso mediante diagramas de secuencia.
- **Diseño de despliegue.** Define el modelo de despliegue de la aplicación dentro de la organización.
- **Diseño de base de datos.** Mediante diagramas de Entidad-Relación se modela la base de datos sobre la que actuará la aplicación.
- **Diseño de prototipo de interfaz de usuario.**

5.4.1. Diseño Arquitectónico.

El diseño arquitectónico de la aplicación se va a basar en una arquitectura por capas, disminuyendo en generalidad según vamos subiendo en las capas. Van a existir cuatro capas bien diferenciadas: la capa del sistema, la capa middleware, la capa genérica y la capa específica.

La capa del sistema es la que se encuentra en el más bajo nivel e incluye operaciones de sistemas operativos y comunicaciones de red. En ella aparece el paquete TCP/IP.

La siguiente capa es la capa Middleware y en ella están incluidos todo tipo de servicios y paquetes de terceros que son necesarios para el correcto funcionamiento de la aplicación. Esta capa está formada por el sistema de gestión de bases de datos, que en nuestro caso será MySQL. También la forma Python como intérprete del lenguaje Python y MySQL-Python para permitir la conexión con la base datos. Como framework de aplicación web se hace uso de Django. En esta capa se incluye también la API de Google Earth y el paquete Scientific.IO.NETCDF para poder conectar la aplicación a servicios NetCDF para la obtención de los datos leídos por los sensores de los dispositivos. Por último se incluye también en este nivel los servidores web y los navegadores.

En las dos capas superiores estarán los propios módulos de esta aplicación. Si subimos un nivel en la arquitectura del sistema nos encontramos con la capa genérica. Esta capa la forman los paquetes mas genéricos de 3DO. Son paquetes que pueden ser reutilizados fácilmente en otra

aplicación, ya que son elementos comunes en muchas aplicaciones. Por lo tanto, en esta capa está contenido el paquete de gestión de usuarios.

Por último, nos encontramos con la capa específica que contiene los paquetes que han sido diseñados específicamente para esta aplicación y que muy difícilmente podrán ser reutilizados en otra aplicación. Estos son los paquetes Cliente y Dispositivos, que se adaptan perfectamente a la estructura de datos que se usa en la organización.

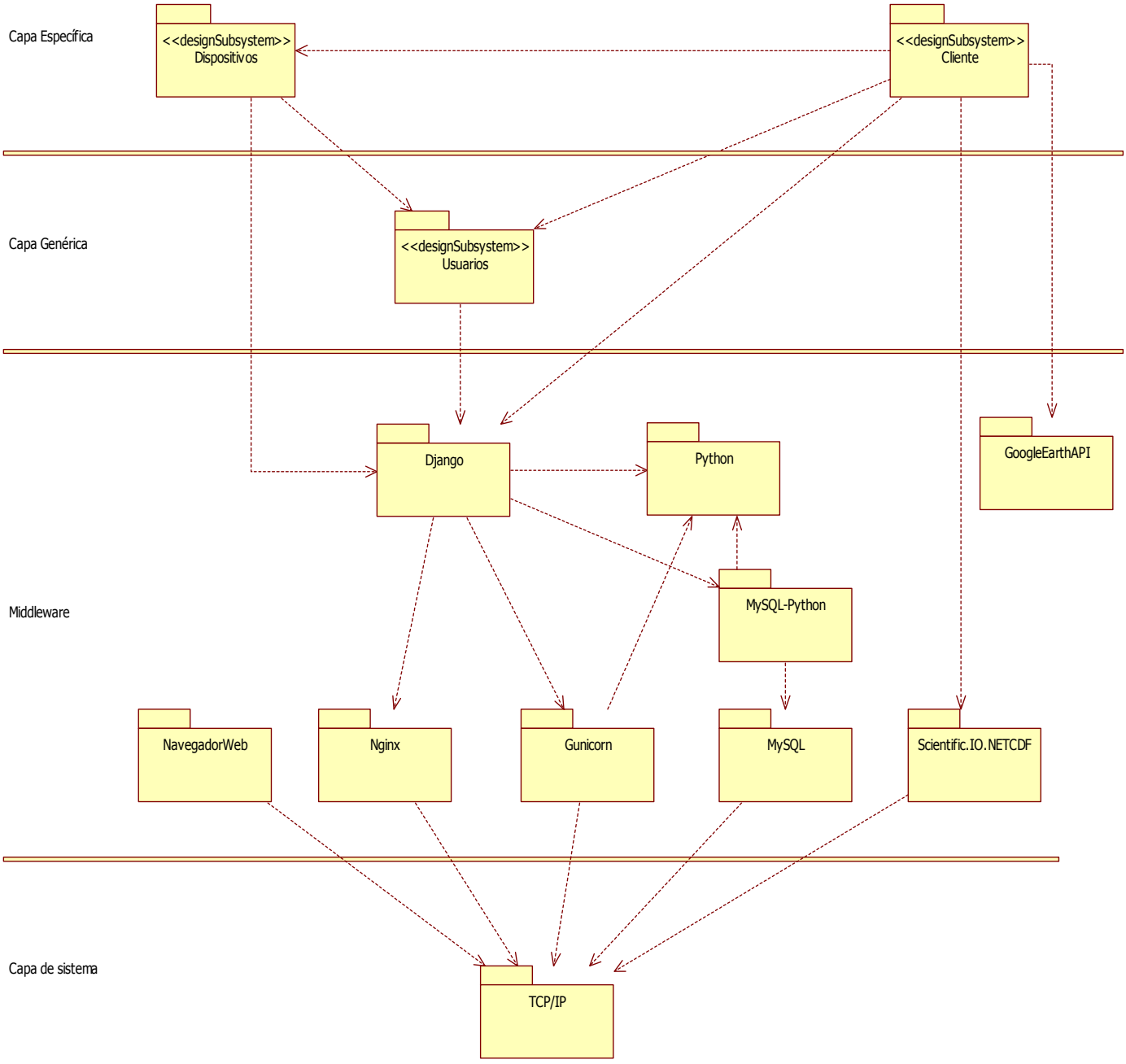


Figura 5.69. Diagrama Diseño Arquitectónico.

5.4.2. Diagramas de Clases y Secuencias

En este subapartado se definirá el funcionamiento de las distintas partes de la aplicación haciendo uso de diagramas de clases y diagramas de secuencias.

Con los diagramas de clase se pretende documentar las relaciones entre las distintas clases que conforman el sistema. Por otro lado, con los diagramas de secuencia se consigue definir el intercambio de mensajes e información entre las clases.

Siguiendo la metodología usada hasta el momento, por facilidad se irán agrupando las realizaciones de caso de uso por paquetes funcionales, tal y como se ha venido haciendo a lo largo del presente documento.

5.4.2.1. Cliente

El paquete de diseño Cliente tiene una relación de traza con el paquete de análisis con el mismo nombre, y éste a su vez con el paquete de casos de uso.

Al igual que toda la aplicación, se hace uso del patrón de diseño MVC (modelo – vista – controlador). Con ello se consigue separar el negocio de la aplicación, de la interfaz de usuario y del modelo de datos. En base a ésto, los controladores se encargan de la parte funcional, mientras que los modelos son los encargados de interactuar con la base de datos.

La página principal del cliente está compuesta por una serie de vistas que incluye un mapa 3D o globo terráqueo virtual, un menú y una sección para las sesiones de usuario.

El controlador del mapa 3D hereda directamente de la API de Google Earth. Además éste está relacionado con los controladores de los usuarios y los dispositivos para obtener la información desde la base de datos a través de los modelos respectivos, sobre todo a nivel de permisos.

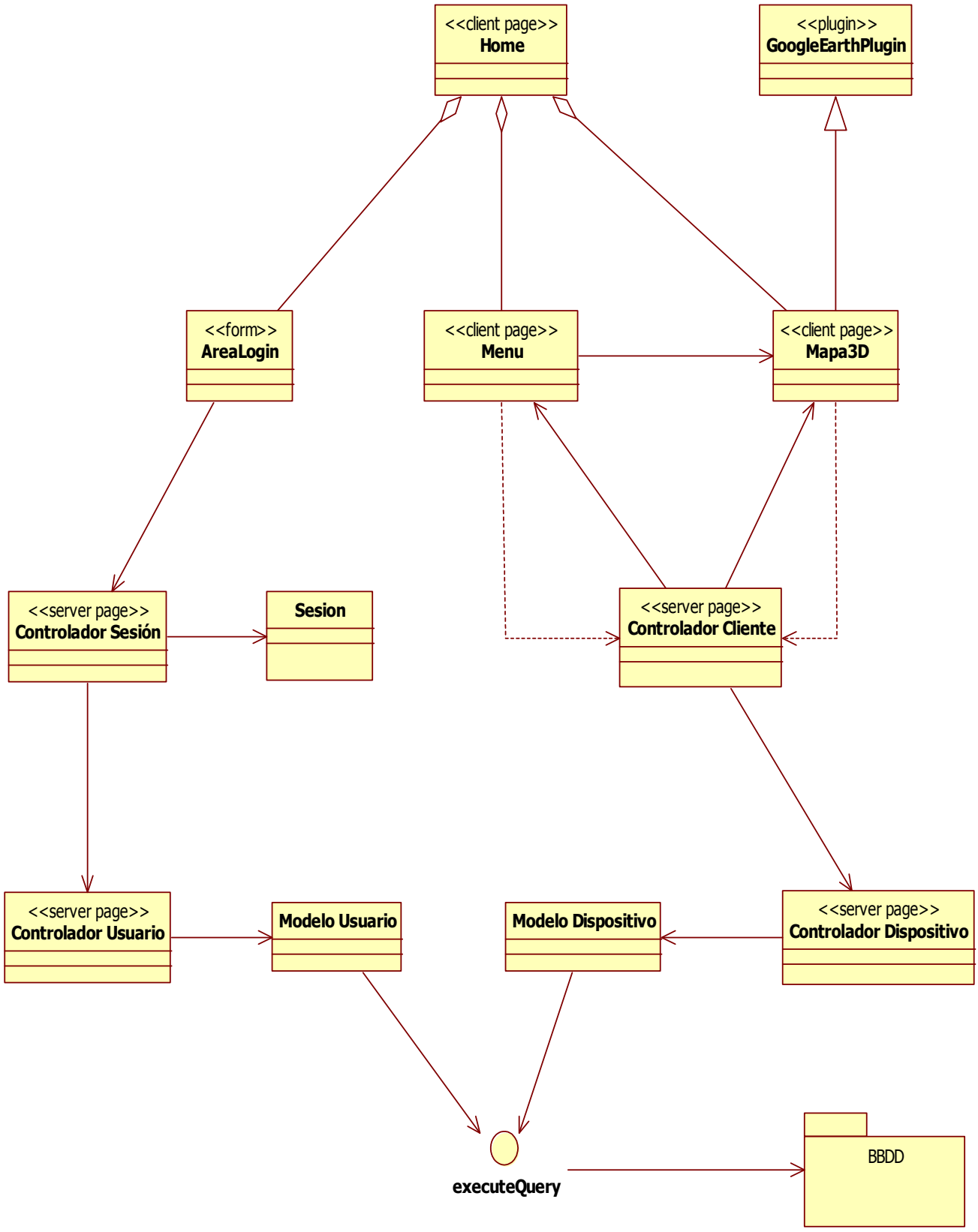


Figura 5.70. Diagrama de clases del cliente.

Ver mapa 3D

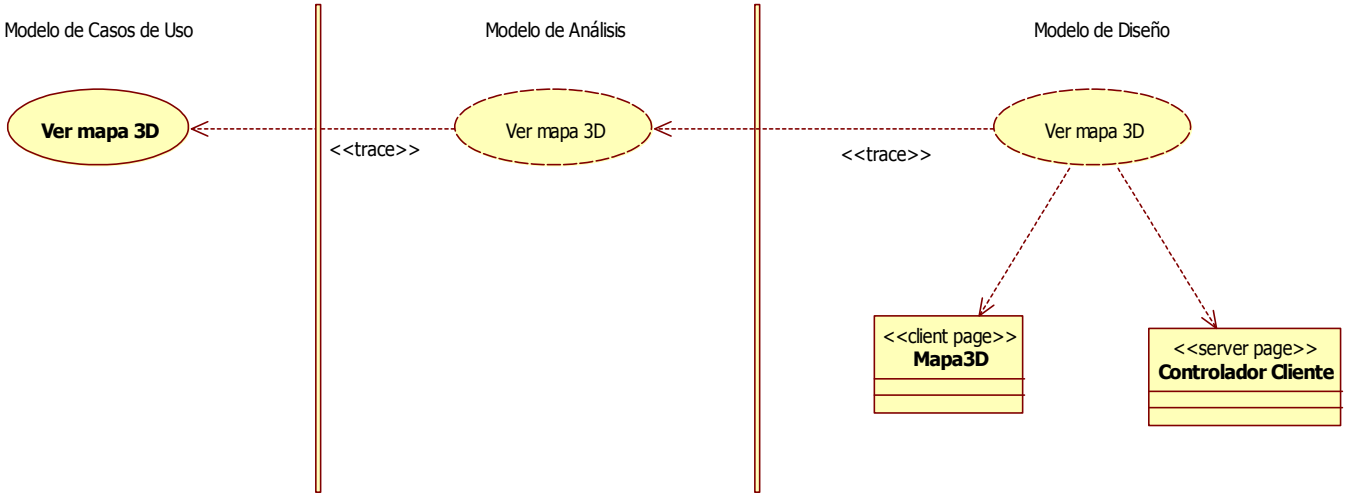


Figura 5.71. Traza (Caso de uso - Análisis - Diseño) ver mapa 3D.

Cuando un usuario accede al cliente de la aplicación, se llama al controlador del cliente que crea un instancia de globo terráqueo, y genera la vista para mostrarlo en una posición inicial.

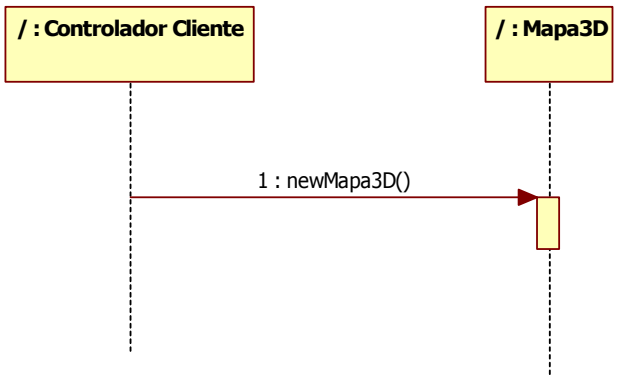


Figura 5.72. Diagrama de secuencia ver mapa 3D.

Ver mundo submarino en mapa 3D

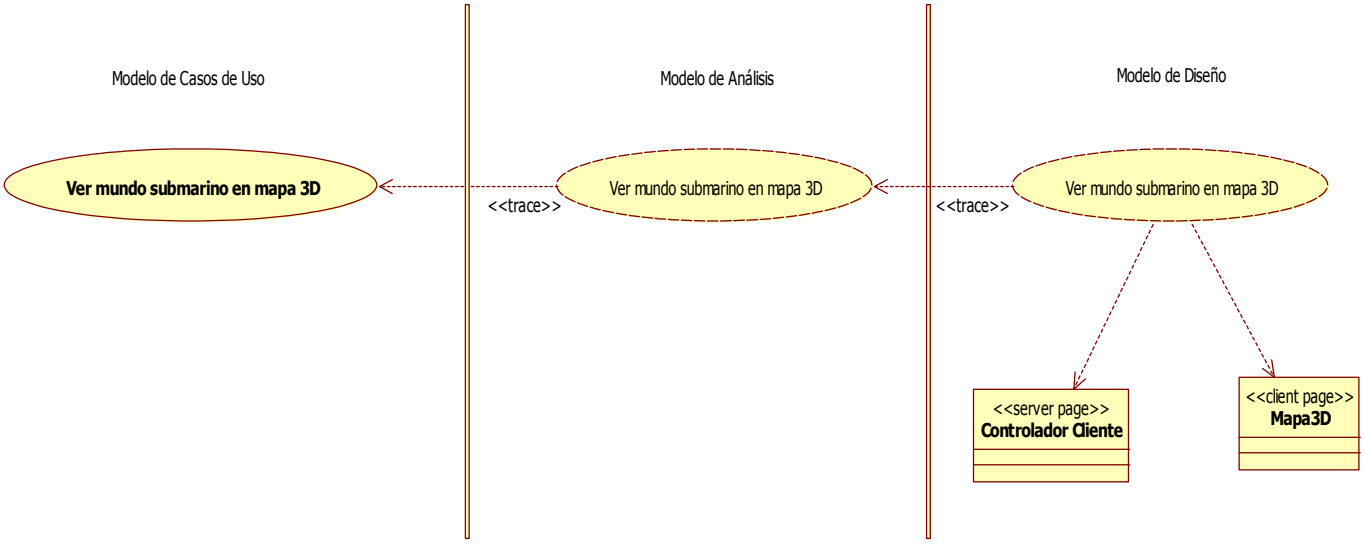


Figura 5.73. Traza (Caso de uso - Análisis - Diseño) ver mundo submarino en mapa 3D.

Cuando ya existe una instancia del mapa 3D y el controlador recibe una petición de ir a una posición submarina, el controlador genera el mapa en dicha posición y renderiza la vista de dicho mapa.

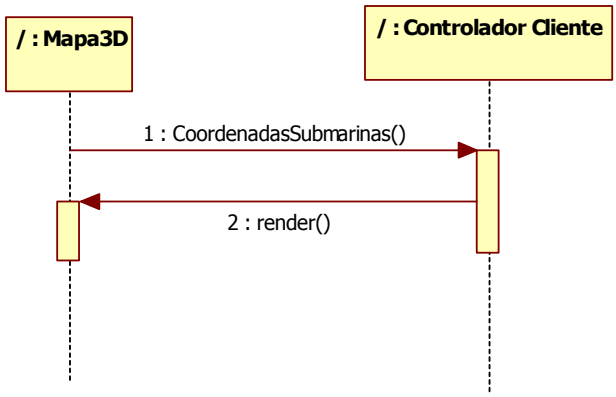


Figura 5.74. Diagrama de secuencia ver mundo submarino mapa 3D.

Mover mapa 3D

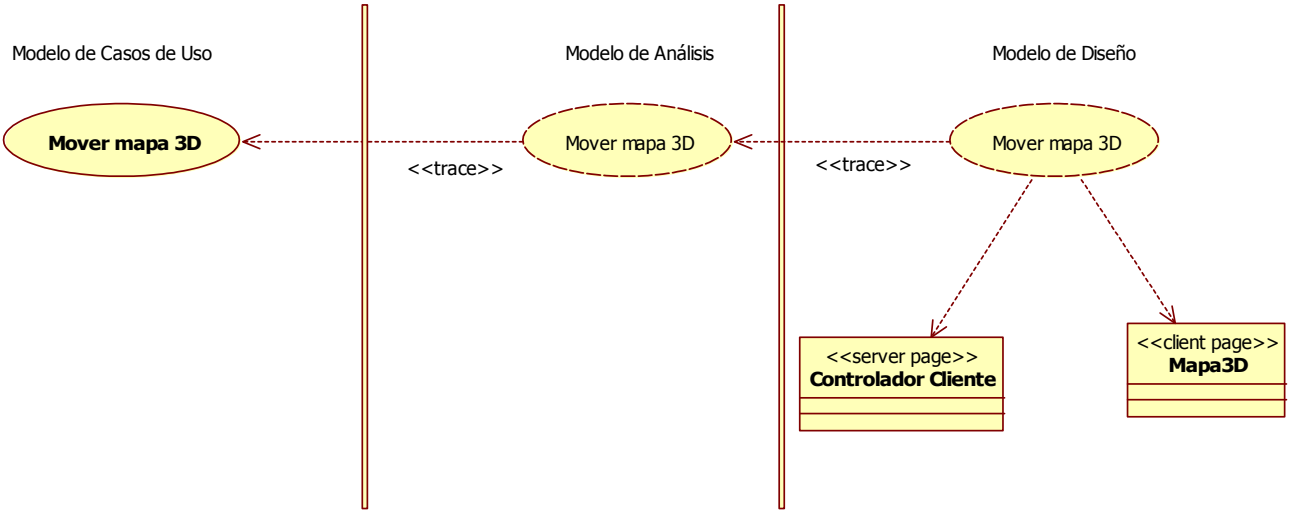


Figura 5.75. Traza (Caso de uso - Análisis - Diseño) mover mapa 3D.

El controlador recibe una petición de cambio de posición en el mapa 3D. Con ello también recibe las nuevas coordenadas a las que debe dirigirse. El controlador trata la petición, genera la nueva vista del globo y renderiza la vista para que sea mostrada en el cliente.

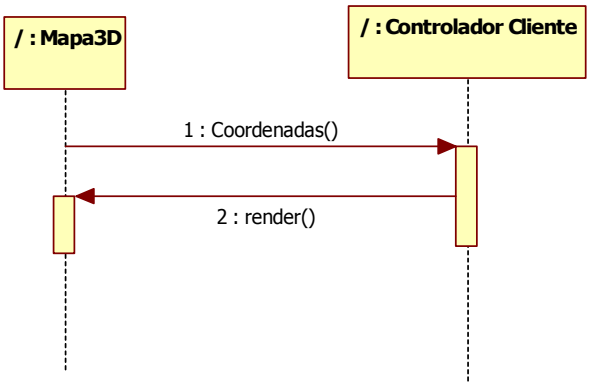


Figura 5.76. Diagrama de secuencia mover mapa 3D.

Hacer zoom en el mapa 3D

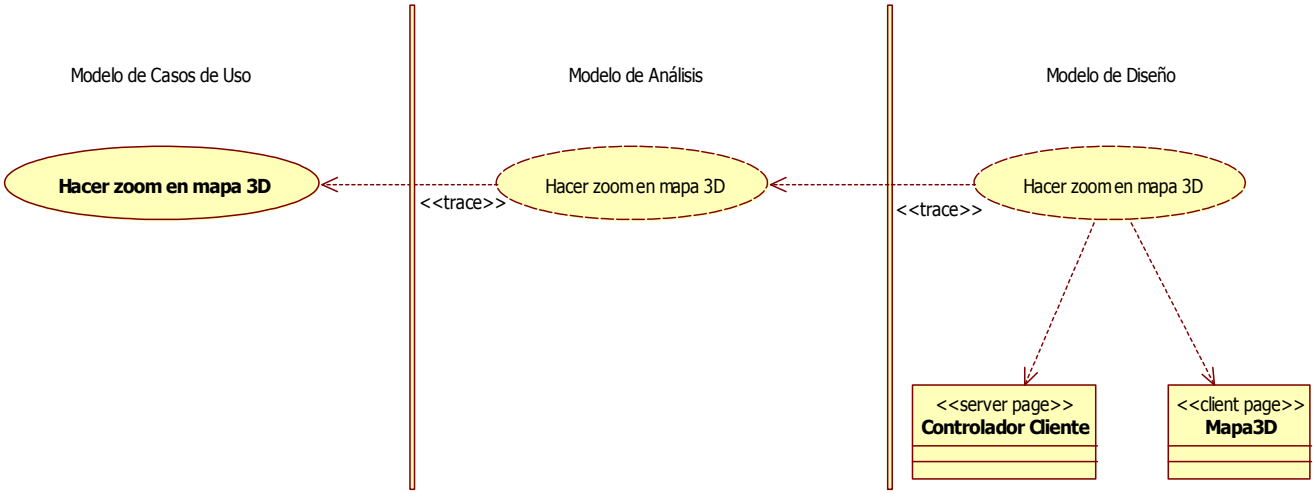


Figura 5.77. Traza (Caso de uso - Análisis - Diseño) zoom en mapa 3D.

Además de movimientos a lo largo del mapa, el cliente también permite el acercar o alejar la vista actual. Para ello el usuario aumenta o disminuye el zoom. Ésto es capturado por el controlador que recibe el número de aumentos o disminuciones. Procesa una nueva vista del mapa y lo renderiza en el cliente.

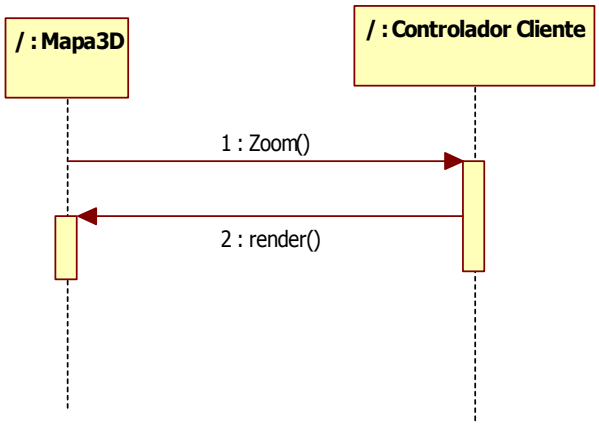


Figura 5.78. Diagrama de secuencia zoom mapa 3D.

Ver dispositivos en el mapa 3D

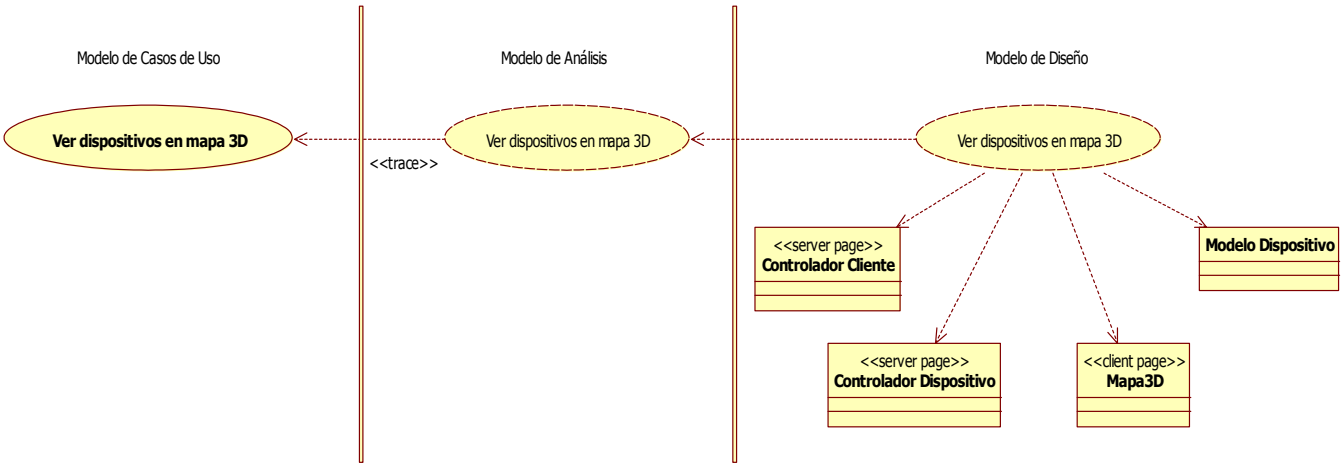


Figura 5.79. Traza (Caso de uso - Análisis - Diseño) ver dispositivos en mapa 3D.

Cuando se carga el mapa 3D, Controlador Cliente solicita al Controlador Dispositivo la lista de los dispositivos que están situados en el rango de coordenadas visible. El controlador de los dispositivos los obtiene por medio de su modelo que hace la consulta a la base de datos y éste es devuelto al controlador del cliente. Con esa información, el controlador genera la vista del mapa 3D incluyendo los dispositivos y la muestra.

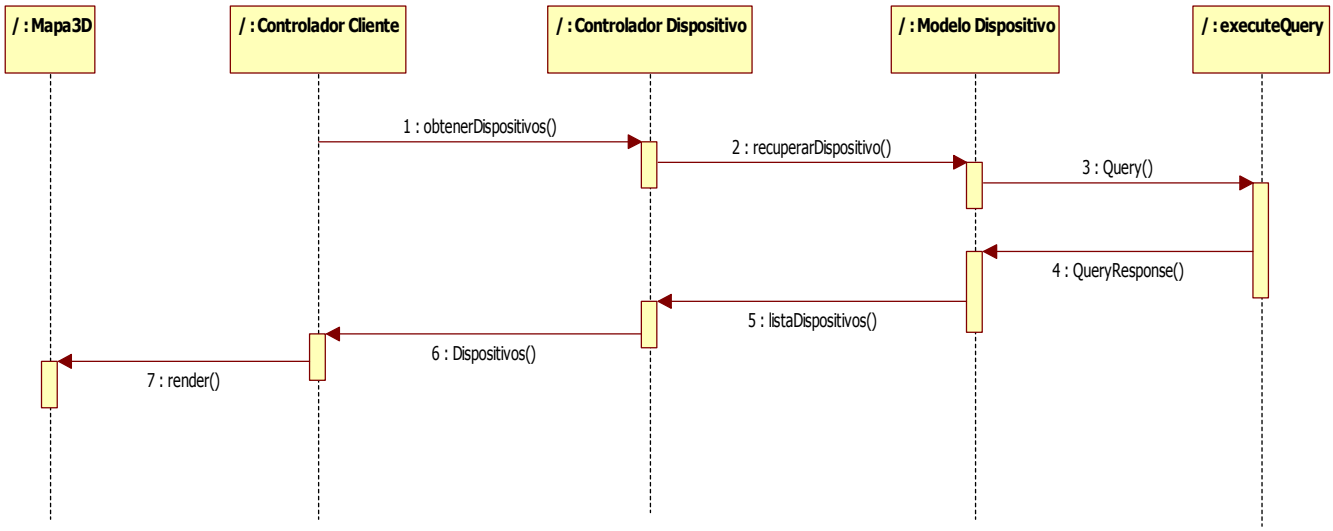


Figura 5.80. Diagrama de secuencia ver dispositivos mapa 3D.

Ver dispositivos en el menú

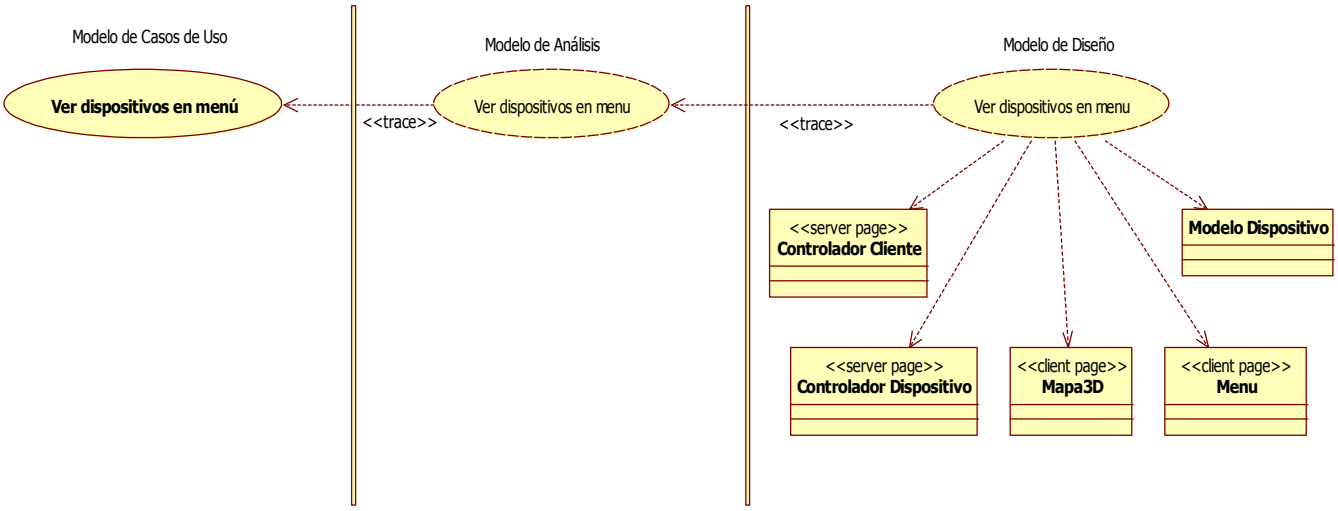


Figura 5.81. Traza (Caso de uso - Análisis - Diseño) ver dispositivos en menú.

Para la creación del menú desplegable, el controlador del cliente solicita al controlador de dispositivos la lista de todos los dispositivos activos en el sistema. Este último la obtiene a través de su modelo que ejecuta la consulta de la base de datos. Con la lista completa el controlador del cliente genera el menú desplegable.

Cuando un usuario hace click sobre un ítem del menú, llega al controlador la petición y genera un vista del mapa 3D donde aparece el dispositivo seleccionado colocado en su posición.

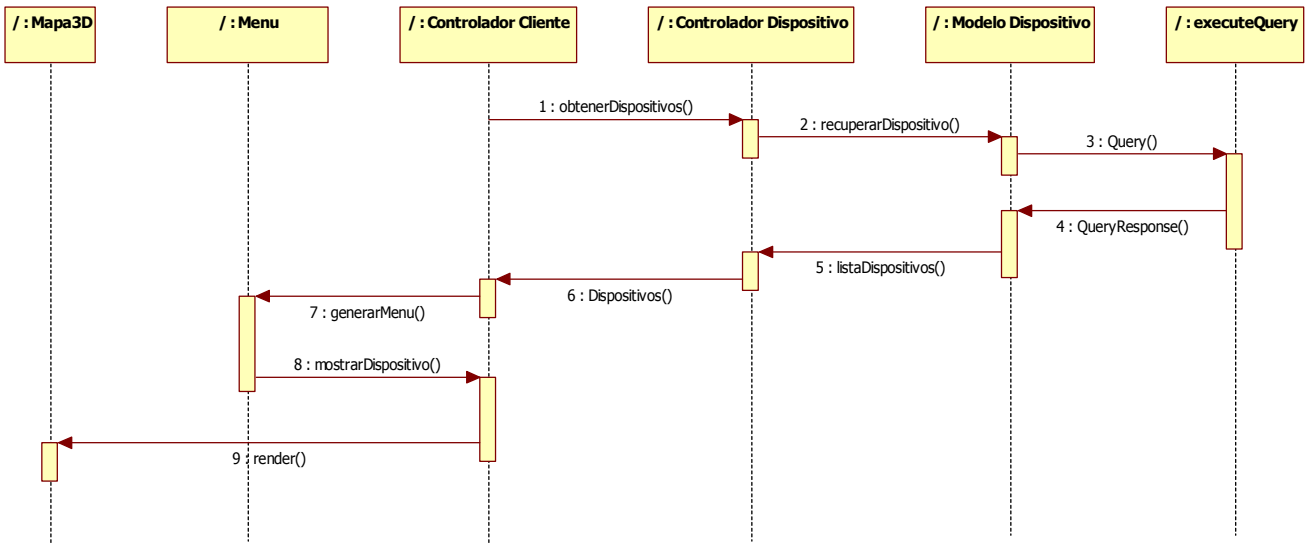


Figura 5.82. Diagrama de secuencia ver dispositivos en menú.

Ver información del dispositivo

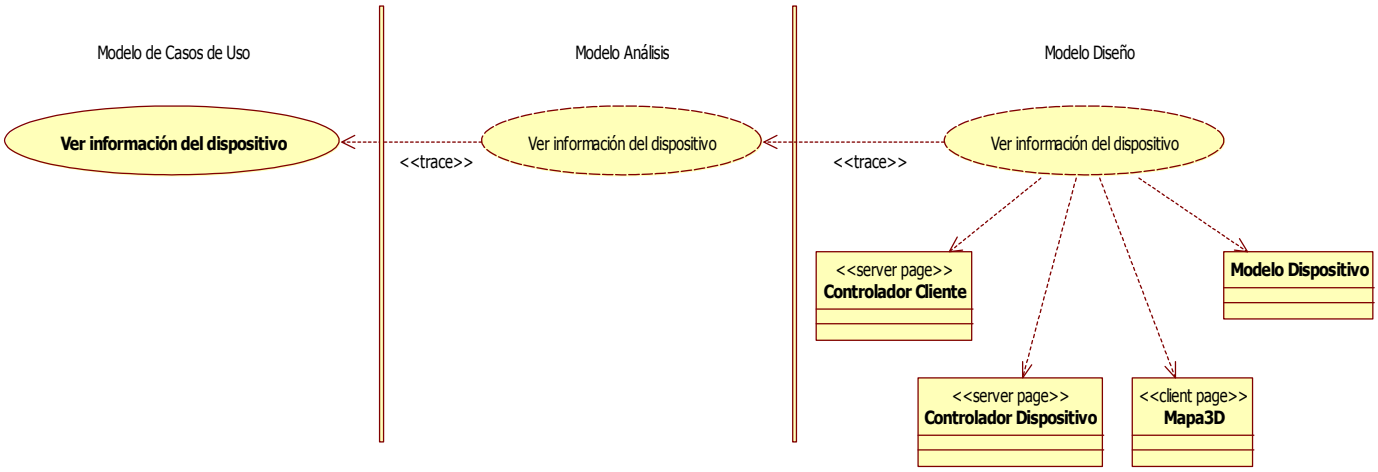


Figura 5.83. Traza (Caso de uso - Análisis - Diseño) ver información de dispositivo.

El controlador del cliente le solicita al controlador de dispositivos la información relativa al dispositivo seleccionado. Controlador Dispositivo la obtiene de la base de datos a través del modelo y se la envía a Controlador Cliente.

Cuando el controlador del cliente tiene la información, la formatea y la incluye en una burbuja sobre el mapa 3D que será mostrada al usuario.

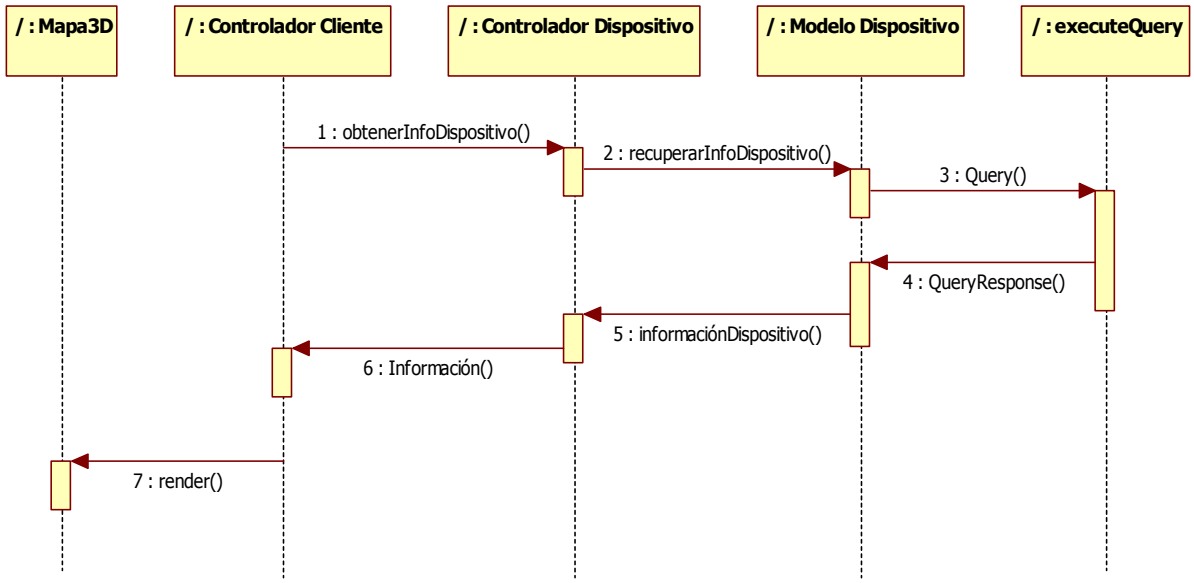


Figura 5.84. Diagrama de secuencia ver información de dispositivo.

Ver lectura de datos del dispositivo

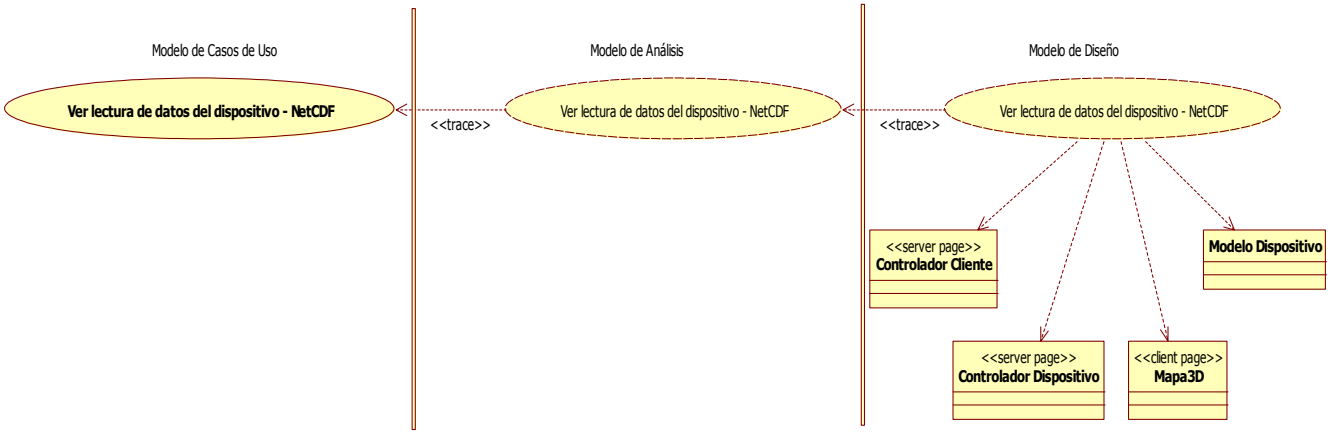


Figura 5.85. Traza (Caso de uso - Análisis - Diseño) ver datos de dispositivo.

Un caso particular del anterior es, una vez mostrada la información, el usuario solicita ver los datos capturados por los sensores del dispositivo. En ese momento el controlador de cliente recibe la petición, se la traslada al controlador de dispositivos y éste la obtiene de la base de datos usando a Modelo Dispositivo.

Al igual que en el caso anterior, Controlador Cliente recibe los datos, los formatea y los muestra en la misma burbuja sobre mapa 3D.

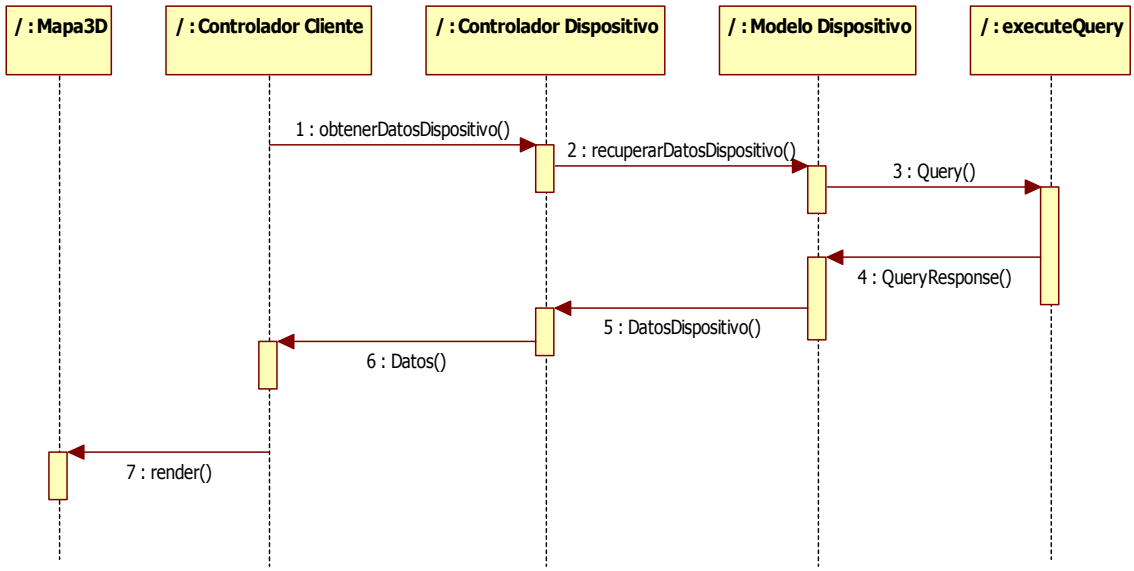


Figura 5.86. Diagrama de secuencia ver datos de dispositivo.

Ver descripción de sensores - SensorML

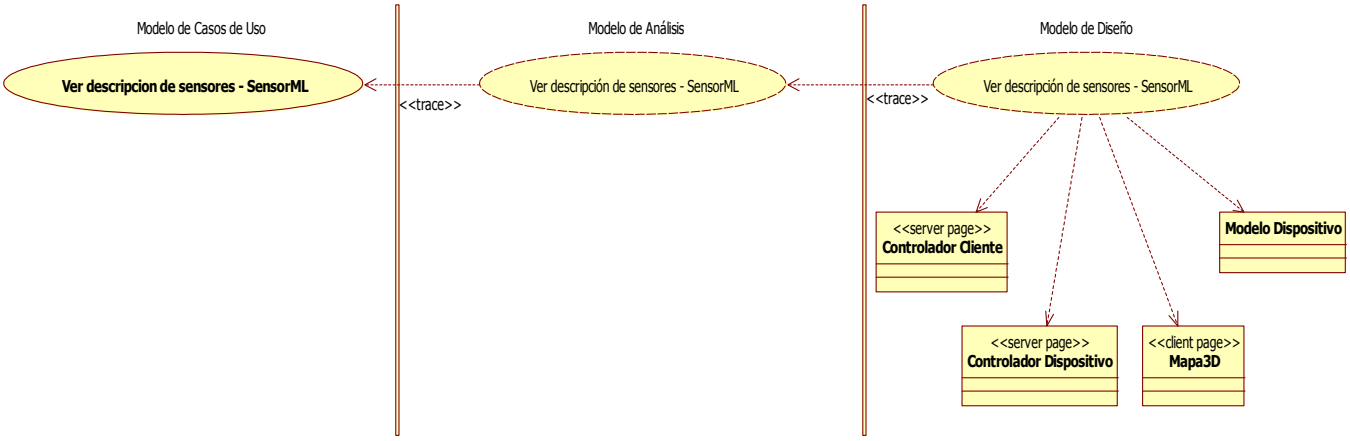


Figura 5.87. Traza (Caso de uso - Análisis - Diseño) ver sensores de dispositivo.

Un caso particular del anterior es, una vez mostrada la información, el usuario solicita ver los datos capturados por los sensores del dispositivo. En ese momento el controlador de cliente recibe la petición, se la traslada al controlador de dispositivos y éste la obtiene de la base de datos usando a Modelo Dispositivo.

Al igual que en el caso anterior, Controlador Cliente recibe los datos, los formatea y los muestra en la misma burbuja sobre mapa 3D.

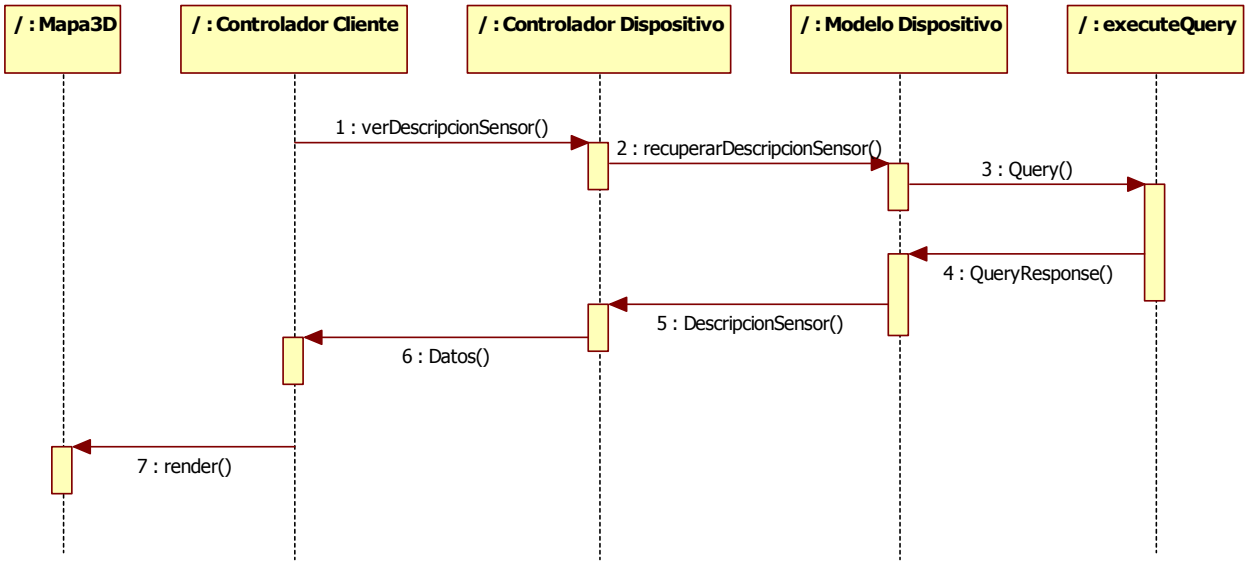


Figura 5.88. Diagrama de secuencia ver sensores de dispositivo.

5.4.2.2. Usuarios

El paquete de diseño Usuarios tiene una relación de traza con el paquete de análisis con el mismo nombre, y éste a su vez con el paquete de casos de uso.

Este paquete pertenece a los paquetes de administración, los cuales tienen una vista común llamada Vista Administración, que da acceso a todas las funciones administrativas del sistema.

El paquete Usuarios implementa todas las funcionalidades relacionadas con la gestión de cuentas de usuario y el control de sesiones.

El diseño también sigue el patrón MVC en el que en el siguiente diagrama de clases queda perfectamente reflejado.

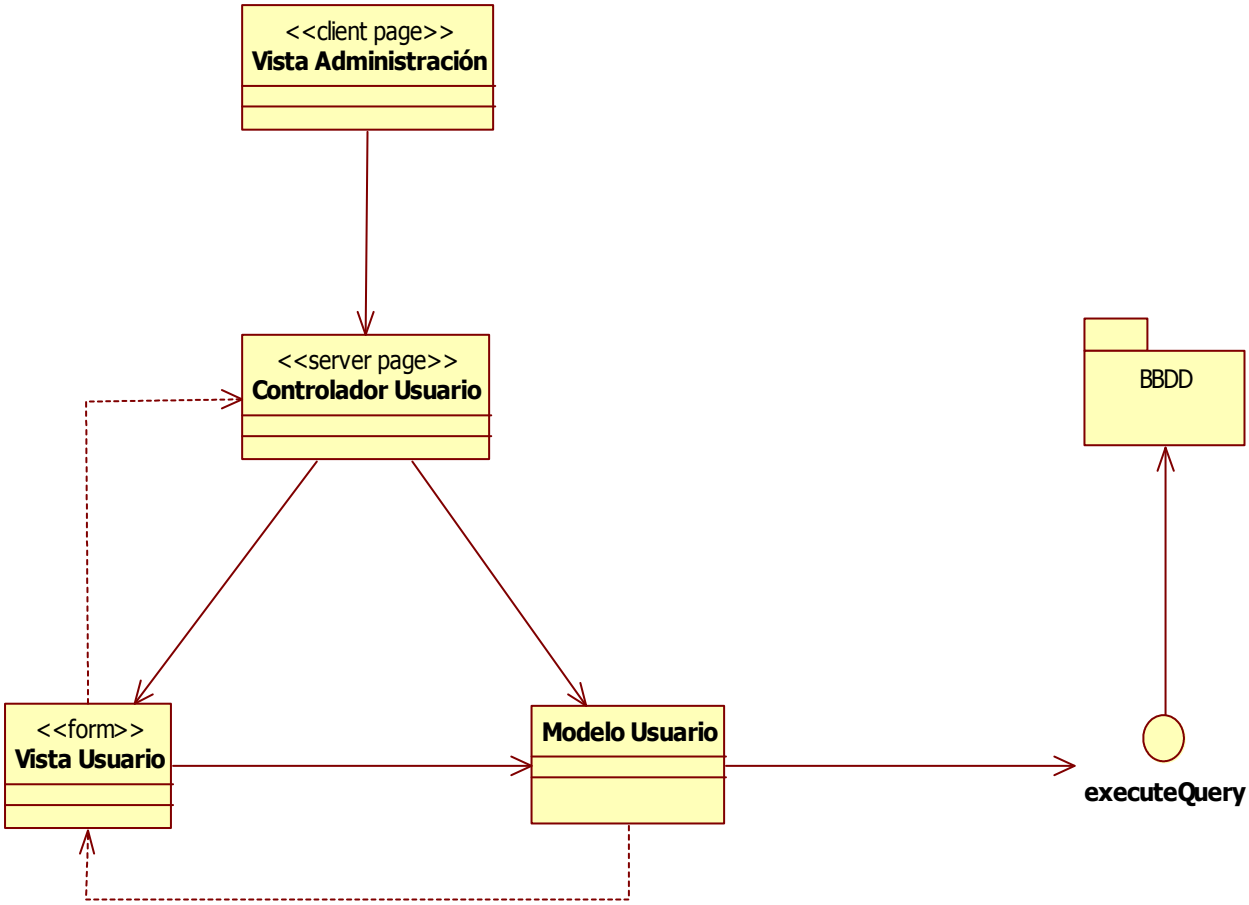


Figura 5.89. Diagrama de clases de usuarios.

Crear usuario

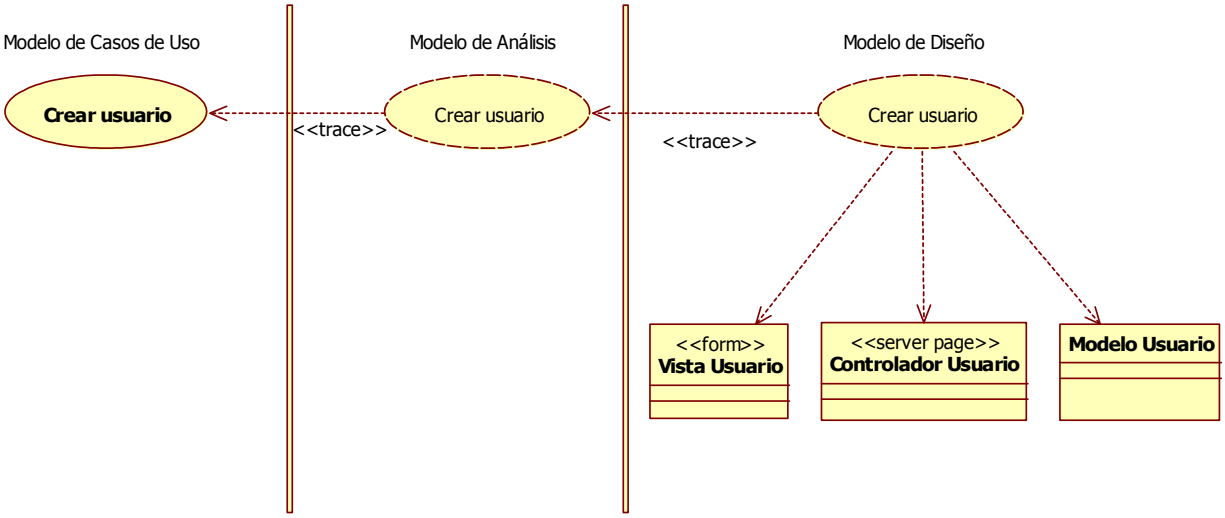


Figura 5.90. Traza (Caso de uso - Análisis - Diseño) crear usuario.

El controlador genera el formulario para la introducción de los datos de usuario. Una vez, recibe los datos se los pasa al modelo para que valide dichos datos, y si todo es correcto lo almacene en la base de datos.

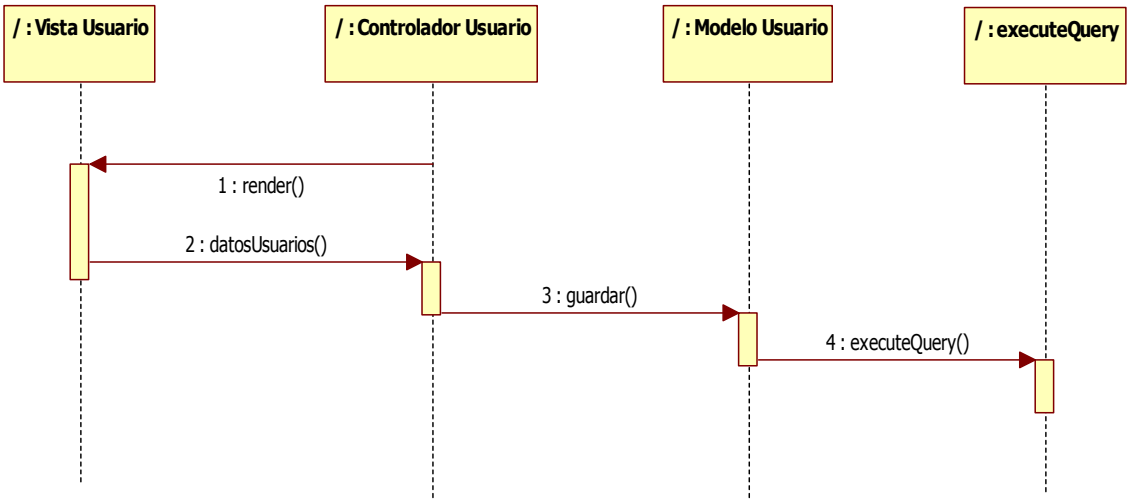


Figura 5.91. Diagrama de secuencia crear usuario.

Editar usuario

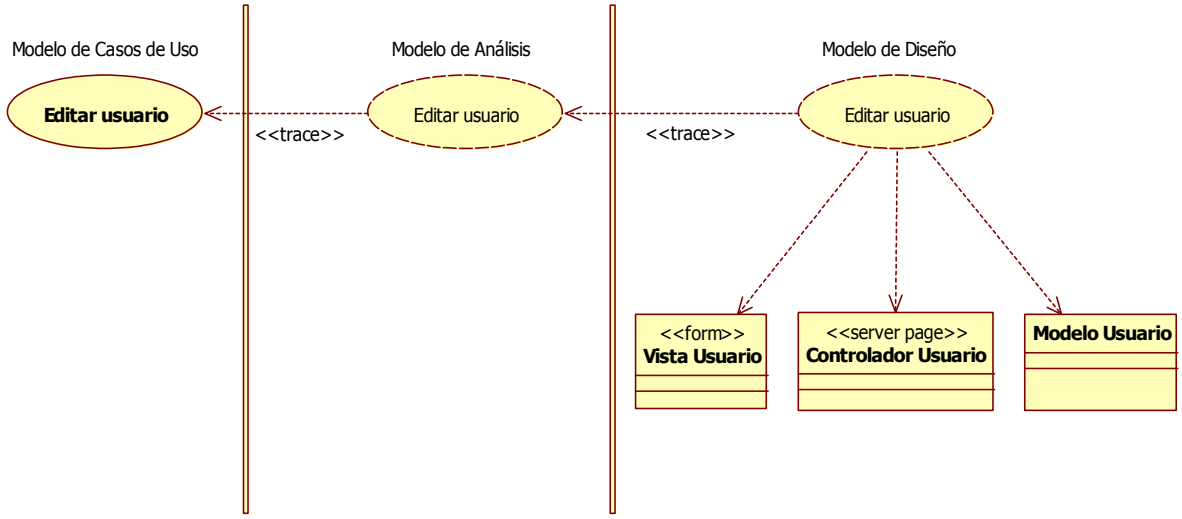


Figura 5.92. Traza (Caso de uso - Análisis - Diseño) editar usuario.

La edición de usuarios es muy parecida a la creación de los mismos. La diferencia es que en la edición el controlador lo primero que hace es obtener los datos del usuario desde la base de datos a través del modelo y generar la vista a partir de ellos.

Una vez se modifican los datos y van de vuelta el controlador, este los envía al modelo para que los valide y los almacene en la base de datos.

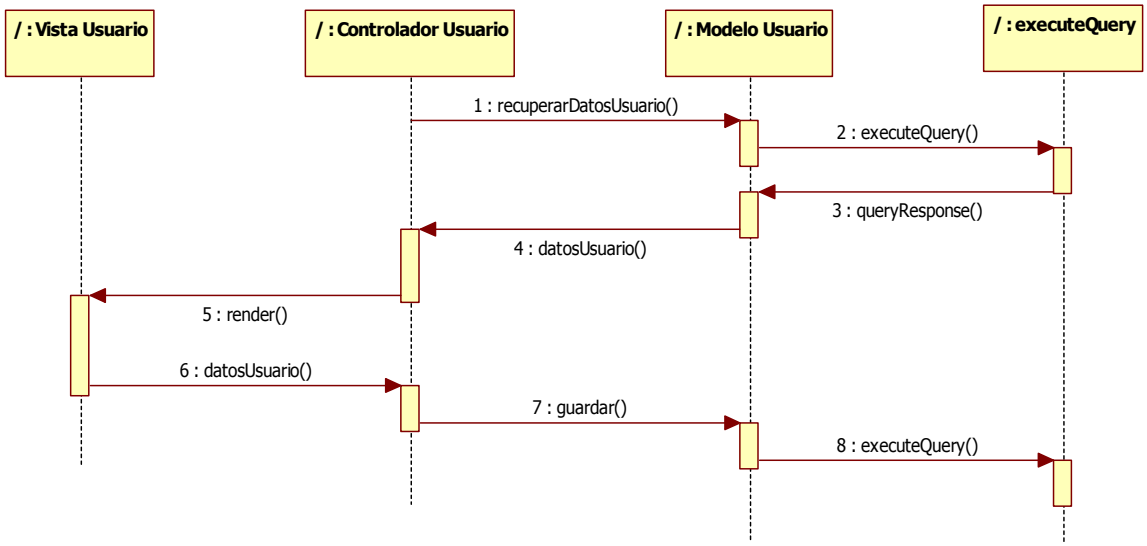


Figura 5.93. Diagrama de secuencia editar usuario.

Eliminar usuario

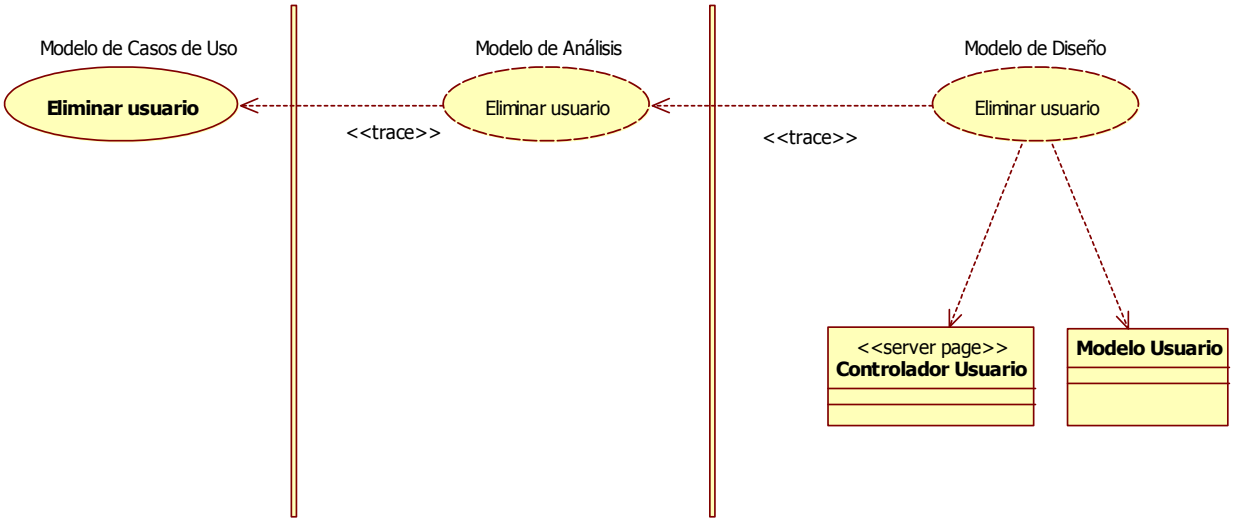


Figura 5.94. Traza (Caso de uso - Análisis - Diseño) eliminar usuario.

Cuando al controlador le llega una petición de eliminar una cuenta de usuario, éste ordena al modelo que lo elimine. Modelo Usuario accede a la base de datos y elimina el usuario con el identificador dado.

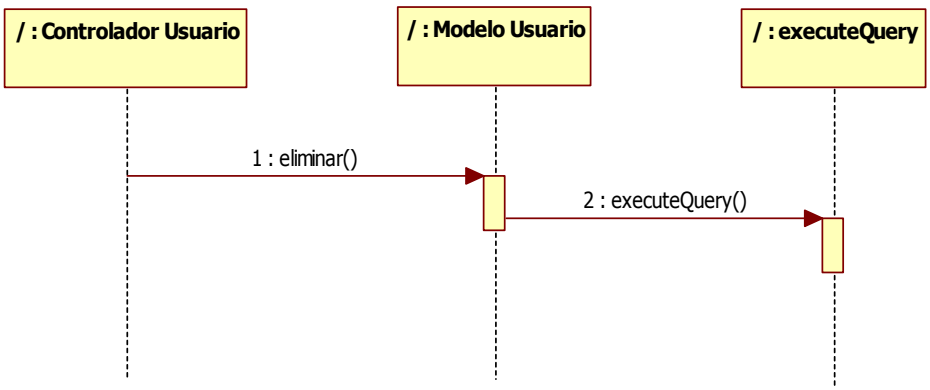


Figura 5.95. Diagrama de secuencia eliminar usuario.

Controlar acceso a usuarios anónimos

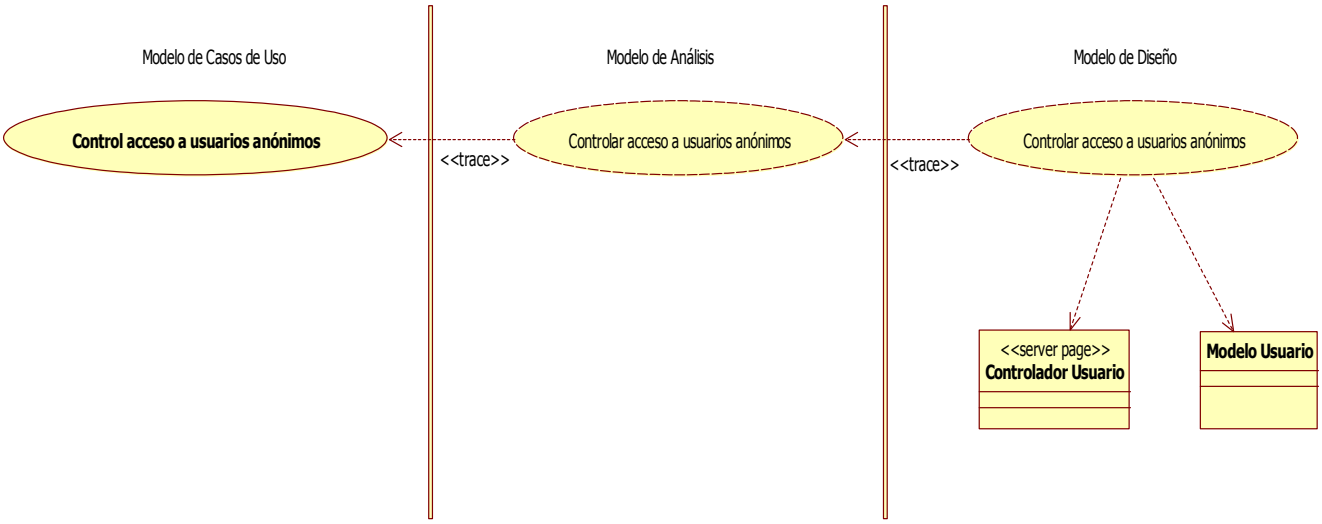


Figura 5.96. Traza (Caso de uso - Análisis - Diseño) acceso anónimos.

Cuando el administrador modifica el grado de privacidad de la aplicación, el controlador de usuarios actualiza el valor en la base de datos a través del modelo.

El controlador del cliente cuando se llama por primera vez comprueba este valor, y si el estado es cerrado, solo permite cargar el globo terráqueo a usuarios que se encuentren identificados en el sistema.

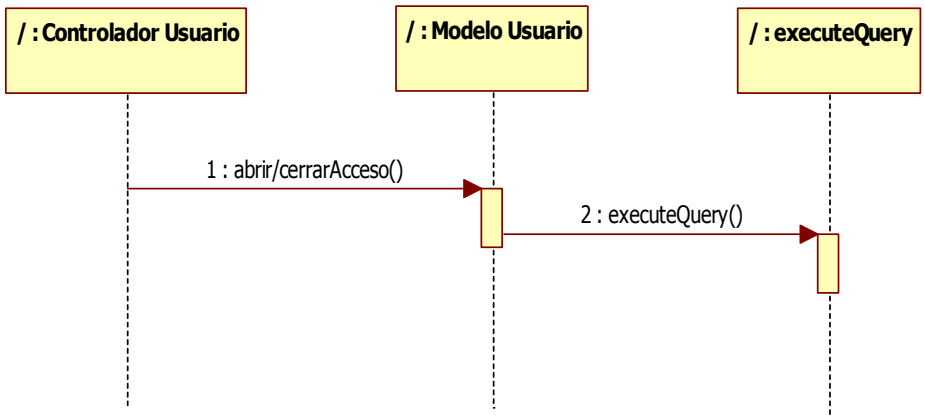


Figura 5.97. Diagrama de secuencia acceso anónimos.

Iniciar sesión

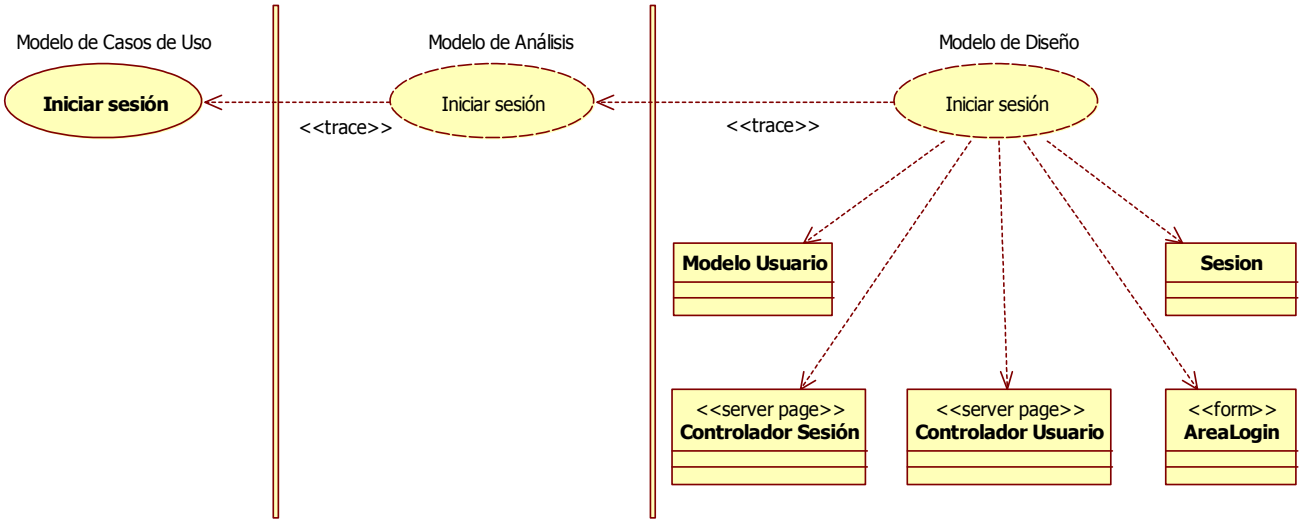


Figura 5.98. Traza (Caso de uso - Análisis - Diseño) iniciar sesión.

El usuario introduce los credenciales de acceso a través del formulario AreaLogin. Estos datos le llegan al Controlador Sesion que necesita la contraseña cifrada del usuario que intenta identificarse para cotejarla.

Para ello, se la solicita al controlador de usuarios que la obtiene de la base de datos a través del modelo y se la envía al controlador de sesiones. Éste comprueba que la contraseña introducida tras cifrarla y la contraseña cifrada almacenada en la base de datos son la misma. Si es así crea una sesión de usuario.

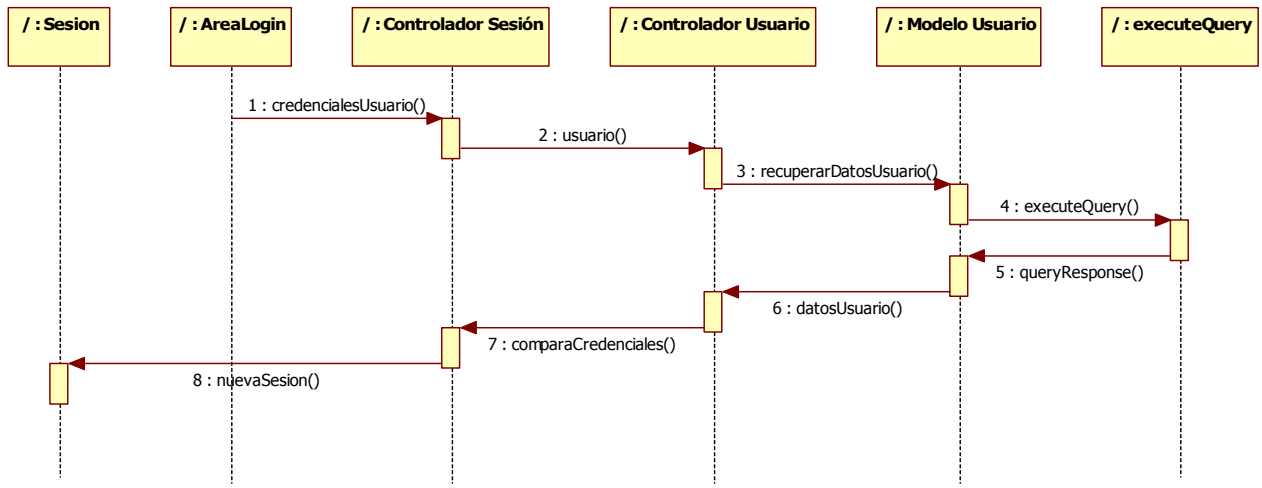


Figura 5.99. Diagrama de secuencia iniciar sesión.

Cerrar sesión

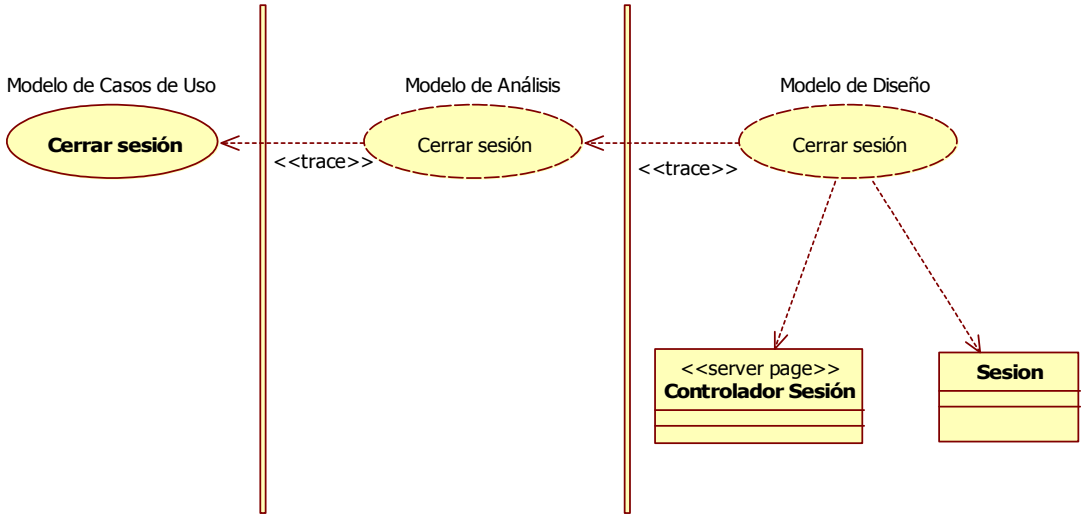


Figura 5.100. Traza (Caso de uso - Análisis - Diseño) cerrar sesión.

El proceso para cerrar una sesión de usuario abierta es bastante más simple. Basta con que el usuario se lo manifieste al controlador de sesión y que destruya el objeto sesión.

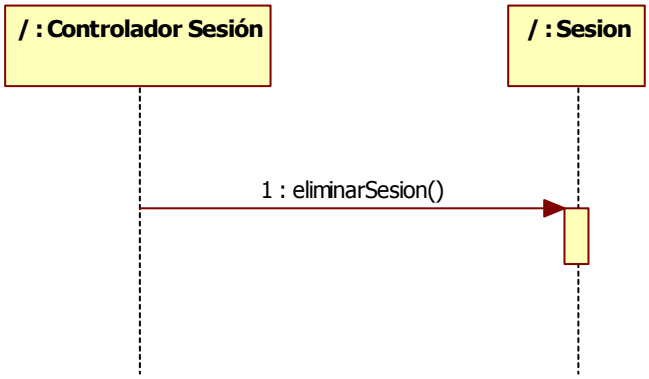


Figura 5.101. Diagrama de secuencia cerrar sesión.

Cambiar contraseña

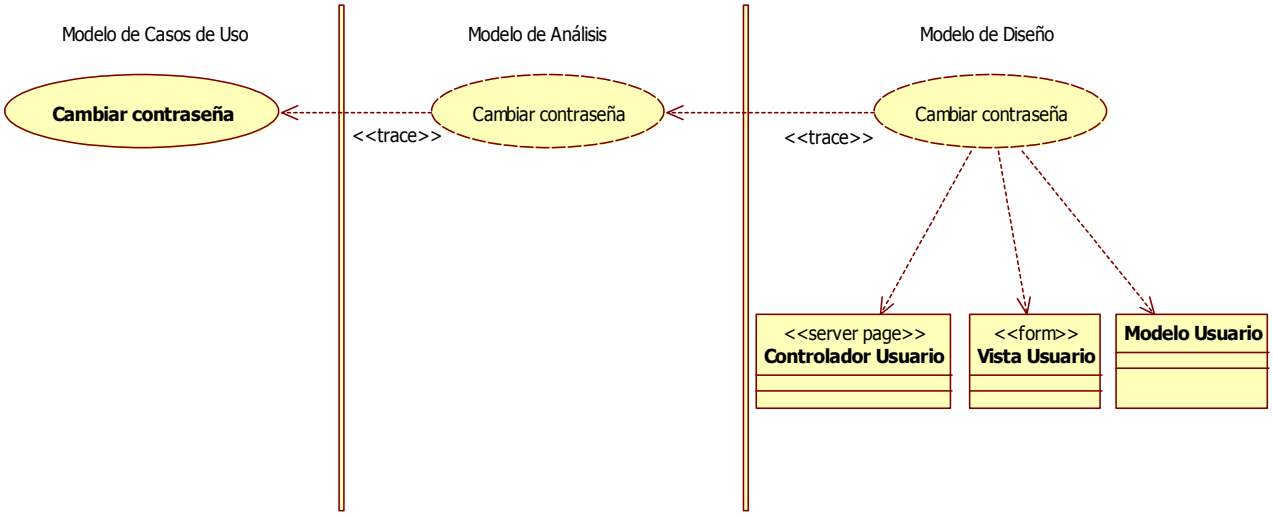


Figura 5.102. Traza (Caso de uso - Análisis - Diseño) cambiar contraseña.

Cuando un usuario solicita modificar su contraseña, Controlador usuario genera un formulario que el usuario rellenará con su contraseña antigua y su contraseña nueva por duplicado.

El controlador recibe los datos y se lo hace llegar al modelo, que los valida y los almacena en la base de datos.

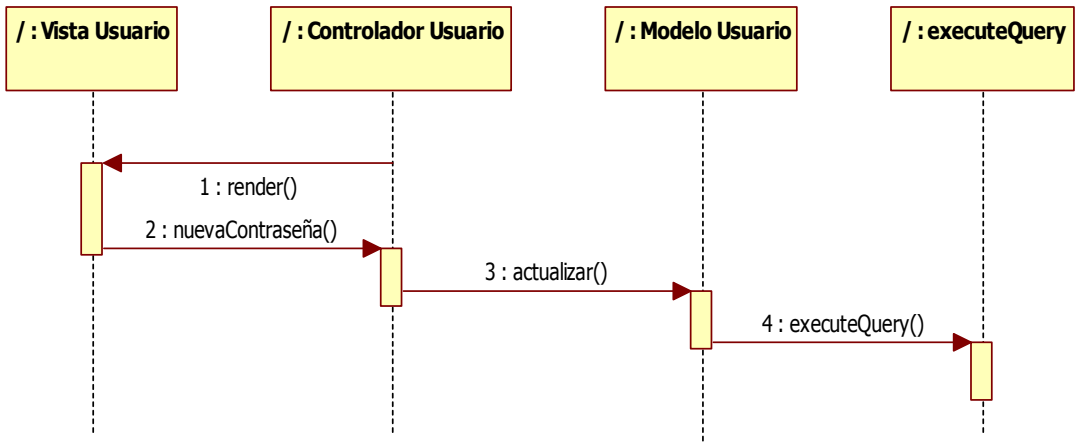


Figura 5.103. Diagrama de secuencia cambiar contraseña.

5.4.2.3. Dispositivos

El paquete de diseño Dispositivos tiene una relación de traza con el paquete de análisis con el mismo nombre, y éste a su vez con el paquete de casos de uso.

Este paquete también pertenece a los paquetes de administración, los cuales tienen una vista común llamada Vista Administración, que da acceso a todas las funciones administrativas del sistema.

El paquete Dispositivos implementa todas las funcionalidades relacionadas con la gestión de dispositivos, sensores, tipos de dispositivos y modelos de representación 3D de los dispositivos.

El diseño también sigue el patrón MVC para cada una de sus clases tal y como muestra el siguiente diagrama de clases, donde queda perfectamente reflejado.

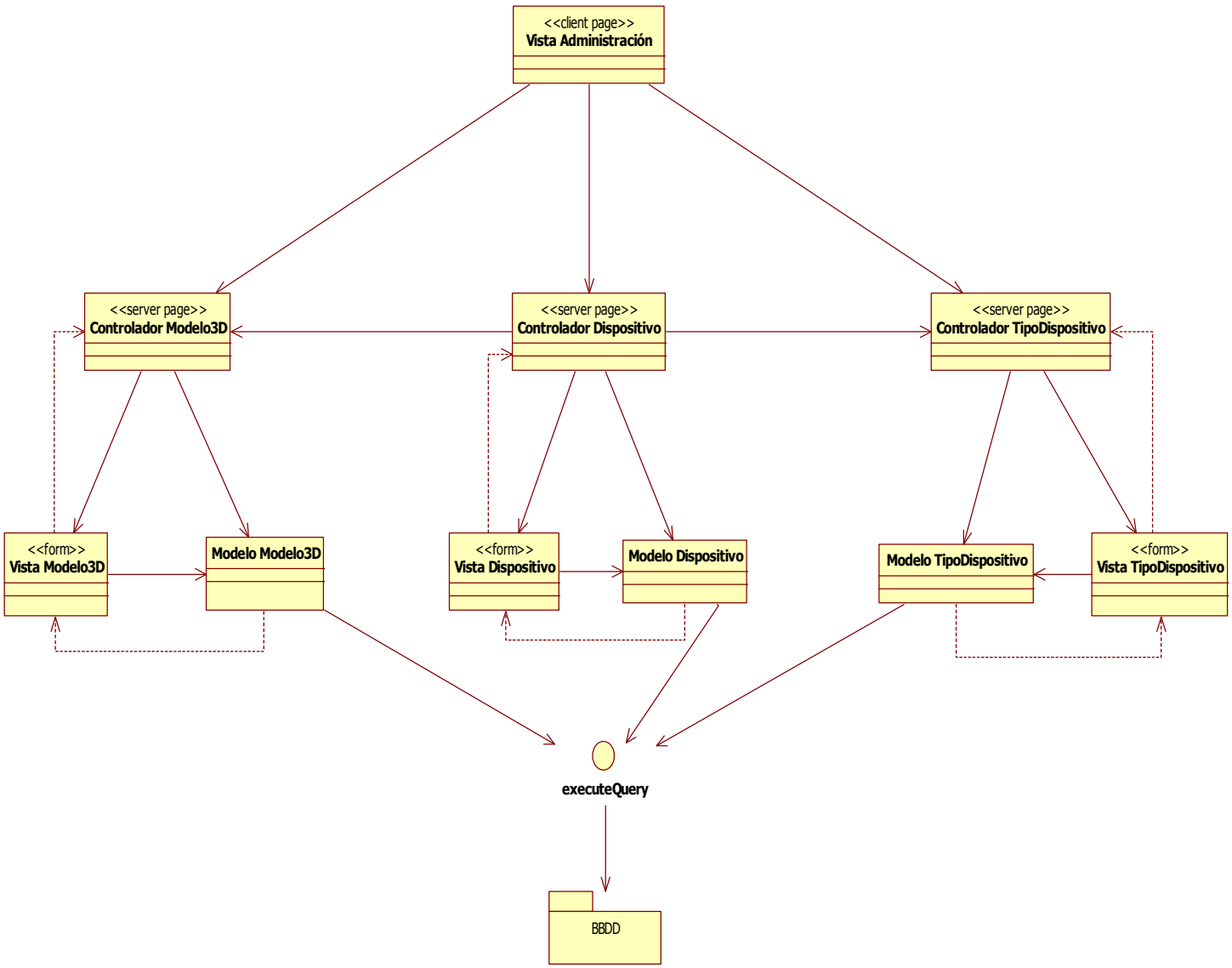


Figura 5.104. Diagrama de clases de dispositivos.

Crear dispositivo

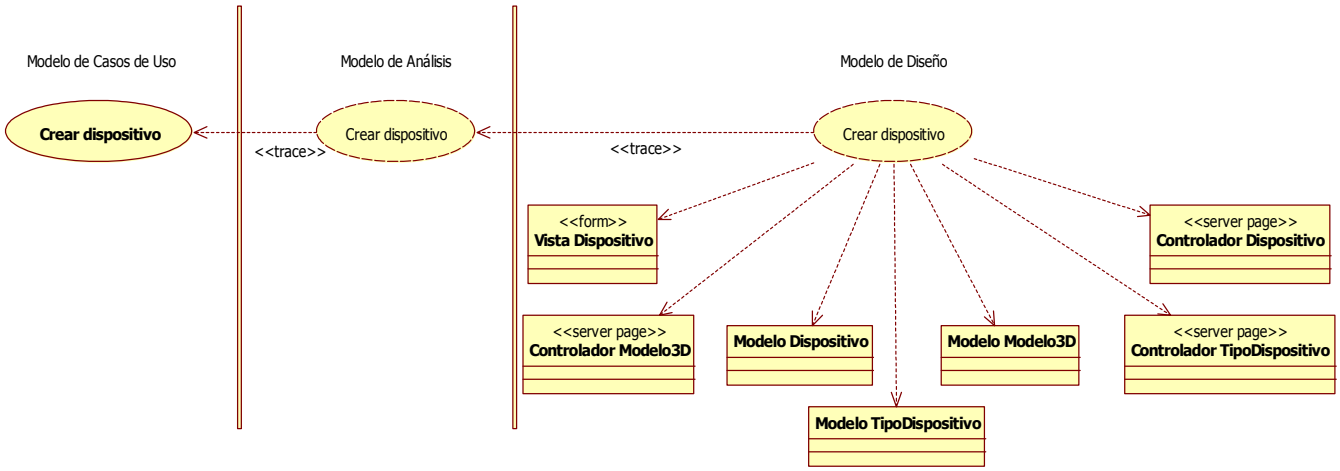


Figura 5.105. Traza (Caso de uso - Análisis - Diseño) crear dispositivo.

Para rellenar los campos de tipos de dispositivos y los modelos 3D con los existentes en el sistema para que el usuario seleccione el deseado se deben obtener desde la base datos. Para ello el controlador de dispositivos se los solicita a cada uno de los controladores correspondiente de forma que estos los obtienen desde la base de datos a través del sus correspondientes modelos y posteriormente se los devuelve al controlador que lanzó la petición.

El propio controlador de dispositivos renderiza la vista con el formulario que el usuario rellenará y al enviar el formulario nuevamente será tratado por el controlador que tratará los datos y los almacenará en la base de datos a través del modelo dispositivos.

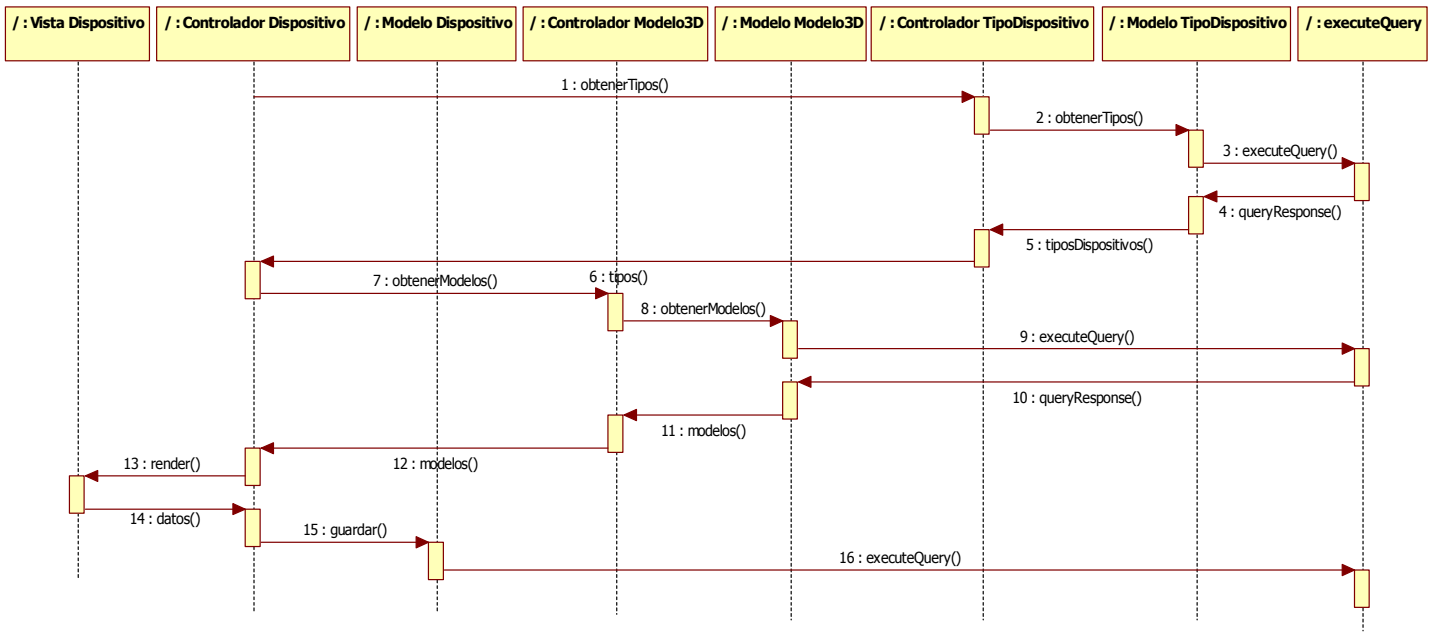


Figura 5.106. Diagrama de secuencia crear dispositivo.

Editar dispositivo

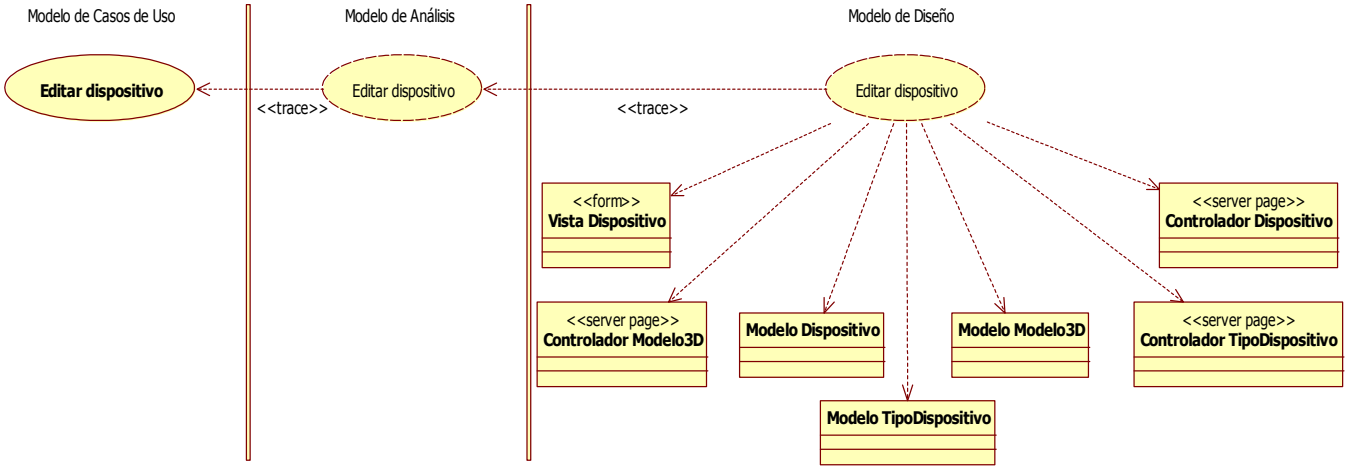


Figura 5.107. Traza (Caso de uso - Análisis - Diseño) editar dispositivo.

El proceso para editar un dispositivo es muy parecido al de la creación con la salvedad de que el primer paso es recuperar los datos del dispositivo desde la base datos. A continuación y al igual que sucede en la creación se obtienen los campos tipo de dispositivo y modelos 3D aunque esta vez se preselecciona el que ya se haya recuperado del dispositivo. La secuencia continúa como en el anterior, se renderiza la vista con el formulario, y cuando el usuario realiza los cambios y envía el formulario, el controlador trata los datos y los almacena en la base de datos mediante el correspondiente modelo.

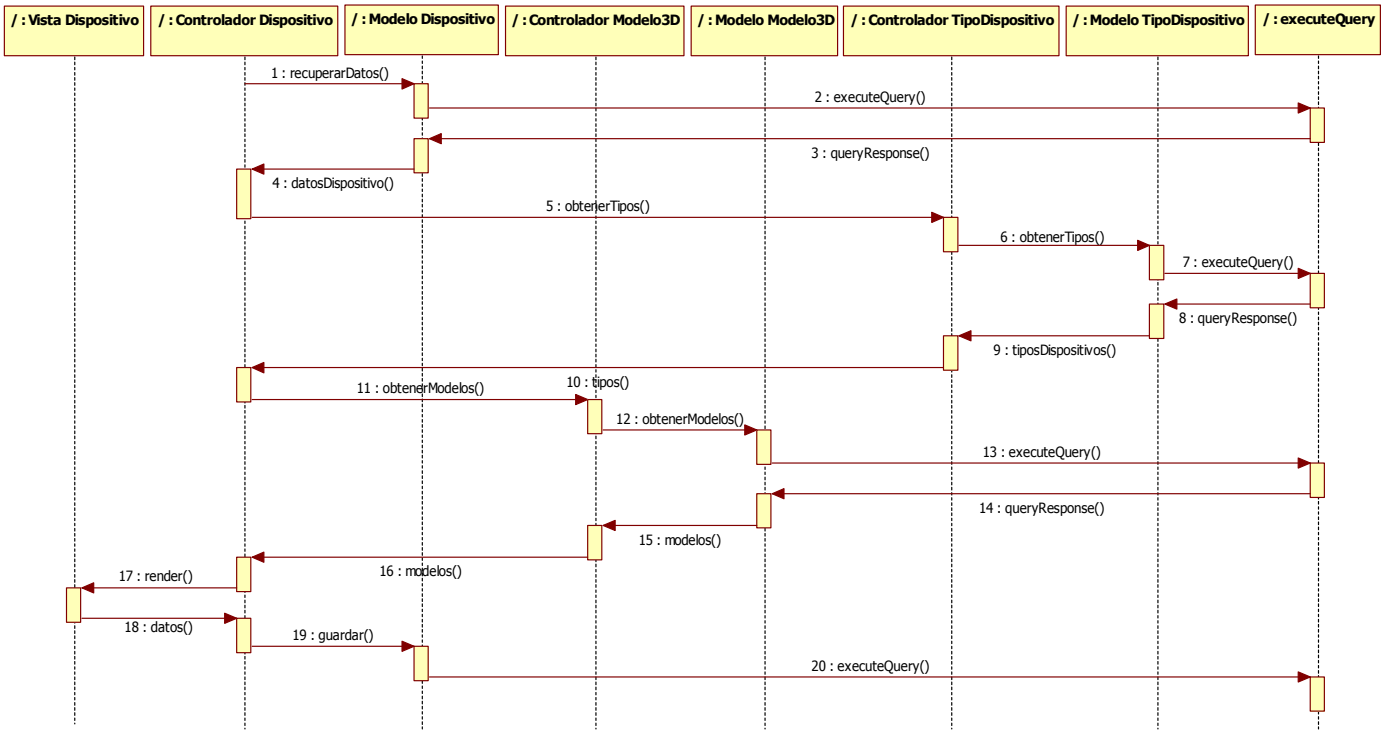


Figura 5.108. Diagrama de secuencia editar dispositivo.

Eliminar dispositivo

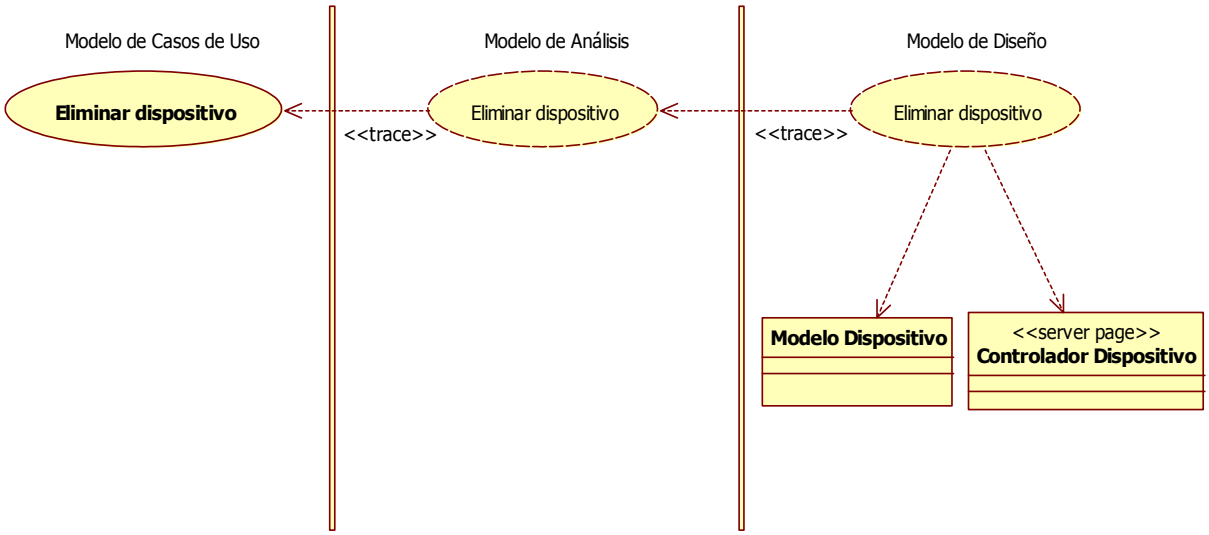


Figura 5.109. Traza (Caso de uso - Análisis - Diseño) eliminar dispositivo.

Cuando el controlador de dispositivos recibe una solicitud de eliminación de dispositivo hace las comprobaciones pertinentes, y éste le envía la petición al modelo de dispositivos que ejecuta la sentencia de "Delete From" de la base de datos con el identificador dado.

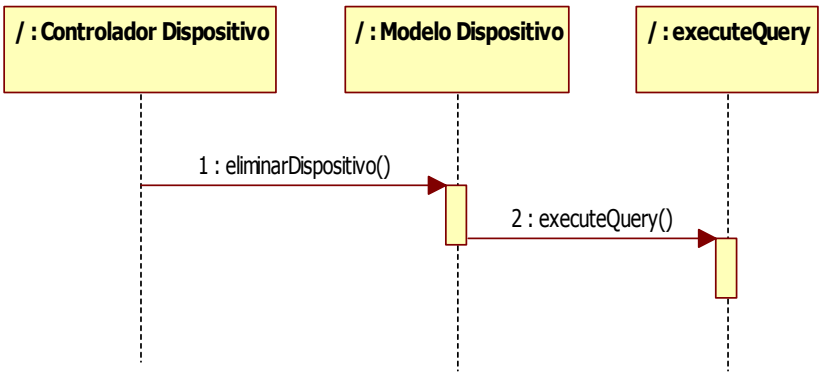


Figura 5.110. Diagrama de secuencia eliminar dispositivo.

Crear tipo de dispositivo

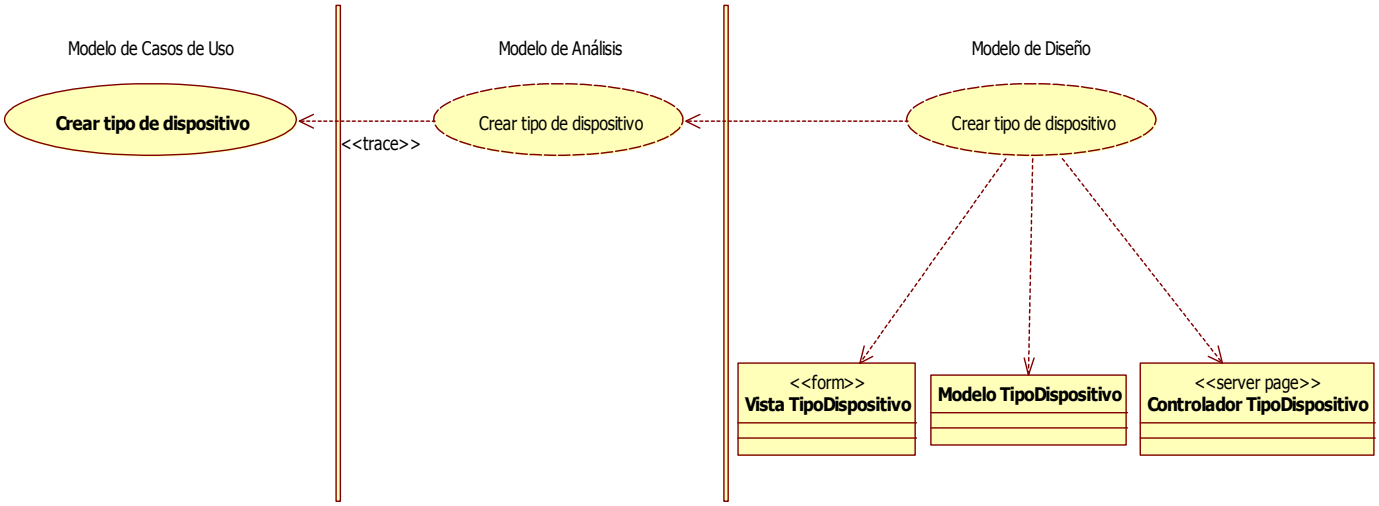


Figura 5.111. Traza (Caso de uso - Análisis - Diseño) crear tipo dispositivo.

El controlador de tipo de dispositivos genera el formulario de inserción de datos y lo renderiza en la vista para que el usuario rellene los campos. Una vez el usuario los envía el controlador recoge la petición y una vez valida y trata los datos los almacena en la base de datos mediante el modelo.

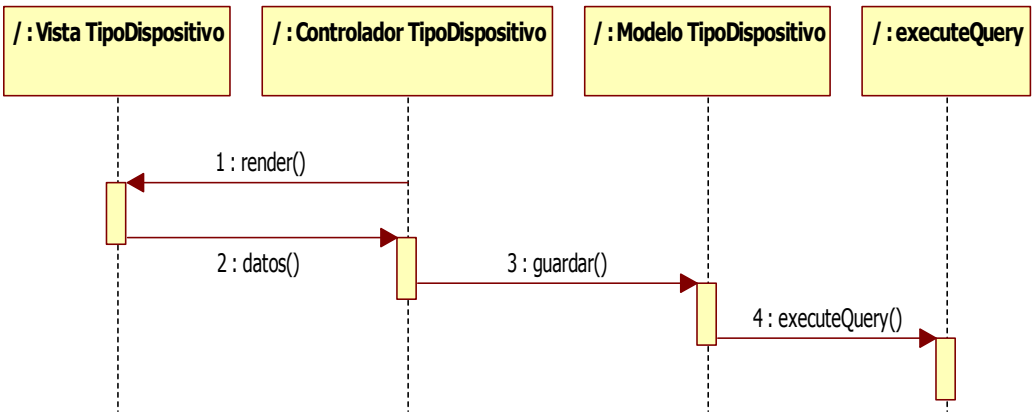


Figura 5.112. Diagrama de secuencia crear tipo dispositivo.

Editar tipo de dispositivo

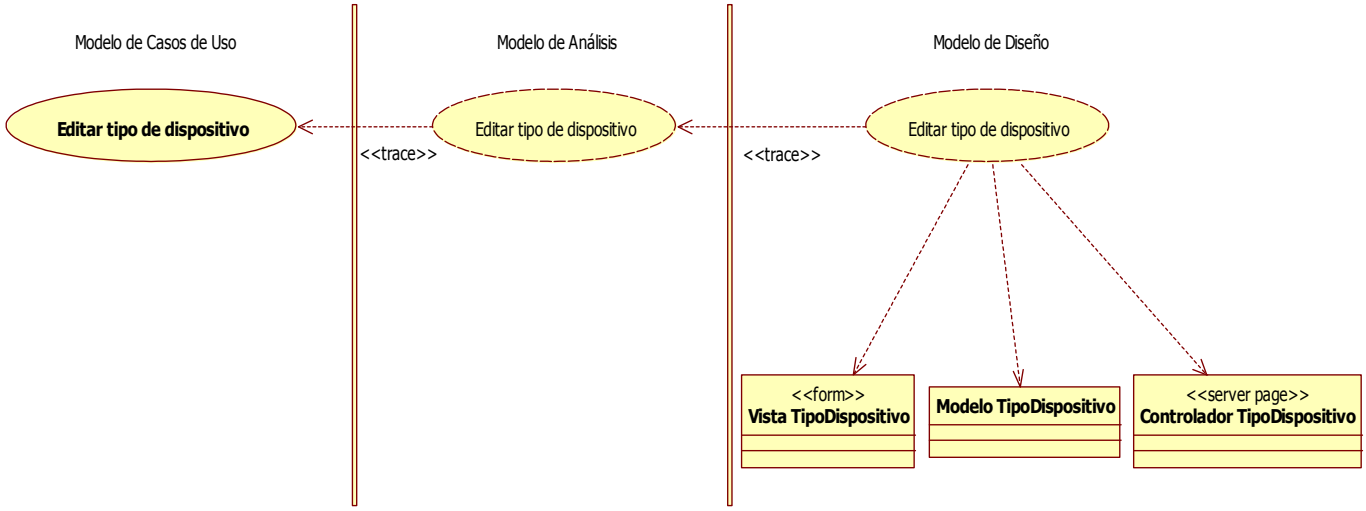


Figura 5.113. Traza (Caso de uso - Análisis - Diseño) editar tipo dispositivo.

Para editar un tipo de dispositivo el controlador recupera los datos del tipo desde la base de datos haciendo uso del modelo. Una vez recuperados los datos muestra la vista donde se encuentra el formulario con los datos preinsertados. Una vez el usuario los modifica e intenta guardarlos el controlador captura la petición, valida los datos y el modelo los actualiza en la base de datos.

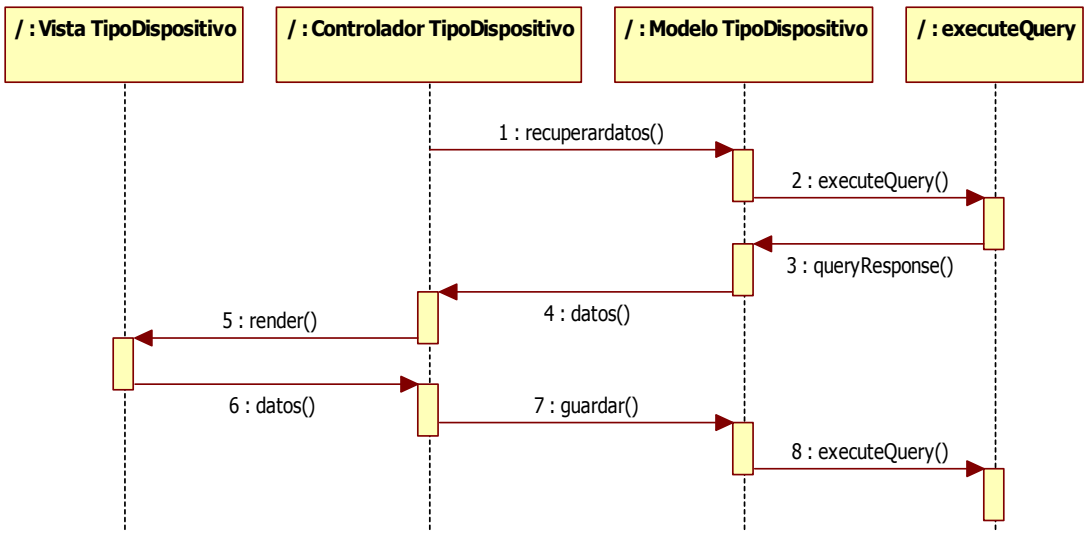


Figura 5.114. Diagrama de secuencia editar tipo dispositivo.

Eliminar tipo de dispositivo

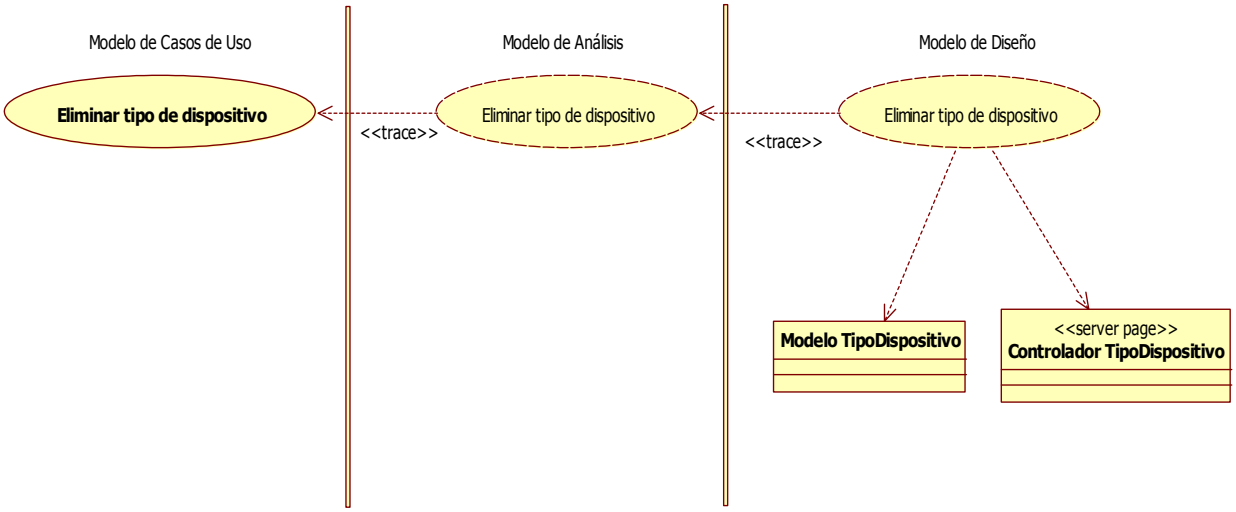


Figura 5.115. Traza (Caso de uso - Análisis - Diseño) eliminar tipo dispositivo.

Para eliminar un tipo de dispositivo el controlador de tipos de dispositivo recibe la petición y se la traslada al modelo que es el encargado de llevar a cabo la eliminación en la base de datos borrando así el registro perteneciente al identificador seleccionado.

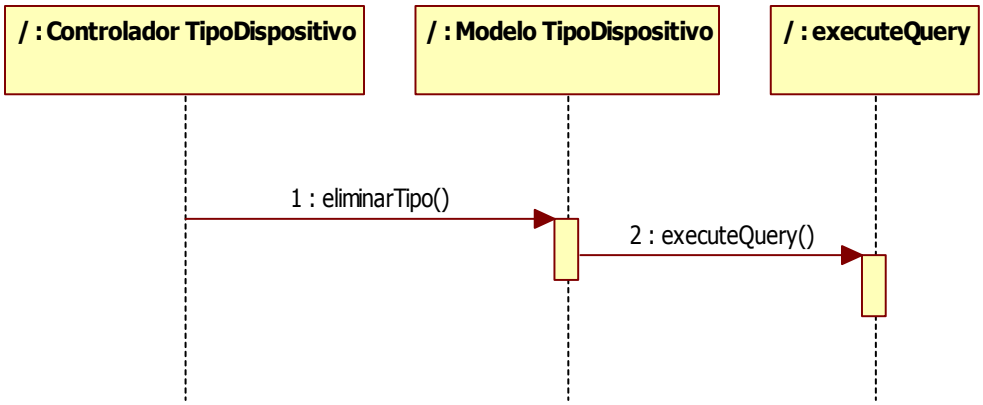


Figura 5.116. Diagrama de secuencia eliminar tipo dispositivo.

Introducir modelo

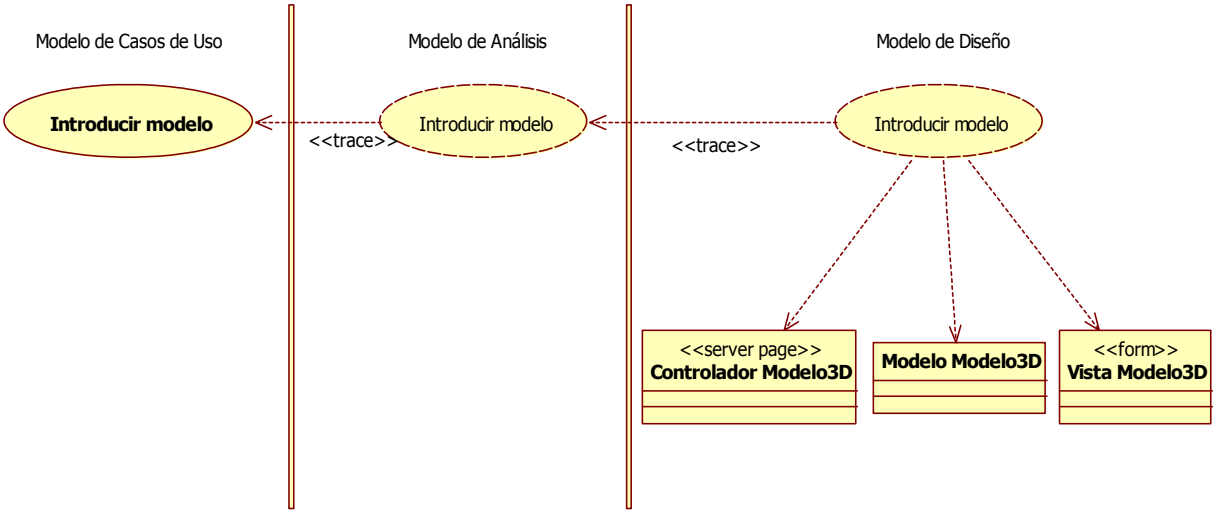


Figura 5.117. Traza (Caso de uso - Análisis - Diseño) introducir modelo 3D.

El controlador modelo3D genera la vista con el formulario para la inserción de los datos del modelo 3D y el fichero en formato collada que se desea insertar. Una vez el usuario introduce los datos y el modelo, el controlador los recibe y los guarda en la base de datos mediante el modelo correspondiente.

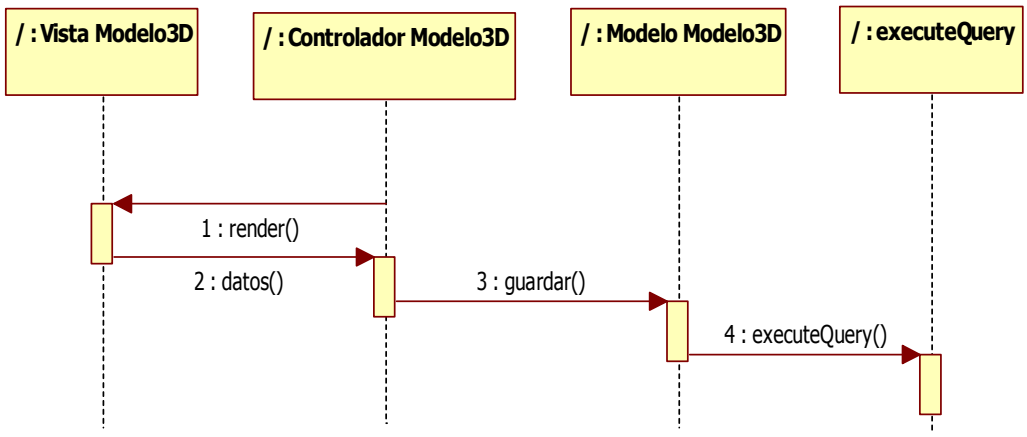


Figura 5.118. Diagrama de secuencia introducir modelo 3D.

Editar modelo

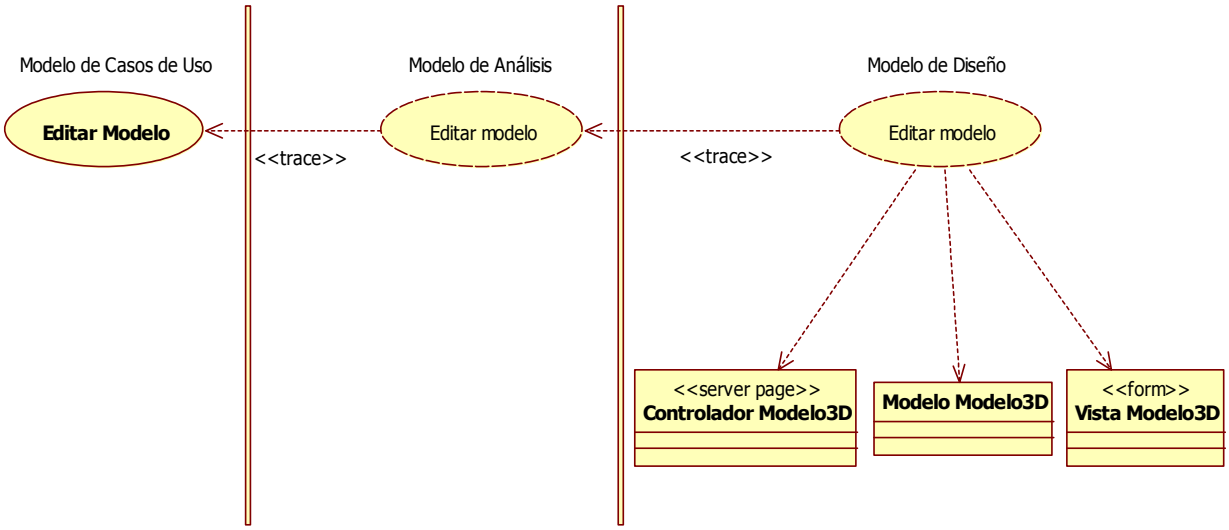


Figura 5.119. Traza (Caso de uso - Análisis - Diseño) editar modelo 3D.

Cuando se va a editar un modelo 3D el controlador recupera los datos de la base de datos haciendo uso de su correspondiente modelo. Una vez hecho, inserta esos datos en el formulario y renderiza la vista. Cuando los datos son modificados y se recibe la petición de guardado el controlador valida los datos y se los pasa al modelo que es el encargado de almacenarlo en la base de datos.

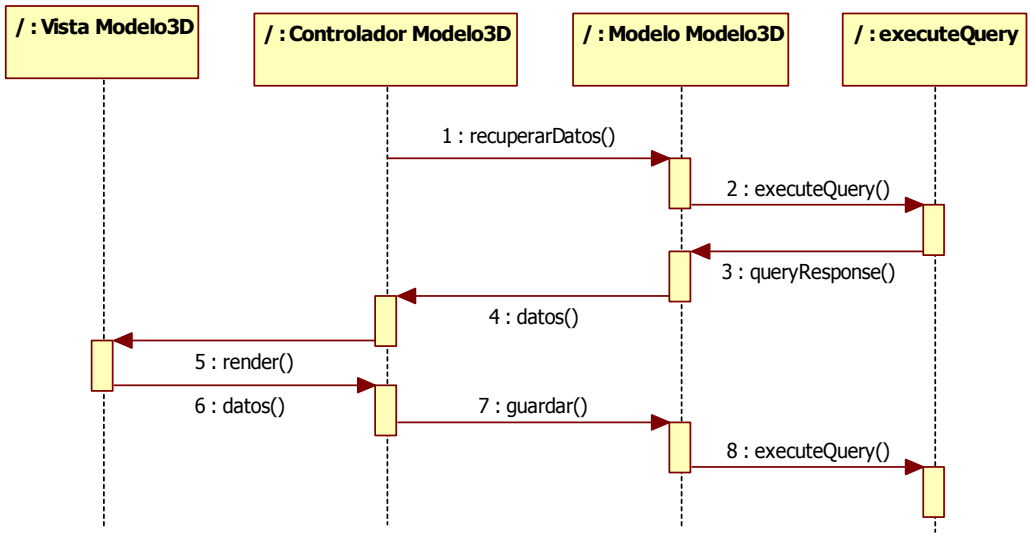


Figura 5.120. Diagrama de secuencia editar modelo 3D.

Eliminar modelo

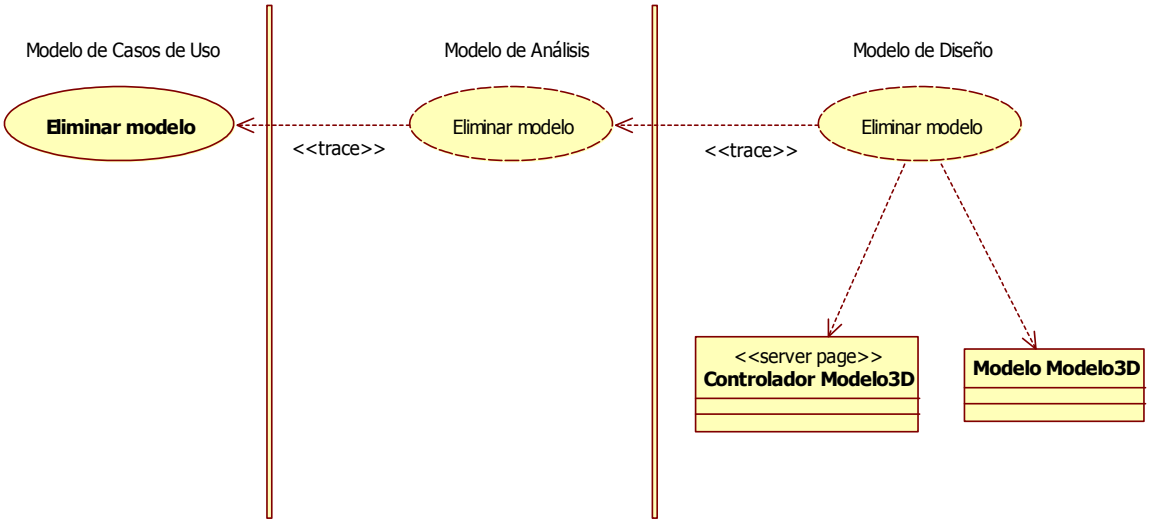


Figura 5.121. Traza (Caso de uso - Análisis - Diseño) eliminar modelo 3D.

Cuando un usuario desea eliminar un modelo 3D el controlador captura la petición y el identificador del elemento seleccionado. Ésto se le pasa al modelo y él lo elimina de la base de datos.

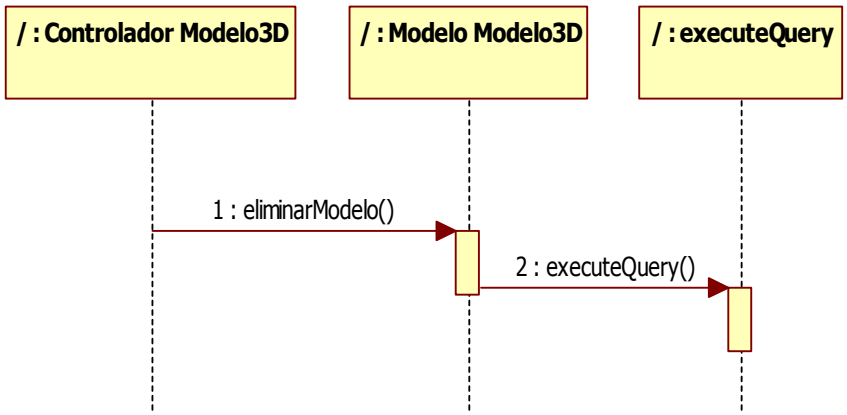


Figura 5.122. Diagrama de secuencia eliminar modelo 3D.

5.4.3. Modelo de despliegue.

El gráfico siguiente representa el despliegue de la aplicación tanto en los dominios de la organización como en el ámbito externo a ésta. Como se observa, hay dos partes claramente diferenciadas: la parte superior representa a las máquinas presentes en la organización y la parte inferior de la imagen son los PCs desde los cuales los usuarios pueden conectarse vía internet y acceder a los contenidos del portal.

Tanto las bases de datos como el servidor web se encuentran dentro de la intranet de PLOCAN pero son accesibles desde Internet. Sin embargo, los servidores de datos oceanográficos se encuentran de forma remota.

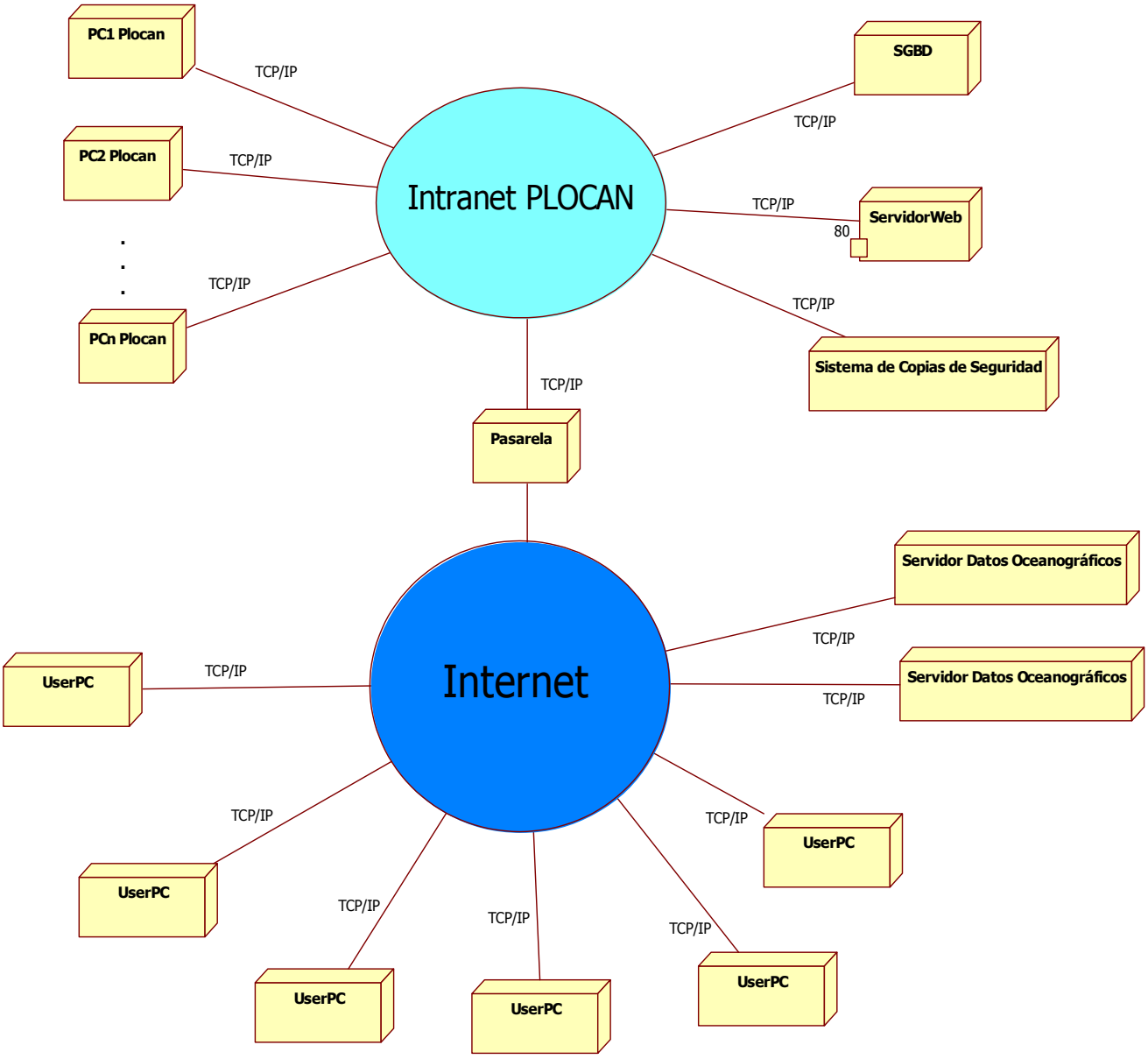


Figura 5.123. Diagrama del modelo de despliegue de la aplicación.

5.4.4. Diseño de la base de datos

A continuación se presentará el diseño de la base de datos mediante diagramas de Entidad-Relación. Se usa esta metodología ya que el proceso de creación de las tablas y sus relaciones es un paso directo entre el diagrama y su correspondiente sentencia SQL. Además de ello la documentación es totalmente clara y precisa.

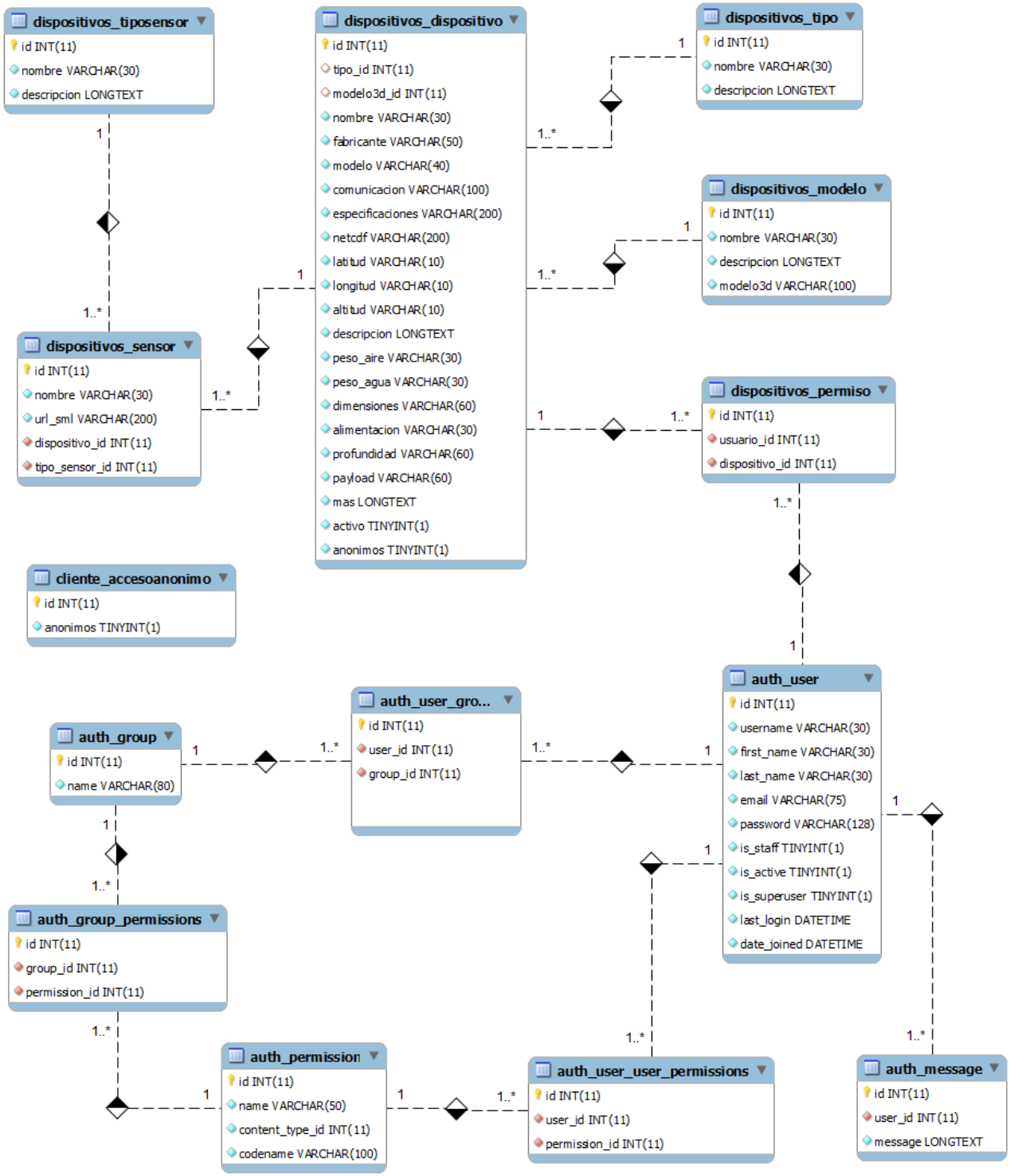


Figura 5.123. Diagrama Entidad-Relación de la base de datos.

5.4.5. Prototipo de Interfaz de Usuario

3DO es una aplicación web y por lo tanto su interfaz de usuario es basada en páginas web. La interfaz tiene dos partes perfectamente diferenciadas. Primero tiene una interfaz pública que es visible a todo el mundo y que se corresponde con el cliente de la aplicación. Y en segundo lugar hay una interfaz privada y restringida a los usuarios con los permisos pertinentes y se corresponde con la interfaz de administración de la aplicación.

5.4.5.1. Cliente

La interfaz del cliente está formada por una cabecera principal, un menú, un área de login y el elemento mas importante que es el mapa 3D.

<table>
<tr><td colspan="3">Cabecera Principal</td></tr>
<tr><td colspan="2">Menú Principal</td><td>Área de sesión de usuario</td></tr>
<tr><td>Menú Secundario</td><td colspan="2">Mapa 3D</td></tr>
</table>

El menú es un menú desplegable donde se listan los dispositivos que son accesibles, sus sensores y la plataforma oceánica. Al hacer click sobre un ítem del menú, el mapa entrará en acción y mostrará la localización del dispositivo seleccionado mostrando una vista general de la zona.

Al hacer click sobre un dispositivo en el mapa se abrirá una burbuja con la información de interés del dispositivo.

5.4.5.2. Administración

La interfaz de administración es de uso privado a usuarios con los permisos necesarios.

Se divide en 4 bloques: una cabecera, una zona de mensajes al usuario y acciones sobre esa cuenta de usuario, un bloque de administración de usuarios y un bloque de administración de dispositivos.

<table>
<tr><td colspan="2">Cabecera Principal</td></tr>
<tr><td>Bienvenida – Fecha:hora</td><td>Acciones sobre usuario actual</td></tr>
<tr><td colspan="2">Área de administración de usuarios</td></tr>
<tr><td colspan="2">Área de administración de dispositivos</td></tr>
</table>

El área de administración de usuarios solo será visible para los usuarios con permisos de administrador. En él se podrá acceder a la creación, modificación y eliminación de usuarios.

Por otro lado, el área de administración de dispositivos será visible tanto para usuarios con permisos de editor como de administrador. En este área estarán las herramientas para creación, edición y eliminación de dispositivos, tipos, modelos y sensores.

Una vez se accede a cualquiera de estos apartados de administración, todos tienen el mismo aspecto con la diferencia de los campos que aparecen en cada uno, pero siguiendo un mismo estilo.

5.5. Implementación

El presente epígrafe se centrará en la fase de implementación del producto. Durante esta etapa se construye el producto software a partir de los documentos y artefactos obtenidos hasta el momento en la fase de diseño.

La fase de implementación está marcada principalmente por la generación del código fuente de la aplicación, y por lo tanto, la mayor parte de la documentación va introducida en el propio código a forma de comentarios. Por esta razón la documentación de esta fase, en cuanto a este documento se refiere, estará estructurada de la siguiente manera:

1. **Instalación y configuración de Django.** Se enumeran los pasos llevados a cabo para la instalación y configuración del framework y librerías adicionales en cada uno de los entornos existentes.
2. **Control de versiones.** Se explica el sistema de control de versiones que se ha seleccionado y la manera de ejecutarlo con los recursos que se tenían disponibles.
3. **Organización del código.** Se define cómo está organizado el código y se muestran algunas partes relevantes donde se hayan implementado partes destacables o que sirvan de ejemplo para entender de forma global el proyecto.

5.5.1. Instalación y Configuración de Django

En el presente subapartado se explicará paso a paso la forma de instalar y configurar el framework de desarrollo Django y el resto de librerías Python utilizadas en el proyecto, además de otras herramientas como puede ser el propio interprete de Python, el sistema de gestión de base de datos MySQL o los diferentes servidores web para el despliegue de la aplicación.

La configuración es un poco diferente en el servidor de desarrollo y en los servidores de calidad y de producción, ya que el propósito de cada uno es distinto. En el servidor de desarrollo se estará escribiendo código y haciendo pruebas continuas y el estado y la configuración puede estar cambiando cada momento. Sin embargo, en el servidor de calidad o el servidor de producción, el estado es bastante mas estático y se espera una mayor estabilidad y disponibilidad. La principal diferencia radica en que para el servidor de desarrollo basta con el servidor web ligero que proporciona Django, mientras que para el servidor de calidad, donde se realizarán las pruebas beta, y para el servidor de producción, hacen falta servidores web robustos separados para la aplicación y los datos estáticos (ficheros CSS, JS, imágenes, etc).

5.5.1.1. Servidor de Desarrollo

El servidor de desarrollo estará montado sobre la propia máquina de desarrollo y se lanzará y se parará de forma dinámica según vaya haciendo falta en cada momento. Para la instalación y configuración de esté se llevarán a cabo los siguientes pasos:

1. **Instalar Python.** Django es un framework escrito en Python y por lo tanto requiere de su intérprete. Usaremos la versión 2.7 de Python. Para los sistemas Linux y Mac OS X Python viene preinstalado por defecto. En caso de sistemas Windows se puede obtener de la siguiente dirección (https://www.python.org/downloads), donde también están los recursos para Mac OS X y Linux por si se quiere actualizar el que tiene el sistema por defecto. En Windows, una vez instalado se debe añadir el directorio de Python al PATH del sistema para que el comando Python esté operativo desde cualquier ubicación.

2. **Instalar pip como gestor de librerías Python.** Para instalar pip como gestor de librerías lo primero que se debe hacer es instalar easy_install. Para ello se debe descargar ez_setup.py (https://bootstrap.pypa.io/ez_setup.py) y ejecutarlo con permisos de administrador mediante el siguiente comando:

   ```
   python ez_setup.py
   ```

   Una vez instalado easy_install ya podemos instalar pip de la siguiente manera:

   ```
   easy_install pip
   ```

   A partir de este momento, todos los paquetes y librerías Python se instalarán haciendo uso de pip.

3. **Instalar Django.** Para instalar Django se utiliza pip ya que se encuentra en el repositorio de Python.

   ```
   pip install django
   ```

   Con ésto ya se tendría Django instalado y listo para ser usado.

4. **Instalar MySQL.** El siguiente paso es llevar a cabo la instalación del Sistema de Gestión de Bases de Datos MySQL. Para la plataforma Windows se debe descargar el ejecutable desde su página web (http://dev.mysql.com/downloads/) e instalarlo en el sistema.

   Para sistemas Linux la forma más sencilla es a través de su gestor de paquetes. En ubuntu:

   ```
   sudo apt-get install mysql
   ```

   Y por otro lado es necesario instalar el conector de Python con MySQL.

   ```
   pip install mysql-python
   ```

5. **Instalar resto de librerías Python.** El proyecto hace uso de varias librerías Python externas que se deben instalar. Al igual que con los otros paquetes Python que se han instalado, se instalan mediante pip.

   ```
   #Instalación de simplekml, da soporte para kml
   pip install simplekml


   # Instalación ScientificPython, da soporte a NetCDF, y tiene dependencia con numpy
   pip install numpy
   pip install ScientificPython


   #Instalación xmltodict, soporte para pasar de xml a estructura de dict
   pip install xmltodict
   ```

6. **Crear la base de datos.** El siguiente paso es la creación de la base de datos para el proyecto.

   ```
   mysqladmin create 3do -u root -p
   ```

7. **Crear el usuario de la base de datos.** Por seguridad, se debe crear un usuario con permisos sólo sobre esta base de datos para no hacer uso del usuario root.

   ```
   mysql -u root -p GRANT ALL ON p360.* TO 'p360'@'localhost' IDENTIFIED BY 'p360'
   ```

8. **Copiar el código del programa.** Una vez se copia el código del programa ya solo queda modificar el fichero settings.py para configurar la conexión con la base de datos.

```
#Fichero SETTINGS.PY
...
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': '3do',
        'USER': '3do',
        'PASSWORD': '',
        'HOST': '127.0.0.1',
        'PORT': '3306',
    }
}
...
```

9. **Crear las tablas en la BD.** Para crear las tablas en la base de datos basta con ejecutar un comando desde la raíz del proyecto.

```
python manage.py syncdb
```

10. **Arrancar servidor de desarrollo.**

```
python manage.py runserver
```

Con ésto la aplicación queda accesible desde http://localhost:8000

5.5.1.2. Servidor de Calidad y Producción

Aunque el servidor de calidad y de producción serán dos servidores distintos, su instalación y configuración será igual porque ambos están montados sobre sistemas Linux bajo distribuciones derivadas de Red Hat (CentOS, Fedora, etc).

Por un lado el servidor de calidad estará alojado en una instancia Cloud Computing de Amazón EC2. Y por otro lado, el servidor de producción será un servidor propiedad de la Plataforma Oceánica de Canarias con sistema CentOS.

A continuación se pasará a detallar cada uno de los pasos necesarios para la instalación y configuración del framework que, sobre todo en los primeros pasos, tiene bastante coincidencia con los enumerados en el aparatado anterior para el servidor de desarrollo.

1. **Instalar Python.** Al ser un sistema Linux, en principio Python debería estar preinstalado por defecto pero en caso de querer asegurarnos se debería ejecutar el siguiente comando.

```
sudo yum install python
sudo yum install python-devel
```

2. **Instalar pip como gestor de librerías Python.** Como ya se explicó anteriormente, para instalar pip como gestor de librerías lo primero que se debe hacer es instalar easy_install. Para ello se debe descargar ez_setup.py (https://bootstrap.pypa.io/ez_setup.py) y ejecutarlo y luego instalar pip.

```
python ez_setup.py
easy_install pip
```

3. **Instalar Django.** Para instalar Django se utiliza pip ya que se encuentra en el repositorio de Python.

```
pip install django
```

4. **Instalar MySQL.** El siguiente paso es llevar a cabo la instalación del Sistema de Gestión de Bases de Datos MySQL.

```
sudo yum install mysql
```

Y por otro lado es necesario instalar el conector de Python con MySQL.

```
pip install mysql-python
```

5. **Instalar resto de librerías Python.**

```
#Instalación de simplekml, da soporte para kml
pip install simplekml

# Instalación ScientificPython, da soporte a NetCDF, y tiene dependencia con numpy
pip install numpy
pip install ScientificPython

#Instalación xmltodict, soporte para pasar de xml a estructura de dict
pip install xmltodict
```

6. **Crear la base de datos.** El siguiente paso es la creación de la base de datos para el proyecto.

```
mysqladmin create 3do -u root -p
```

7. **Crear el usuario de la base de datos.** Por seguridad, se debe crear un usuario con permisos sólo sobre esta base de datos para no hacer uso del usuario root.

```
mysql -u root -p GRANT ALL ON p360.* TO 'p360'@'localhost' IDENTIFIED BY 'p360'
```

8. **Copiar el código del programa.** Una vez se copia el código del programa ya sólo queda modificar el fichero settings.py para configurar la conexión a la base de datos.

```
#Fichero SETTINGS.PY
...
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': '3do',
        'USER': '3do',
        'PASSWORD': '',
        'HOST': '127.0.0.1',
        'PORT': '3306',
    }
}
...
```

9. **Crear las tablas en la BD.** Para crear las tablas en la base de datos basta con ejecutar un comando desde la raíz del proyecto.

```
python manage.py syncdb
```

10. **Instalar de nginx.** Nginx es un servidor webinverso ligero de alto rendimiento y es el servidor web que vamos a usar para servir la aplicación.

```
sudo yum install nginx
```

Una vez instalado nginx, se debe modificar el fichero */etc/nginx/nginx.conf* para que quede de la siguiente manera:

```
# nginx.conf


worker_processes 1;

user nginx;

pid /tmp/nginx.pid;

error_log  /var/log/nginx/error.log;


events {

    worker_connections 1024;

    accept_mutex off;

}


http {

    include mime.types;

    default_type application/octet-stream;

    access_log /tmp/nginx.access.log combined;

    sendfile             on;

    gzip                 on;

    gzip_http_version    1.1;

    gzip_disable         "msie6";
```

```
    gzip_vary            on;
    gzip_min_length      1100;
    gzip_buffers         64 8k;
    gzip_comp_level      3;
    gzip_proxied         any;
    gzip_types         text/plain text/css application/x-javascript text/xml
application/xml;


    upstream app_server {
        server unix:/tmp/gunicorn.sock fail_timeout=0;
        # For a TCP configuration:
        # server 192.168.0.7:8000 fail_timeout=0;
    }


    server {
        listen 80 default;
        client_max_body_size 4G;
        server_name _;
        root /var/www;     # path for static files
        location / {
            try_files $uri @proxy_to_app;
        }
        location @proxy_to_app {
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header Host $http_host;
            proxy_redirect off;
            proxy_pass   http://127.0.0.1:1337;
        }
    }
}
```

11. **Instalar gunicorn.** Gunicorn es un servidor WSGI HTTP para Python.

```
pip install gunicorn
```

Una vez instalado gunicorn se debe instalar en la aplicación. Para eso se debe modificar el fichero settings.py el proyecto.

```
#Fichero SETTINGS.PY
...
INSTALLED_APPS = (
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    #'django.contrib.sites',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'django.contrib.admin',
    'P3DO.dispositivos',
    'P3DO.usuarios',
    'P3DO.cliente',
     'gunicorn',
)
...
```

12. **Arrancar servidor web.** Para que la aplicación sea accesible, lo único que hay que hacer es iniciar nginx y gunicorn.

```
sudo /etc/init.d/nginx start


python /RUTA_DEL_PROYECTO/manage.py run_gunicorn -b 127.0.0.1:1337 -daemon
```

5.5.2. Control de Versiones

Por control de versiones es como se conoce a la gestión de los cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Una versión o revisión es el estado en el que se encuentra el mismo en un momento dado de su desarrollo o modificación. Aunque puede realizarse de forma manual, es muy aconsejable disponer de herramientas que faciliten esta gestión dando lugar a los llamados sistemas de control de versiones. Estos sistemas facilitan la administración de las distintas versiones de cada producto desarrollado y la recuperación de ellas en cualquier momento que sea necesario.

En este proyecto se ha decidido usar como sistema de control de versiones Subversion. Subversion, o SVN, es una herramienta de control de versiones basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Utiliza el concepto de revisión para guardar los cambios producidos en el repositorio. Entre dos revisiones sólo guarda el conjunto de modificaciones, optimizando así al máximo el uso de espacio en disco.

Un repositorio de subversión puede estar en un directorio local o alojado en un servidor remoto accesible a través de una conexión de red. Esta segunda opción es la más recomendada, ya que en caso de problemas con el disco o con el sistema, siempre habría una copia del repositorio en otra máquina remota, y por otro lado, se podría obtener una copia del proyecto desde cualquier máquina con conexión a Internet y con los credenciales oportunos.

Dado que durante el comienzo del proyecto no se disponía de un servidor donde poder montar el repositorio de Subversion y que, por otro lado, no se quería renunciar a tenerlo en remoto por las razones comentadas, había que buscar alguna otra solución que permitiese poder optar a ello.

Entonces la solución encontrada básicamente es una combinación de las dos opciones. Lo que se ha hecho ha sido crear el repositorio SVN en una carpeta de una cuenta de Dropbox. Con ello el repositorio funciona como un repositorio local, ya que con Dropbox se consigue tener un directorio local que a su vez sincroniza con el servicio de Dropbox manteniendo una copia en un servidor remoto. De cara al sistema de Subversion, al hacer “checkout” del repositorio, la copia se obtiene desde un repositorio local, y lo mismo ocurre al validar una versión, pero Dropbox se encarga de mantener una copia en sus servidores de forma que está accesible desde cualquier máquina con conexión a Internet con la cuenta de Dropbox.

Gracias a esta solución se consigue las ventajas deseadas de un repositorio SVN en un servidor remoto sin la necesidad de tener un servidor disponible. Aunque en principio surgió como una prueba, al ver su correcto funcionamiento se siguió usando durante el desarrollo de todo el proyecto sin ningún tipo de problema y funcionando tal y como se esperaba. Por esta razón finalmente se convirtió en la solución definitiva.

5.5.3. Organización del Código

La organización del código del proyecto se basa en las recomendaciones de la documentación de Django. Aunque es un framework muy flexible existen una serie de recomendaciones o buenas prácticas la hora de estructurar el código de un proyecto que lo use como base.

Tal y como se definió en la fase de diseño y buscando como meta el permitir la modularidad y el desarrollo incremental, la aplicación está dividida en tres módulos independientes que interactúan entre sí. En Django, cada módulo es conocido como aplicación y está compuesto básicamente por seis tipos de ficheros, que pueden estar o no, dependiendo de las necesidades del módulo. Estos ficheros son los siguientes:

- **models.py**

  Son los ficheros donde se definen los modelos de los distintos módulos. Los modelos son las clases que interactúan con la base de datos y normalmente tienen una correspondencia directa con las tablas en la base de datos cuyo nombre será NombreMódulo_NombreModelo. Además en los modelos también se incluyen algunos parámetros de validación, tipos de datos, relaciones con otros modelos, etc. A continuación se muestra un ejemplo de modelo.

```
class Sensor(models.Model):
     tipo_sensor = models.ForeignKey(TipoSensor, verbose_name='Sensor Type', null=True,
                    on_delete=models.SET_NULL)


     nombre = models.CharField('Name', max_length=30,unique=True)
     url_sml = models.FileField('SensorML File', upload_to='dispositivos/sml')


     dispositivo = models.ForeignKey(Dispositivo, verbose_name='Device', null=True,
                    on_delete=models.SET_NULL)


     def __unicode__(self):
            return ""


     class Meta:
              ordering = ['nombre']
              verbose_name = "Sensor"
              verbose_name_plural = "Sensors"
```

- **views.py**

  Junto con el fichero de rutas representan al controlador del MVC y es donde se define la lógica de negocio y donde se hace todo el tratamiento y manipulación de los datos. En estos ficheros se definen funciones que serán referenciadas por el resolvedor de rutas y serán las encargadas del backend de la aplicación. Estas funciones serán las encargas de recoger los datos que vienen de los formularios, hacer el tratamiento que sea necesario, hacer las llamadas a los métodos de los modelos para ejecutar las instrucciones oportunas en las base de datos y alimentar y renderizar las plantillas que serán enviadas de nuevo al cliente mediante un HTTP Response. Un ejemplo sería el siguiente:

```
def index(request):
    user = request.user
    tipos = Tipo.objects.all()
    publico = AccesoAnonimo.objects.get(pk=1).anonimos
    dispositivos={}
    for tipo in tipos:
        if user.is_authenticated():
            dispositivos[tipo.nombre]=tipo.dispositivo_set.filter(Q(anonimos=True) |
                    Q(permiso__usuario=user), activo=True)
        else:
            dispositivos[tipo.nombre]=tipo.dispositivo_set.filter(anonimos=True,
                    activo=True)
    tipossensor = TipoSensor.objects.all()
    sensores={}
    for tiposensor in tipossensor:
        if user.is_authenticated():
            sensores[tiposensor.nombre]=
                    tiposensor.sensor_set.filter(Q(dispositivo__anonimos=True)
                    | Q(dispositivo__permiso__usuario=user), dispositivo__activo=True)
        else:

             sensores[tiposensor.nombre]=
                    tiposensor.sensor_set.filter(dispositivo__anonimos=True,
                    dispositivo__activo=True)
    kml= obtenerkml(request)
    try:
        if request.session["error_login"]:
            error_login=True
        else:
            error_login=False
        del request.session["error_login"]
    except KeyError:
        error_login=False
    return render(request,'index.html',
             {"dispositivos":dispositivos,"sensores":sensores,"error_login":
             error_login,"publico": publico,"kml": kml,})
```

- **forms.py**

  Es donde se definen los formularios. En él se definen los campos de un formulario, los tipos de datos que aceptan y las opciones por defecto. Lo siguiente es el ejemplo del formulario de selección de las magnitudes físicas que se desean mostrar en una gráfica.

```
class ChartForm(forms.Form):
    ejex = forms.ChoiceField(label='X-axis',choices=[['TIME','Time']])
    ejey = forms.ChoiceField(label='Y-axis')
    def __init__(self, id, *args, **kwargs):
        super(ChartForm, self).__init__(*args, **kwargs)
        dispositivo = Dispositivo.objects.get(pk=id)
        fpath=dispositivo.netcdf.path
        f=NetCDFFile(fpath,'r')
        opcionesvar=[]
        varnames = f.variables.keys()
        vars = f.variables
        for varname in varnames:
            if varname.find('_QC')>0 or varname=='TIME' or varname=='LATITUDE' or
             varname=='LONGITUDE' or varname=='POSITION':
                continue
            var = vars[varname]
            attrs = dir(var)
            varqc = None
            if 'QC_indicator' in attrs:
                if getattr(var,'QC_indicator') != 1:
                    continue
                else:
                    varqc = None
            else:
                varqcname = varname+"_QC"
                if not varqcname in varnames:
                    varq = None
                else:
                    varqc = vars[varqcname][:]
            if varqc!=None and not 1 in varqc:
                continue
            if 'standard_name' in attrs:
                nombre=getattr(var,'standard_name').replace('_',' ').title()
            else:
                nombre=varname.capitalize()
            opcionesvar.append([varname,nombre])
        self.fields['ejey'].choices = opcionesvar
```

- **Templates (fichero.html)**

  Los fichero con extensión html en Django se corresponden con las vistas del MVC. Django utiliza un sistema de plantillas para la generación del código HTML con el que se presentarán los resultados al usuario final. Este sistema de plantilla se basa en la definición de plantillas más generales, las cuales se van extendiendo para crear las más específicas mediante la declaración de bloques que van siendo redefinidos según las necesidades. Lo siguiente es un ejemplo de lo que se está hablando.

```
{% extends "admin/base_site.html" %}
{% load i18n %}
{% block breadcrumbs %}{% endblock %}
{% block content_title %}{% endblock %}
{% block style_content %}style=" position:relative;top: 50%;left: 50%;width:500px; margin-
left:-250px; margin-bottom:25px"{% endblock %}


{% block extrahead %}
    <script type="text/javascript">
        $(document).ready(function(){
            $("#accesopublico").mouseover(function(){
                $(".accessvisible").hide();
                $(".accesshidden").show();
            });

            $("#accesopublico").mouseout(function(){
                $(".accesshidden").hide();
                $(".accessvisible").show();
            });
        });
    </script>
{% endblock %}
{% block content %}
{% if perms.auth %}
    <div class="moduletitle"> User Management </div>
    <div class="modulenav">
        <div class="modulebuttoncont">
            <a href="auth/user">
                <div class="modulebutton"><img src="/static/images/user3.png"
alt="Usuarios" /><span>Users</span></div>
            </a>
            <a href="auth/group">
                <div class="modulebutton"><img src="/static/images/group3.png"
alt="Grupos" /><span>Groups</span></div>
            </a>
            <a href="{% url 'P3DO.cliente.views.accesopublico' %}">
```

```
                <div class="modulebutton" id="accesopublico">
                {% if publico %}
                    <img src="/static/images/openlock.png" class="accessvisible" />
                     <span class="accessvisible">Public Access<br />Open</span>
                    <img src="/static/images/closelock.png"class="accesshidden" />
                     <span class="accesshidden">Close<br />Public Access</span>
                {% else %}
                    <img src="/static/images/closelock.png" class="accessvisible" />
                     <span class="accessvisible">Public Access<br />Closed</span>
                    <img src="/static/images/openlock.png" class="accesshidden" />
                     <span class="accesshidden">Open<br />Public Access</span>
                {% endif %}
                </div>
            </a>
        </div>
    </div>
{% endif %}
{% if perms.dispositivos %}
    <div class="moduletitle"> Device Management </div>
    <div class="modulenav">
        <div class="modulebuttoncont">
            <a href="dispositivos/dispositivo">
                <div class="modulebutton"><img src="/static/images/dispositivo.png"
                     alt="Dispositivos" /><span>Devices</span></div>
            </a>
            <a href="dispositivos/tipo">
                <div class="modulebutton"><img src="/static/images/tipodispositivo.png"
                     alt="Tipos de Dispositivos" /><span>Device<br>Types</span></div>
            </a>
            <a href="dispositivos/sensor">
                <div class="modulebutton"><img src="/static/images/sensor.png"
                              alt="Sensores" /><span>Sensors</span></div>
            </a>
            <a href="dispositivos/tiposensor">
                <div class="modulebutton"><img src="/static/images/tiposensor.png"
                     alt="Tipos Sensores" /><span>Sensor<br>Types</span></div>
            </a>
            <a href="dispositivos/modelo">
                <div class="modulebutton"><img src="/static/images/3d.png"
                     alt="Modelos 3D" /><span>3D<br>Models</span></div>
            </a>
        </div>
    </div>
{% endif %}
```

- **urls.py**

  Son los ficheros donde se definen, mediante expresiones regulares, las rutas que aceptará la aplicación y a los métodos a los que se redirigirá una HTTP Request. El siguiente ejemplo muestra las rutas del módulo cliente.

```
from django.conf.urls import patterns, include, url


urlpatterns = patterns('',
    url(r'^$', 'P3DO.cliente.views.index', name='index'),
    url(r'^infodevice/(\d+)/$', 'P3DO.cliente.views.infodevice', name='infodevice'),
    url(r'^generagrafica/(\d+)/$', 'P3DO.cliente.views.generagrafica',
            name='generagrafica'),
    url(r'^infosensor/(\d+)/$', 'P3DO.cliente.views.getInfoSensor', name='infosensor'),
)
```

- **admin.py**

  Este tipo de fichero son usados para configurar cómo se visualizará un modelo específico en la interfaz de administración de la aplicación. Básicamente su función es definir qué campos aparecen en un listado o en el mantenimiento del modelo, cómo se agrupan esos campos, qué campos que hacen alusión a modelos relacionados se pueden editar en el mismo formulario, por qué campos se pueden filtrar las búsquedas, por qué campo se ordena por defecto y otro tipo de opciones. El siguiente ejemplo muestra cómo se ha definido estas opciones para el administrador de dispositivos.

```
class DispositivoAdmin(admin.ModelAdmin):
    fieldsets = [
        ('Device',   {'fields': ['tipo','modelo3d','nombre','fabricante', 'modelo',
            'comunicacion','especificaciones','netcdf','latitud','longitud','altitud',
            'descripcion','peso_aire','peso_agua','dimensiones','alimentacion',
            'profundidad','payload','mas']}),
        ('Active / Inactive',   {'fields': ['activo']}),
        ('Permissions',   {'fields': ['anonimos']}),
    ]
    inlines = [PermisosInline,SensoresInline]
    list_display=('nombre','tipo','descripcion',)
    list_filter=('tipo',)
    ordering=('nombre',)
    search_fields=('nombre',)
```

Cada uno de los módulos (“cliente”, “usuarios” y “dispositivos”) son directorios donde están contenidos ficheros de los tipos que se acaban de especificar en la propia raíz del directorio salvo las plantillas que se encuentran en una carpeta “templates” dentro de dicho directorio. Con esto se consigue que cada módulo sea autocontenido y reusable con mínimas modificaciones.

Adicionalmente, al mismo nivel que los módulos existe otro directorio llamado “administración” en donde se definen las plantillas comunes de la interfaz de administración. Se encuentran en un directorio aparte porque no es específico de ningún módulo sino que es el encargado del “look” del área administrativo de la aplicación.

A ese mismo nivel también se puede encontrar el directorio “static”, que es un directorio donde sea alojan cada uno de los elementos públicos de la aplicación. Estos elementos no son más que los ficheros CSS, JavaScript, Collada (modelos 3d de los dispositivos) e imágenes.

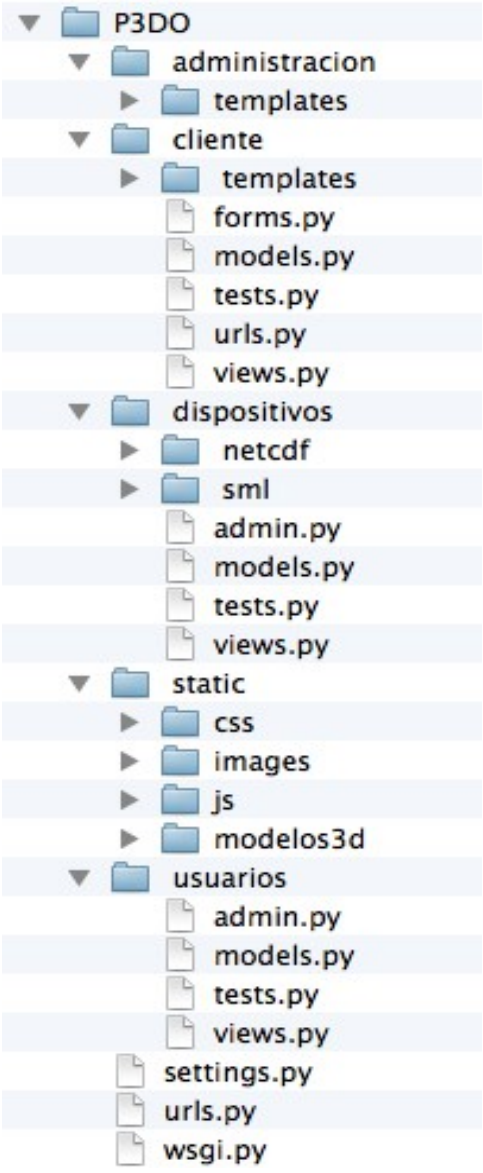


Figura 5.124. Árbol de directorios.

5.6. Pruebas

La última fase dentro del desarrollo de software es la de pruebas. El objetivo de éstas es el realizar un proceso de evaluación donde se comparan los resultados esperados con los obtenidos con la intención de encontrar fallos en el software. Estas pruebas pueden ser automatizadas para facilitar el proceso. Estos tests deberían tener una alta probabilidad de encontrar un fallo, deberían probar si hace lo que debe y no hace lo que no debe y además tienen que ser redundantes. Un proceso de pruebas formal está formado por las siguientes etapas:

- **Planificación de pruebas.** En esta etapa se crea el plan de pruebas donde se determina el alcance de la prueba, los tipos de pruebas a realizar, la estrategia a llevar a cabo, los criterios de salida y otros aspectos a tomar en cuenta.
- **Diseño de pruebas.** Una vez elaborado y aprobado el plan de pruebas en base a la documentación del proyecto existente hasta el momento, se definen los casos de prueba, tanto positivos como negativos, que puedan propiciar la aparición de errores.
- **Implementación y ejecución de pruebas.** Cuando ya se han definido los casos de prueba, se implementan mediante las herramientas que ofrecen los distintos lenguajes de programación y frameworks, y posteriormente se ejecutan obteniendo así los resultados.
- **Evaluación de los criterios de salida.** Con los resultados obtenidos en la etapa anterior se comprueban los resultados de las pruebas y si cumplen con los criterios de éxito de éstos. En caso de fallos, se entra en el proceso de corrección para subsanarlos.

Existen varios niveles de pruebas posibles, que normalmente se dan y se corresponden con puntos del proceso de desarrollo. Esta clasificación es la siguiente:

- **Pruebas unitarias.** Son pruebas ejecutadas durante el desarrollo con las que el desarrollador puede probar que los componentes unitarios cumplen con su propósito en ausencia de errores. Con este tipo de tests se prueban clases individuales o incluso módulos completos.
- **Pruebas de integración.** Son aquellas que se realizan una vez se superan las pruebas unitarias. Con ellas se pretende comprobar la interacción entre distintos componentes del software. Son útiles una vez se integran varios módulos de la aplicación.
- **Pruebas de sistema.** Se deben realizar por personas ajenas al equipo de desarrollo y requiere de una prueba completa del sistema.
- **Pruebas de aceptación.** Son pruebas realizadas por el cliente en un entorno real por posibles usuarios finales. Si el entorno es proporcionado por el equipo de desarrollo son conocidas como “pruebas Alpha”, mientras que si el entorno pertenece al cliente y se ejecuta bajo sus infraestructuras se conocen como “pruebas Beta”.

5.6.1. Testing en Django

Tanto Python como Django, como buen lenguaje y framework que se precie, ofrece soporte para testing de manera nativa y totalmente integrada. En este apartado se hablará de las opciones que ofrece y de la manera de hacerlo. Se explicará cómo se hace en Django ya que es la capa de más alto nivel en la que trabajamos y que a su vez se apoya en Python para hacerlo.

5.6.1.1. Crear el test

Tal y como se vio anteriormente en la organización del código del proyecto, cada módulo de Django contiene un fichero tests.py. En estos ficheros es donde se implementan cada uno de los test como clases Python que heredan de la clase "django.test.TestCase". A la nueva clase se le añaden tantos métodos como funcionalidades se deseen probar donde se ejecutan las acciones que se deseen y posteriormente se comprueba que el resultado es el esperado. Por ejemplo:

```
# -*- encoding: utf-8 -*-

from django.test import TestCase

from P3DO.cliente.models import AccesoAnonimo


class AcessoAnonimoTests(TestCase):

    def test_acceso_anonimo(self):

            acs_anonimo = AccesoAnonimo(anonimos=True)

            self.assertEqual(acs_anonimo.anonimos, True)
```

5.6.1.2. Ejecutar el test

Para ejecutar el test basta con lanzar la siguiente orden desde la raíz del proyecto:

```
python manage.py test cliente
```

Y obtiene una respuesta como la siguiente que nos dice que todo ha ido correctamente:

```
Creating test database for alias 'default'...

.

----------------------------------------------------------------------

Ran 1 test in 0.015s

OK

Destroying test database for alias 'default'...
```

Para que esto haya funcionado ha pasado lo siguiente: “python manage.py test cliente” buscó en la aplicación cliente y encontró una subclase de **django.test.TestCase**, creó una base de datos especial para los tests, buscó los métodos de prueba cuyo nombre comienza por “test_” y lo ejecutó. El assertEqual se encargó de comprobar que los valores que se evaluaban eran iguales.

Si quisiéramos forzar un fallo, podríamos modificar el código de la siguiente manera para que la evaluación fuera errónea.

```
self.assertEqual(acs_anonimo.anonimos, False)
```

Y volviendo a ejecutar:

```
Creating test database for alias 'default'...

======================================================================

FAIL: test_acceso_anonimo (P3DO.cliente.tests.AcessoAnonimoTests)

----------------------------------------------------------------------

Traceback (most recent call last):

  File "C:\Users\debon\Desktop\p3do\P3DO\cliente\tests.py", line 9, in test_acceso_anonimo

    self.assertEqual(acs_anonimo.anonimos, False)

AssertionError: True != False



----------------------------------------------------------------------

Ran 1 test in 0.011s



FAILED (failures=1)

Destroying test database for alias 'default'...
```

Se pueden crear tantas clases y métodos como se deseen, cuantas más mejor.

5.6.1.3. Tests para Views

Django provee un cliente para test, **Client**, para simular la interacción del usuario con el código a nivel view. Podemos usarlo en tests.py o incluso en el shell. Client también es una clase de django.test y funciona simulando peticiones HTTP que ejecutaría un navegador web cuando un usuario interacciona con la página.

La forma de crear el test es exactamente la misma que en el caso anterior. El siguiente código muestra un ejemplo muy simple de test de un controlador. En él se comprueba dos URL, una que existe y una que no.

```
# -*- encoding: utf-8 -*-

from django.test import TestCase

from django.test.client import Client

from P3DO.cliente.models import AccesoAnonimo


class ClienteViewsTest(TestCase):

     def test_view_cliente(self):

             cliente = Client()


             response = cliente.get('/')

             self.assertEqual(response.status_code,200)


             response = cliente.get('/rutaquenoexiste')

             self.assertEqual(response.status_code,404)
```

Y el resultado de la ejecución del test sin errores.

```
Creating test database for alias 'default'...

.

----------------------------------------------------------------------

Ran 1 test in 0.259s


OK

Destroying test database for alias 'default'...
```

Capítulo 6: Conclusiones y Trabajo Futuro

6.1. Conclusiones

Tras un largo camino, muchas reuniones, una gran cantidad de diagramas, líneas de código y muchas horas se finaliza el proyecto 3DO. Es ahora, y no antes, el momento para analizar y valorar si se han alcanzado los objetivos marcados al comienzo tanto como producto, como proceso de creación de software.

Desde el punto de vista del proceso, he tenido la suerte de poder realizar un proyecto con un "pseudocliente" real, con sus requisitos, sus infraestructuras reales, sus organigramas y con una necesidad existente que tenían que cubrir y solucionar. Esta parte le da una riqueza como proyecto fin de carrera porque todo se ha fraguado en un entorno real, un entorno como al que me tengo que enfrentar en mi etapa profesional, donde los clientes tienen sus exigencias, donde los plazos toman importancia, donde el resultado debe estar libre de errores porque el trabajo de otros profesionales dependen de él, y donde los recursos hardware, software y humanos son limitados porque los recursos económicos marcan el tope de la inversión que se puede hacer en el proyecto.

Con este escenario, ha sido vital desde un primer momento una excelente comunicación entre los participantes en el proyecto, donde existían a lo largo de todo el proceso reuniones periódicas donde se marcaban objetivos y se establecían plazos, y cómo no, se debían negociar para que intereses contrapuestos no se vieran perjudicados. Creo que estas reuniones, y lo que se aprenden de ellas, son cosas que no se pueden adquirir durante la carrera. Es necesario que hayan partes con intereses encontrados y donde el aspecto económico sea importante para que surjan ciertas confrontaciones que se deben resolver mediante la negociación, ya sean requisitos deseables, plazos o asuntos de cualquier otra índole.

El afrontar un proyecto desde cero también te da otra visión más global del proceso de creación de software. Permite unificar conocimientos de materias dispares con el único propósito de buscar la mejor solución a un problema planteado. Durante este proceso se ha tenido que aplicar las aptitudes y conocimientos adquiridos en materias de ingeniería del software, bases de datos, redes, programación, metodologías de programación, seguridad, administración de sistemas operativos, etc.

Probablemente la tarea mas difícil a la que me he tenido que enfrentar durante este proyecto es a la de planificación. Me he dado cuenta que la inexperiencia hace de esta labor que sea prácticamente imposible acertar en el tiempo estimado para llevar algo a cabo. Pero a su vez he comprobado que según iba avanzando en el proyecto, era capaz de ir ajustando cada vez más la previsión de tiempo que me llevaría enfrentarme a una tarea.

Al haber desarrollado el proyecto mediante el Proceso Unificado de Desarrollo, y ser una proceso iterativo e incremental, me ha permitido ir obteniendo pequeños resultados independientes, hecho que ayuda especialmente a marcar objetivos a corto plazo y mejorar la motivación. Ésto es algo especialmente importante en un proyecto de larga duración.

Por otro lado, desde el punto de vista del producto, se ha obtenido una aplicación totalmente funcional, usable, testada y lista para ponerla en marcha si se desea. Así que por el lado del software se han cumplido todos y cada uno de los objetivos marcados al comienzo de este proyecto.

Como valor añadido, se han usado infinidad de herramientas y tecnologías, y además todas ellas bajo licencias abiertas. Para ello ha sido necesario un largo proceso de aprendizaje tanto de lenguaje, como frameworks u otras herramientas. El requisito de que la aplicación fuese en Python y sobre Django fue una propuesta de PLOCAN ya que sus aplicaciones internas están desarrolladas bajo estas tecnologías y les parecía un punto a favor mantener la homogeneidad de sus sistemas. Hasta ese momento desconocía ambos pero me pareció una oportunidad de aprender un nuevo lenguaje y framework que estaban siendo usados en una organización con un buen departamento TIC y era una forma más de acercarme a los perfiles que se demandan profesionalmente.

Para seguir con el propósito de hacer una herramienta haciendo uso de software libre, se dedicaron muchas horas a la investigación de librerías, plugins, frameworks, paquetes de imágenes e iconos, etc, que cumplieran al cien por cien con este requisito. Y en este proceso de búsqueda me dí cuenta de que existe una gran cantidad de herramientas y software gratuitos que permite desarrollar infinidad de soluciones sin la necesidad de invertir demasiado en licencias y otros aspectos que encarecerían el proyecto.

En conclusión, y en base a lo argumentado anteriormente, creo que se ha desarrollado una herramienta que cubre una necesidad real en una organización real resolviendo exitosamente los requisitos que se marcaron durante la definición del proyecto. Eso me produce una doble satisfacción, por un lado la de la culminación de mi proceso de formación universitaria que concluye una de las etapas más importantes e intensas de mi vida; y por otro lado, la de comprobar que he sido capaz de proporcionar una herramienta útil para personas que puede mejorar y facilitar su trabajo resolviendo una necesidad que no había sido cubierta antes de la creación de este software.

6.2. Trabajos Futuros

El presente proyecto ha desembocado en una aplicación completamente funcional y preparada para hacer uso de ella. Aunque se han abordado los principales requisitos que se necesitaban, por la limitación temporal y de recursos, han quedado propuestas pero no abordadas otras funcionalidades que podría implementarse de cara al futuro. Como se puede observar al comienzo del capítulo 5 del presente documento, en la lista de características, existen algunas de ellas que aparecían como requisitos deseables pero que no fueron aceptadas en el proyecto actual.

Algunas de estas características se pasan a enumerar a continuación como posibles candidatas a posteriores ampliaciones del proyecto:

- Visualización de vídeos online tomados desde los dispositivos.
- Superposición de imágenes reales tomadas desde los dispositivos sobre el fondo marino del globo.
- Obtención de las medidas de los sensores en tiempo real mediante conexión por satélite.
- Extracción de datos oceanográficos mediante protocolo OGC – SOS – O&M.
- Gestión de eventos que se ejecuten cuando se den ciertas condiciones oceánicas y atmosféricas basadas en los datos obtenidos por las plataformas de observación.
- Creación de web services para permitir conexión desde otra aplicaciones o servicios.
- Integración de contenidos lúdicos con Infinity 3D.
- Creación de mundo virtual interactivo con Unity 3D.

Anexo

Anexo A: Manual de Usuario

El presente manual de usuario pretende presentar la aplicación 3DO al lector y mostrarle las distintas funcionalidades que ofrece y las instrucciones para su correcto uso.

1. ¿Qué es 3DO?

3DO es un observatorio oceánico virtual accesible vía web desde cualquier ordenador con navegador y conexión a Internet. Permite que una organización, en este caso la Plataforma Oceánica de Canarias, administre una serie de dispositivos y sensores, con su información y metainformación, para ser publicado y que cualquier persona interesada pueda acceder de manera remota para conocer su actividad e incluso ver los datos recabados en sus expediciones.

3DO también ofrece la posibilidad a la organización de decidir que datos son publicados o quién puede verlos. Para ello existe un módulo de administración de usuarios y grupos donde se definen los perfiles con los distintos permisos sobre la aplicación.

3DO está pensado tanto para usuarios expertos en la materia como personal de la propia organización y científicos externos a ella como para usuarios sin conocimientos oceanográficos y cuyo único interés sea meramente curiosidad. Esto es así porque la aplicación tiene también un perfil divulgativo donde mediante recursos gráficos e interactivos, como son los globos terráqueos virtuales, se pretende captar la atención para acercar la ciencia a la sociedad en general.

3DO puede funcionar como herramienta docente para colegios, institutos e incluso Universidad para de una forma lúdica y llamativa despertar el interés de los alumnos en las ciencias del mar y dar a conocer las actividades que se llevan a cabo en los océanos por la comunidad científica.

2. Requisitos necesarios

- Ordenador con sistema operativo Windows o Mac Os.
- Conexión a Internet.
- Navegador Web.

3. La Pantalla Principal

A continuación se muestra la pantalla principal de 3DO. Esta pantalla esta divida en 3 secciones bien diferenciadas:

- Área de inicio de sesión de usuario.
- Menú principal, donde se puede elegir entre ver la plataforma, los dispositivos o los sensores. Tanto la opción "Devices" como "Sensors" son formas distintas de llegar a un mismo sitio. En ambos casos el objetivo final es mostrar un dispositivo en el mapa 3D o su ventana de información. La diferencia es que en el primer caso se accede desde el propio dispositivo y en el segundo al dispositivo que contiene el sensor seleccionado.
- Menú secundario, donde se encuentra los botones de acción.
- Globo terráqueo virtual navegable e interactivo, donde se muestran los dispositivos.

Figura A.1. Pantalla principal.

4. Moverse por el Globo Terráqueo

Para moverse por el globo terráqueo basta con tenerlo activo haciendo click en él y haciendo cualquier de las siguientes acciones:

- **Mantener Click + Mover Ratón →** Mover la esfera terrestre de manera natural.
- **Girar rueda de scroll →** Acercar o alejar el zoom.
- **Ctrl + Scroll →** Girar en el mismo plano (cambia de dirección Norte, Sur, Este, Oeste)
- **Mayus + Scroll →** Cambiar la altitud del punto de vista.

5. Ver Plataforma Oceánica

Para ver la plataforma oceánica se debe seleccionar en el menú principal la opción "Offshore Platform" y hacer click en el bottom "Show" del menú secundario.

Figura A.2. Ver Plataforma Oceánica.

6. Ver Dispositivos y Sensores

Para ver un dispositivo y sus correspondientes sensores en 3DO existen varias formas de hacerlo. Por un lado se pueden localizar navegando en el globo terráqueo donde se podrán observar distribuidos por el mundo. La forma de localizarlos es mediante logos de PLOCAN situados a lo largo y ancho del Océano.

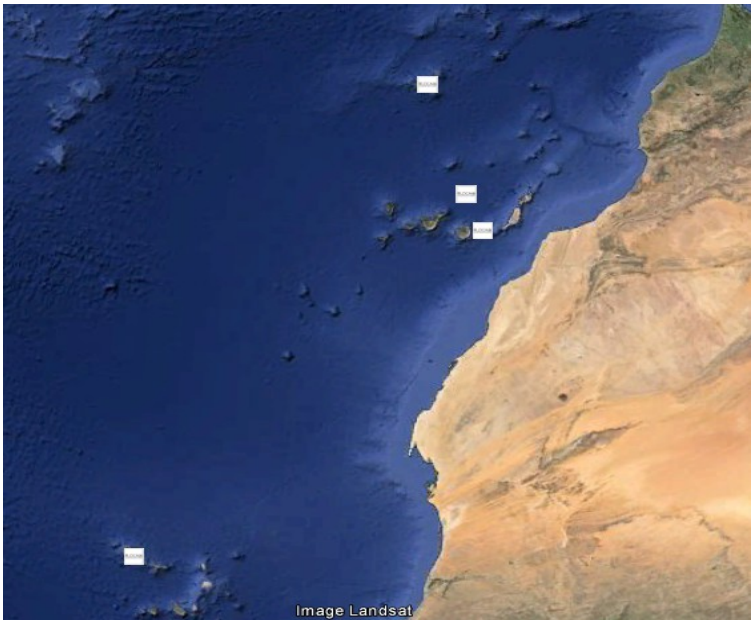


Figura A.3. Dispositivos en el globo terráqueo.

6.1. Ver modelo 3D del dispositivo

Para ver el modelo 3D de un dispositivo se pueden seleccionar cualquiera de las siguientes opciones:

- Navegar por el globo acercando el zoom hasta que vea el dispositivo.
- Hacer Click con el botón derecho del ratón en su icono encontrado en el mapa.
- Acceder a “Devices” o “Sensors” en el menú principal, seleccionar el tipo de dispositivo o sensor según corresponda y clickar sobre el icono que acompaña al dispositivo.

Figura A.3. Dispositivos en el globo terráqueo.

6.2. Ver Información de Dispositivos

Para ver la información de un dispositivo, los sensores que lo componen y los datos recabados por ellos, existen dos vías posibles. Una a través del globo terráqueo y otra a través del menú.

Para acceder desde el mapa basta con hacer un click con el botón izquierdo del ratón sobre el logo del dispositivo que aparece en el mapa 3D.

Para acceder desde el menú hay que seleccionar "Devices" o "Sensors" en el menú principal, seleccionar el tipo de dispositivo o sensor en el menú secundario y hacer click sobre el botón que aparece junto al dispositivo o sensor seleccionado.

Entonces aparece la ventana de información de dispositivos, donde existen tres secciones. Una primera sección con la información general del dispositivo. Una segunda donde se obtienen los datos recabados por él. Y una tercera opción donde aparecen los sensores que lo conforman y la información relativa a ellos.

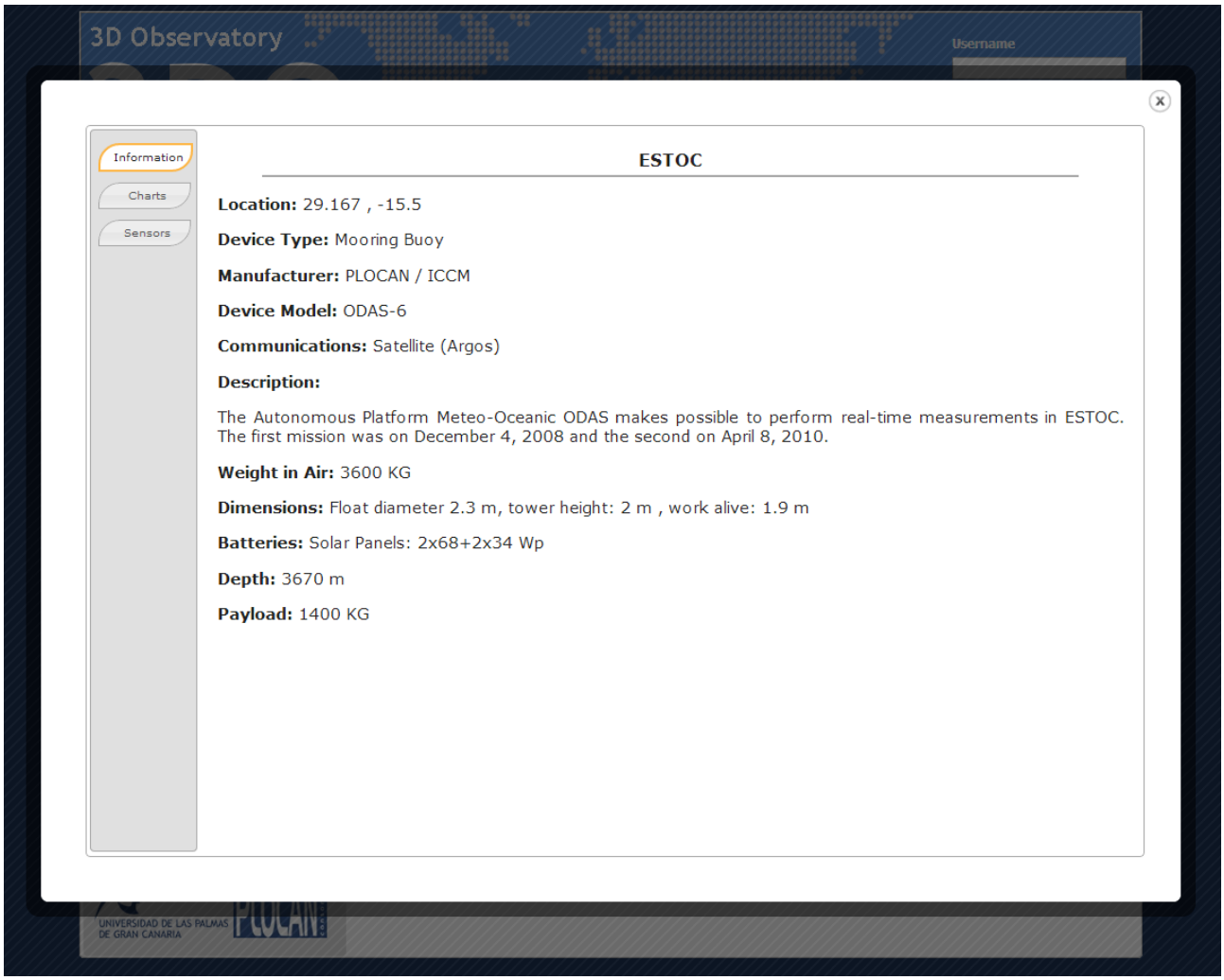


Figura A.4. Ventana de información de dispositivo. Pestaña información general.

Para obtener los datos recabados se debe hacer click sobre la pestaña “Charts” donde aparece un formulario donde se deben seleccionar las magnitudes posibles.

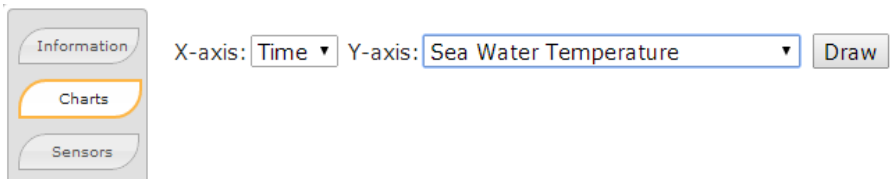


Figura A.5. Formulario de generación de gráfica de datos.

Una vez seleccionadas las magnitudes deseadas se le presiona el botón “Draw” generándose así una gráfica navegable. Se puede seleccionar los periodos de muestreos moviendo el área seleccionada en parte inferior de la gráfica.

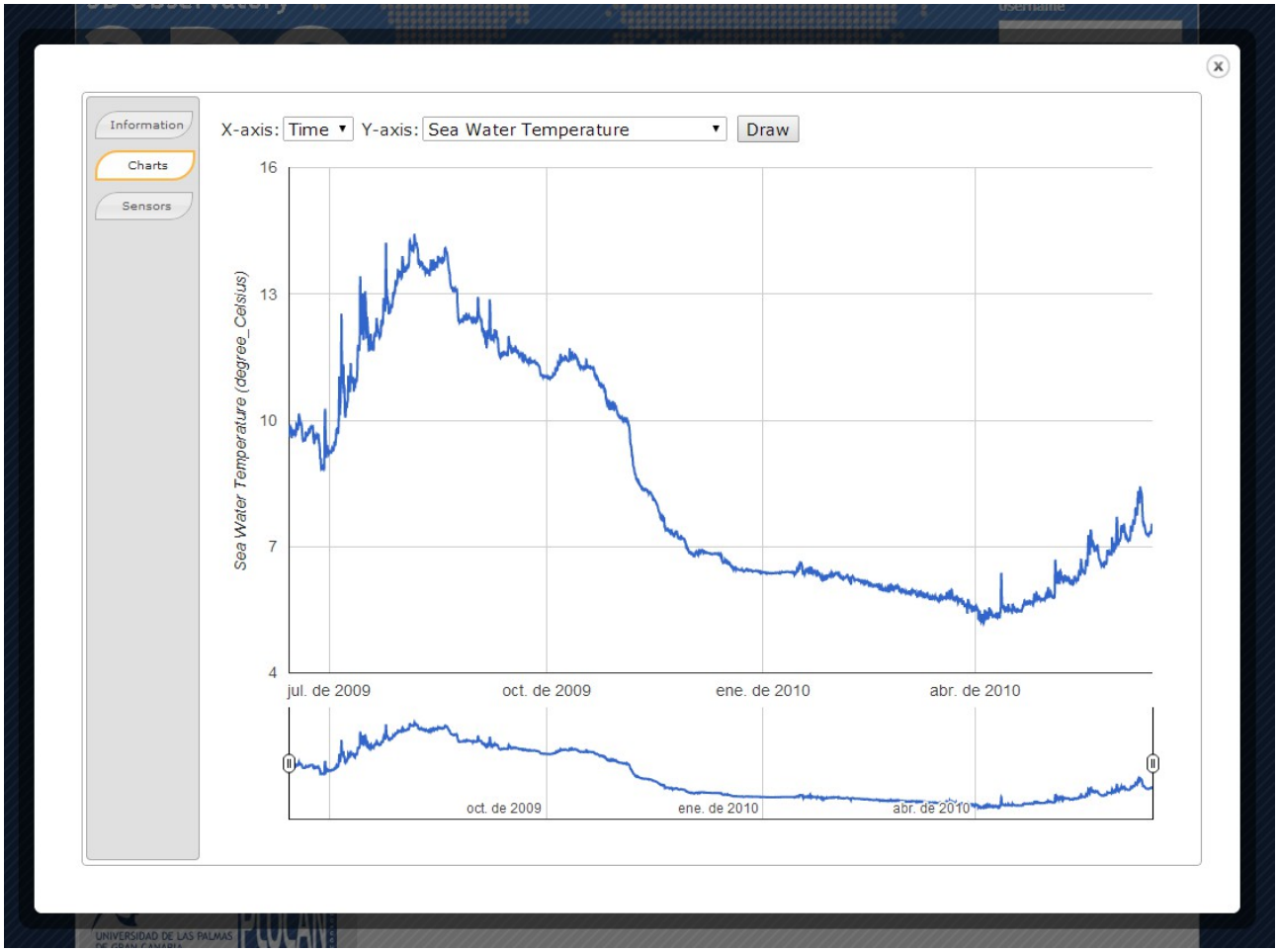


Figura A.6. Gráfica de datos.

Por último en la pestaña “Sensors” se muestran cada uno de los sensores montados en el dispositivo e información relativo a ellos.

7. Iniciar Sesión de Usuario

Para iniciar una sesión de usuario basta con instroducir los credenciales del usuario en el área de usuarios y presionar el botón “Login”. Dependiendo de los permisos del usuario se tendrá acceso a ciertas partes de la aplicación.

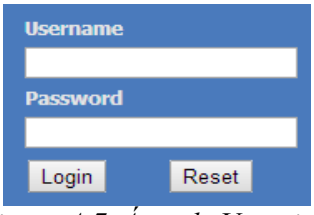


Figura A.7. Área de Usuarios.

8. Área de Administración

El área de administración es donde los usuarios con permisos de administrador gestionan la aplicación. Dependiendo de los permisos que tenga un usuario podrá ver todos los módulos o sólo a los que tenga acceso.

Figura A.8. Pantalla de administración.

La pantalla de administración está dividida en tres partes. Una primera de funciones generales donde se encuentra el icono para volver a la pantalla principal y las opciones para cerrar sesión y cambiar contraseña. Otra para la gestión de usuarios y accesos al sistema. Y por último, la de gestión completa de los dispositivos.

8.1. Acceder a Área de Administración

Para acceder a la interfaz de administración, una vez logueado un usuario en el sistema, siempre y cuando tenga los permisos necesarios, basta con presionar el botón "Admin" que aparece en pantalla principal de la aplicación.

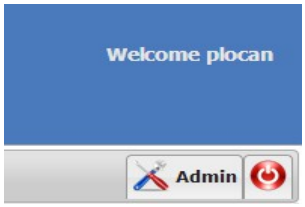


Figura A.9. Acceso Admin.

8.2. Cambiar Contraseña

Para modificar la contraseña hay que acceder en el menú general de la interfaz de administración a la opción "Change Password". Una vez ahí se debe introducir la contraseña antigua y la nueva por duplicado.

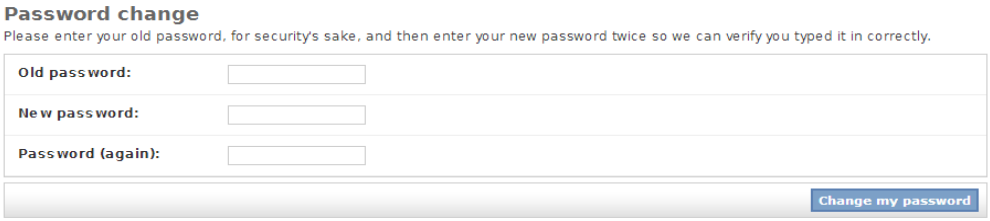


Figura A.10. Pantalla de cambio de contraseña.

8.3. Abrir/Cerrar Acceso a Usuarios Anónimos.

Para abrir o cerrar el acceso a la aplicación a usuarios anónimos que no se hayan logueado en el sistema sólo hay que abrir o cerrar el cerrojo presionando en él.

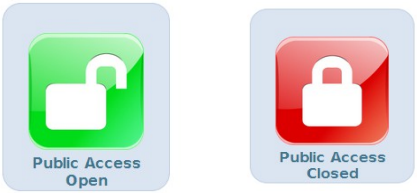


Figura A.11. Abrir/Cerrar Acceso.

8.4. Administración de Usuarios

En la administración de usuarios es importante destacar ciertos aspectos de relevancia. Cuando se crea o se edita un usuario se puede observar información de diferente naturaleza. Por un lado el nombre de usuario y la contraseña que serán los credenciales de la cuenta de usuario. Además contiene la información personal del usuario y el grupo, que no es más que un perfil con un conjunto de permisos.

El siguiente bloque es importante porque define la relación del usuario con la organización. Si se marca la opción "Superuser status" convertimos al usuario en un super usuarios con todos los permisos. La opción "Staff status" marca al usuario como usuario de personal interno de la organización y por lo tanto con acceso a la interfaz de administración aunque dependiendo del grupo verá unas categorías u otras. Y por último la opción "Active" sirve para activar o desactivar una cuenta de usuario sin la necesidad de ser borrada.

El último bloque se corresponde con permisos individuales sobre dispositivos donde para dispositivos que no se encuentre publicados para todos los usuarios, se le puede dar acceso a usuarios independientes.

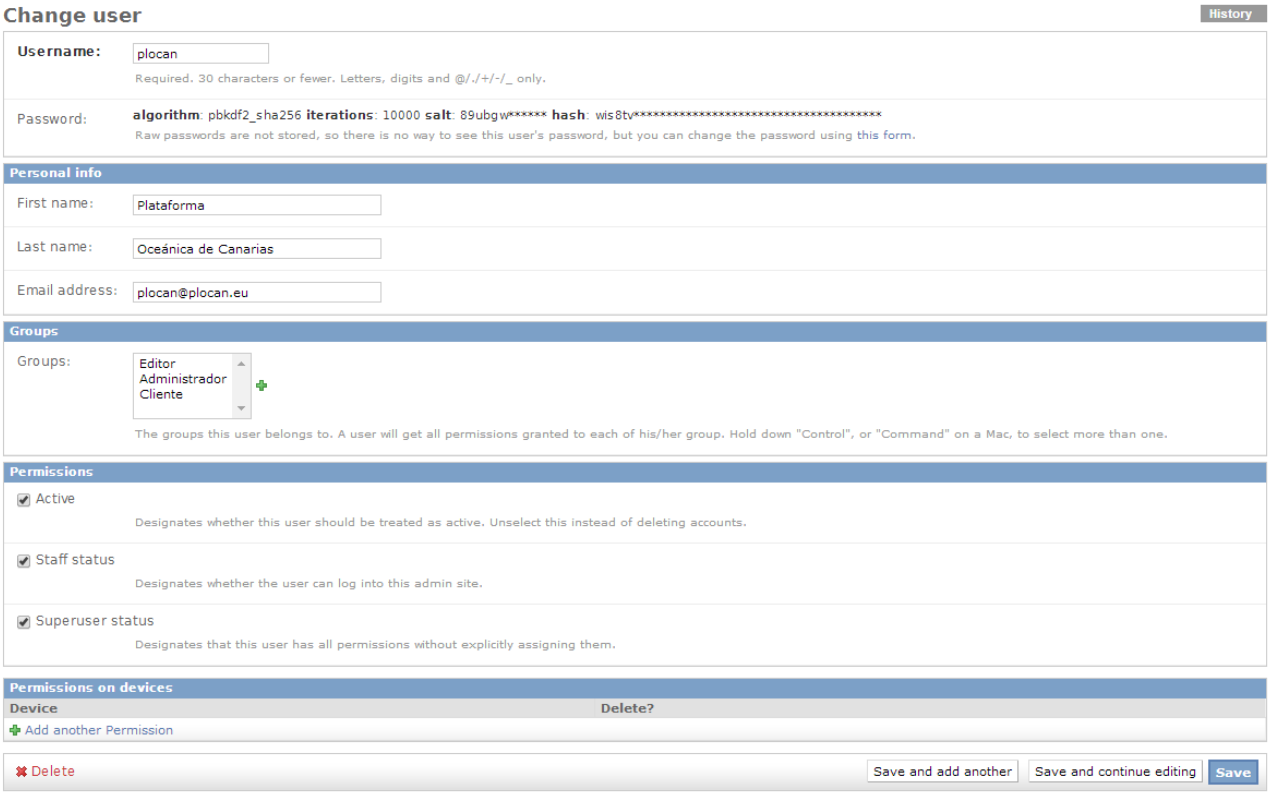


Figura A.12. Pantalla de edición de usuarios.

8.5. Administración de Dispositivos

La administración de dispositivos no incluye únicamente los dispositivos propiamente dicho sino que incluye también los tipos de dispositivos, los sensores, los tipos de sensores y los modelos 3D de los dispositivos.

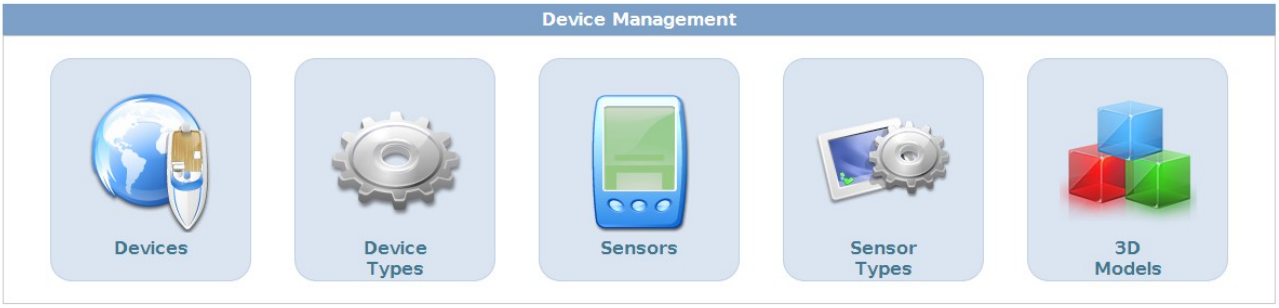


Figura A.13. Sección de administración de dispositivos.

En cada una de las secciones se puede obtener un listado, crear, eliminar y editar en lo relativo a la propia sección. Cuando existen relaciones entre las distintas secciones se pueden crear y modificar desde ambas secciones. Por ejemplo, un dispositivo contiene sensores. Pues desde la pantalla de edición de un dispositivo se puede añadir o eliminar un sensor. Y también cuando se crea o se modifica un sensor se le puede especificar a que dispositivo pertenece.

8.6. Acciones globales

Existe ciertas acciones que son comunes a cualquier categoría de la pantalla de administración. Estas acciones son la creación, modificación, eliminación y listado. Para no caer en la repetición se explicará de manera global y no para cada categoría independientemente.

8.6.1. Listar

Para listar los elementos de una categoría hay que acceder a la categoría deseada mediante el botón correspondiente. Dentro del listado pueden existir filtros de búsquedas y opciones por defecto.

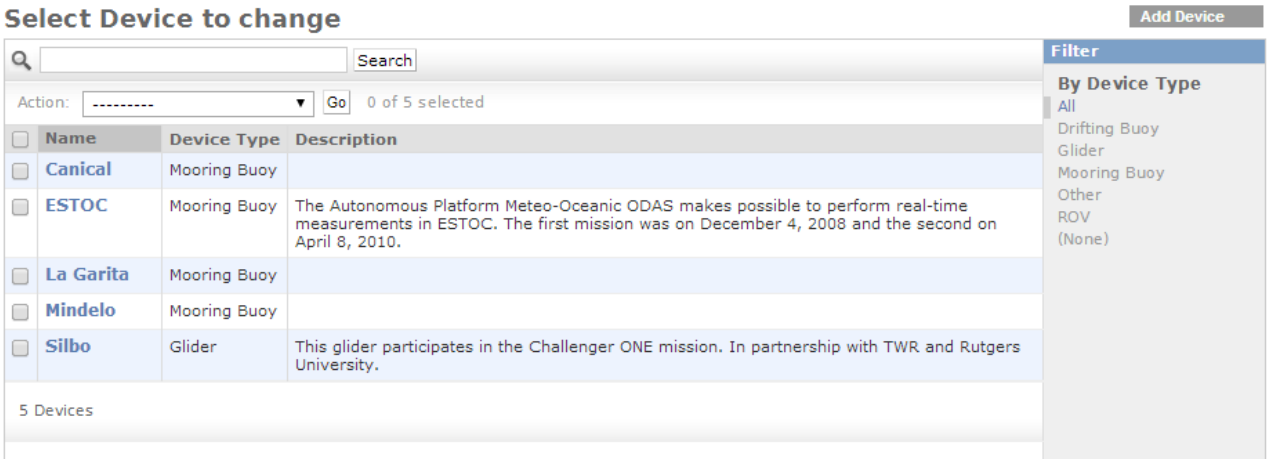


Figura A.14. Pantalla de listado.

8.6.2. Crear

Para crear un elemento de una categoría, desde la pantalla de listado se pulsa sobre el botón "Add ...".

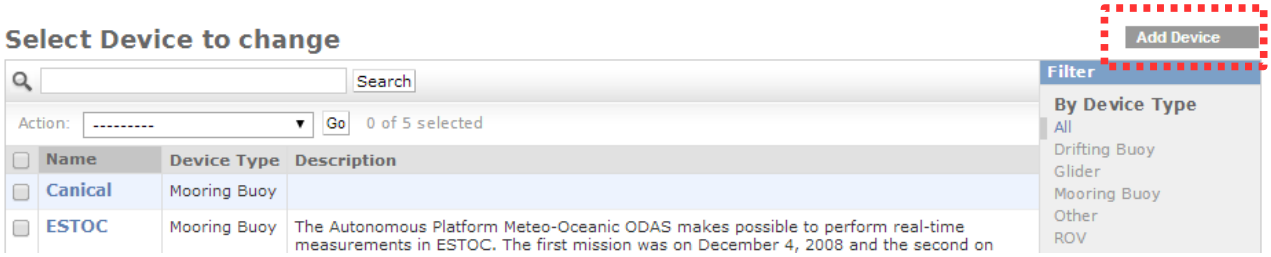


Figura A.15. Crear elemento.

8.6.3. Eliminar

Para eliminar un elemento de una categoría, desde la pantalla de listado se marcan los elementos que se desean eliminar, se selecciona la opción "Delete selected ..." y se pulsa en el botón "Go"

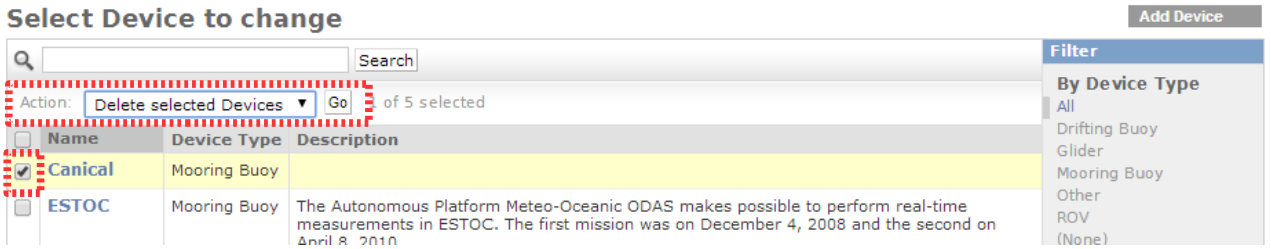


Figura A.16. Eliminar elemento.

8.6.4. Editar

Para editar un elemento de una categoría, desde la pantalla de listado se hace click sobre el nombre del elemento que se desea editar.

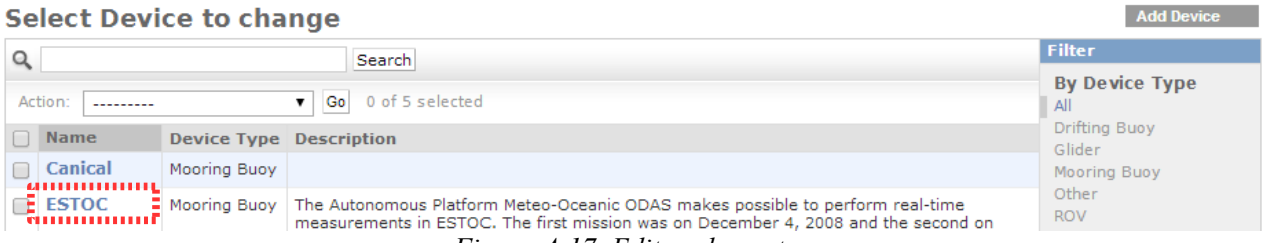


Figura A.17. Editar elemento.

8.6.5. Pantalla de Creación/Edición

La pantalla de creación y de edición de un elemento prácticamente es la misma salvo que en el caso de edición los campos están rellenos por defecto. Este tipo de pantalla consta de los campos agrupados por bloques semánticos y al final el menú de botones donde se puede eliminar el elemento, guardarlo y salir al listado o guardarlo y seguir editando.

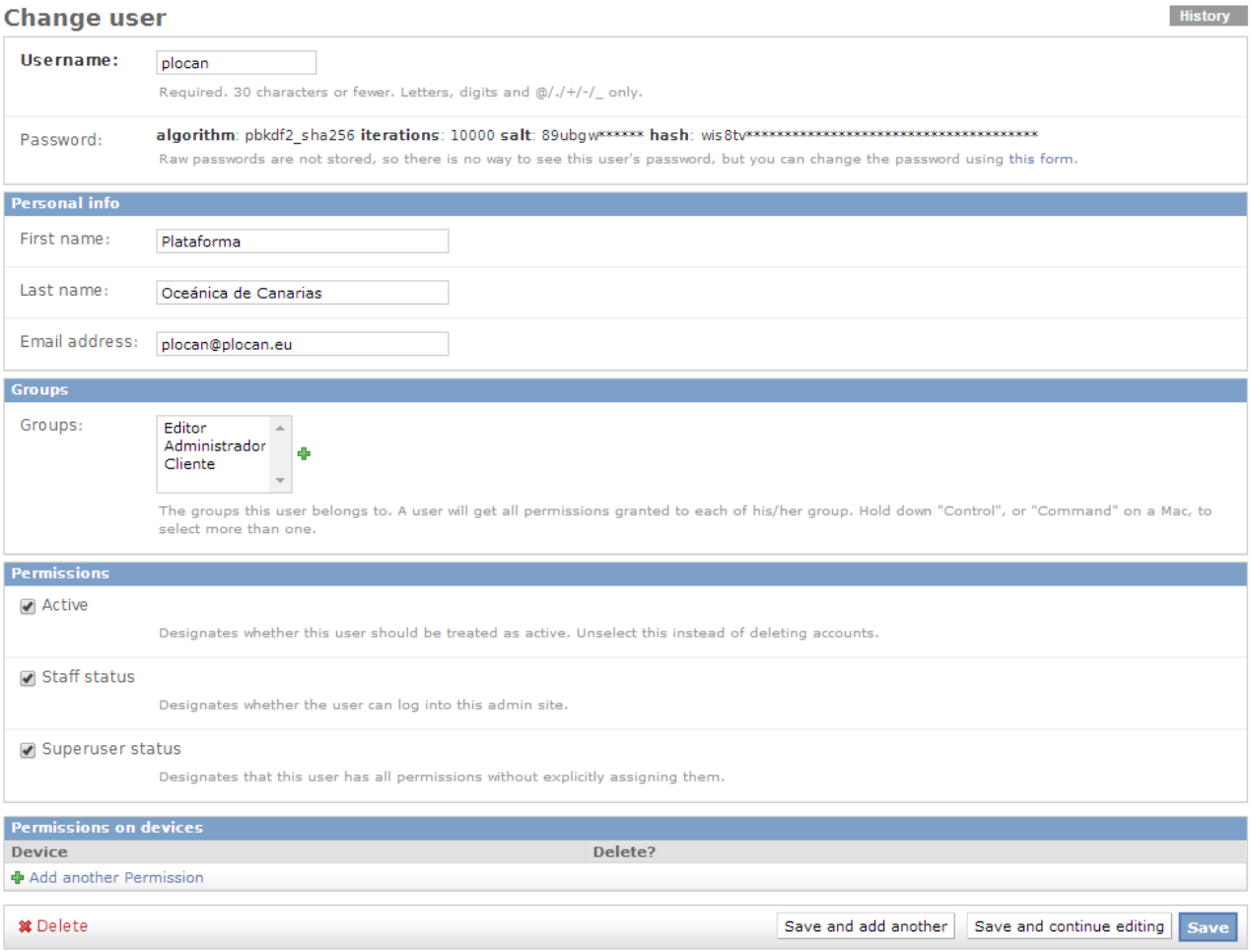


Figura A.18. Pantalla creación/edición.

Anexo B: Comparativa de APIs de Generación de Gráficas

Una de las partes importante de la aplicación es la forma en la que se muestran los datos científicos al usuario. Para que resulte más atractivo para los usuarios no especialistas, ya que se pretende la divulgación y acercar la ciencia a todos, los datos serán mostrados en forma de gráficas.

Por esta razón, en este anexo se darán a conocer varias herramientas que sirven para la generación dinámica de gráficos a partir de datos obtenidos de bases de datos o ficheros de datos. Además como requisito es indispensable que estén pensadas e implementadas en tecnologías web. Aún así el número de herramientas, APIs y plugins de este tipo es altísimo, por lo que se enumerarán los mas relevantes.

1. Chart.js

Chart.js es una librería JavaScript con gráficos orientados a objetos del lado del cliente que hace uso del elemento canvas de HTML5. Actualmente soporta 6 tipos de gráficos (líneas, barras, radiales, circulares, áreas polares y de anillos). Cada uno de ellos son animados, totalmente personalizables y visibles incluso en pantallas de retina.

Figura B.1. Ejemplos Chart.js

2. Google Charts Tools

Google Chart es una API desarrollada por Google que crea de forma dinámica gráficos y tablas para incrustar fácilmente en páginas web mediante una petición HTTP a la API. Google ofrece una gran cantidad de tipos de gráficos para su uso. La apariencia se puede personalizar para adaptarse al estilo del sitio web. Los gráficos son muy interactivos y soporta eventos que se pueden entrelazar para crear cuadros de mando complejos. Mientras que la versión original se basó en las imágenes, la mejora de la API hizo que los gráficos se representan mediante la tecnología HTML5/SVG para proporcionar compatibilidad entre navegadores (incluyendo VML para las versiones anteriores de IE) y la portabilidad entre plataformas para iOS y Android sin necesidad de plugin.

Todos los tipos de gráficos se rellenan con los datos utilizando la clase DataTable, lo que hace que sea fácil cambiar entre los tipos de gráficos según las necesidades. El DataTable proporciona métodos para la clasificación, modificación y filtrado de datos, y se puede llenar directamente desde una página web, una base de datos, o cualquier proveedor de datos que soporte el protocolo Chart Tools Datasource. Este protocolo incluye un lenguaje de consulta similar a SQL y es implementado por Google Spreadsheets, Google Fusion Tables, y los proveedores de datos de terceros, como SalesForce.

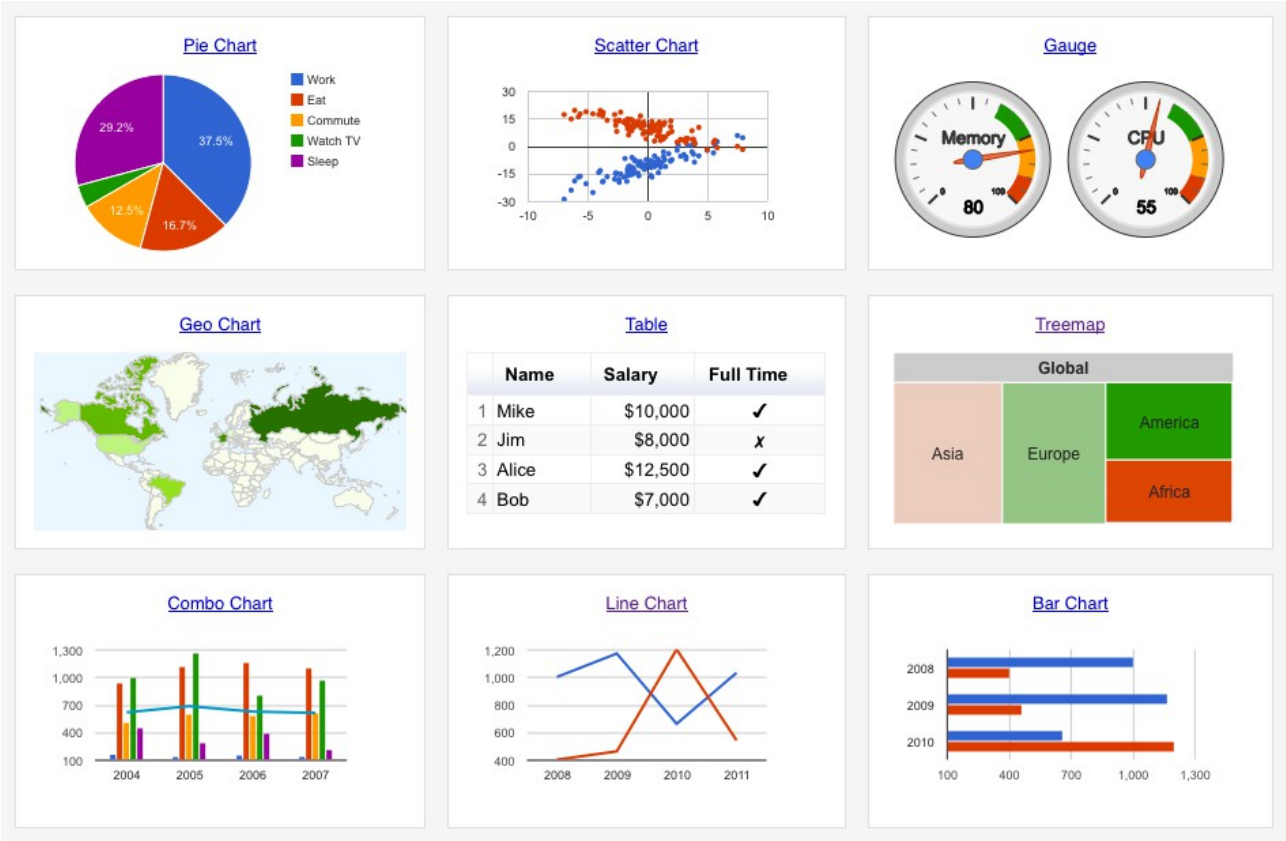


Figura B.2. Ejemplos Google Charts Tools

3. Highcharts

Highcharts es una biblioteca de gráficos escrita en JavaScript puro, que ofrece una forma fácil de añadir gráficos interactivos a una web. A través de una API completa, puede añadir, eliminar y modificar series y puntos o modificar los ejes en cualquier momento después de la creación gráfica.

Actualmente soporta gráficos de línea, área, columnas, barras, circulares, de dispersión, patrones angulares, rangos, burbuja, diagrama de caja, barras de error, tipos de gráficos polares, etc.

Es gratuito para uso no comercial y de pago para uso comercial con diferentes configuraciones de licencias. Con cualquiera de las licencias, gratuita o de pago, se permite descargar el código fuente y hacer modificaciones. Esto permite adaptaciones personalizadas y una gran flexibilidad.

Figura B.3.. Ejemplos Highchart

4. Morris.js

Morris.js es una biblioteca ligera que utiliza jQuery y Raphaël (otra librería gráfica javascript básica) para dibujar fácilmente gráficos de series temporales en aplicaciones web. Morris.js comenzó como herramienta base que alimentaba los gráficos en howmanyleft.co.uk. Posteriormente se convirtió en una librería de código abierto licenciada bajo licencia BSD.

Funciona con todos los navegadores modernos. La API pública es muy simple, limitándose a una única función: Morris.Line(opciones), donde las opciones son un objeto que contiene todas opciones de configuración. Soporta gráficas lineales, de barras, de área y de anillo.

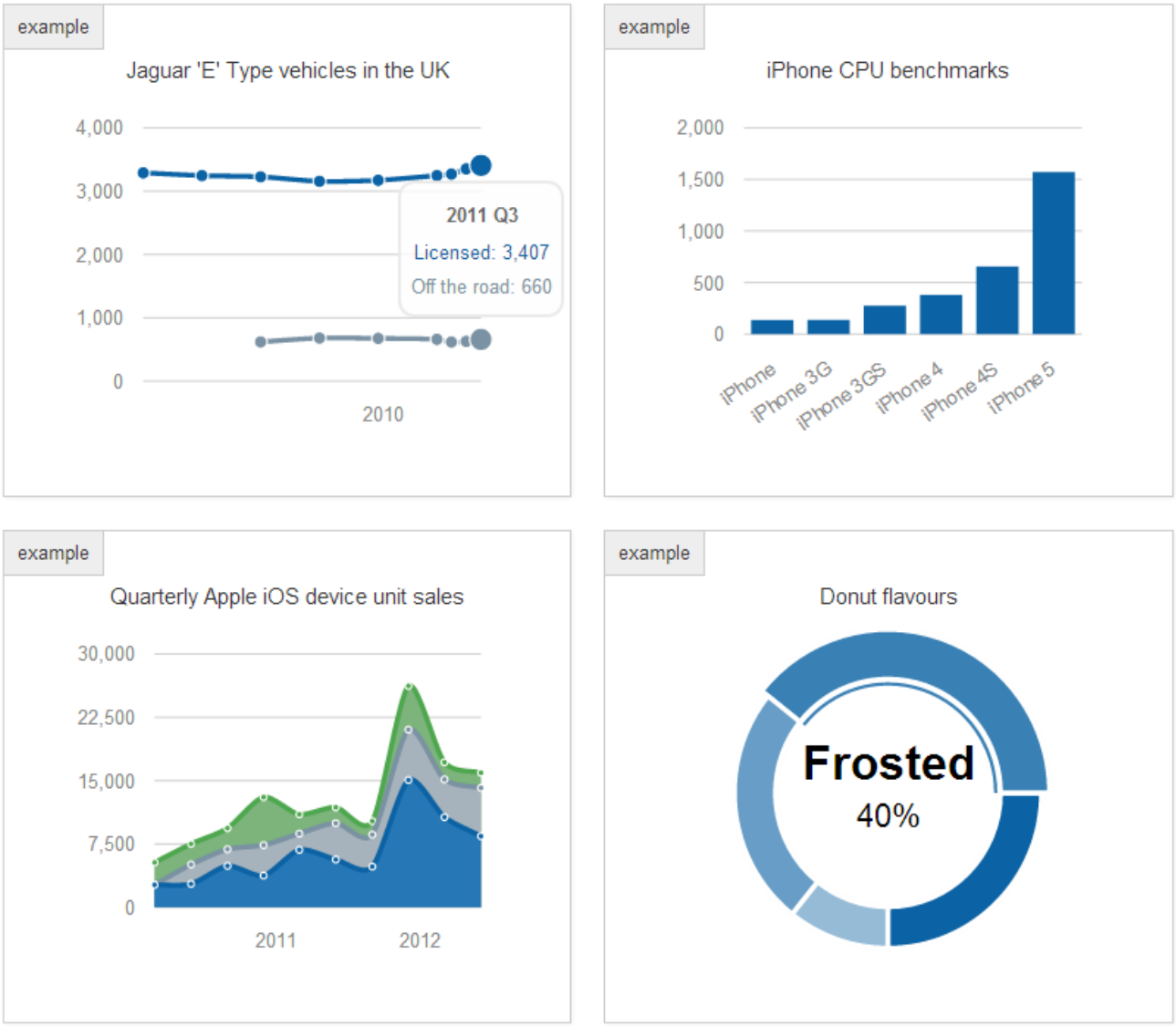


Figura B.4. Ejemplos Morris.js

5. jqPlot

jqPlot es una librería escrita en JQuery y Javascript que utiliza el elemento Canvas para representar del lado del cliente gráficos dinámicos.

Entre las mejores características de este plugin están: una extensa gama de gráficos a implementar, formato personalizado en coordenadas, gráficos de hasta 9 ejes, rotación de texto, escalado automático, cálculo automático de la tendencia de la línea y resaltado de datos.

jqPlot es un software gratuito y de código abierto doblemente licenciado bajo las licencias MIT y GPL2. Cada uno es libre de elegir la licencia que mejor se adapte a su proyecto.

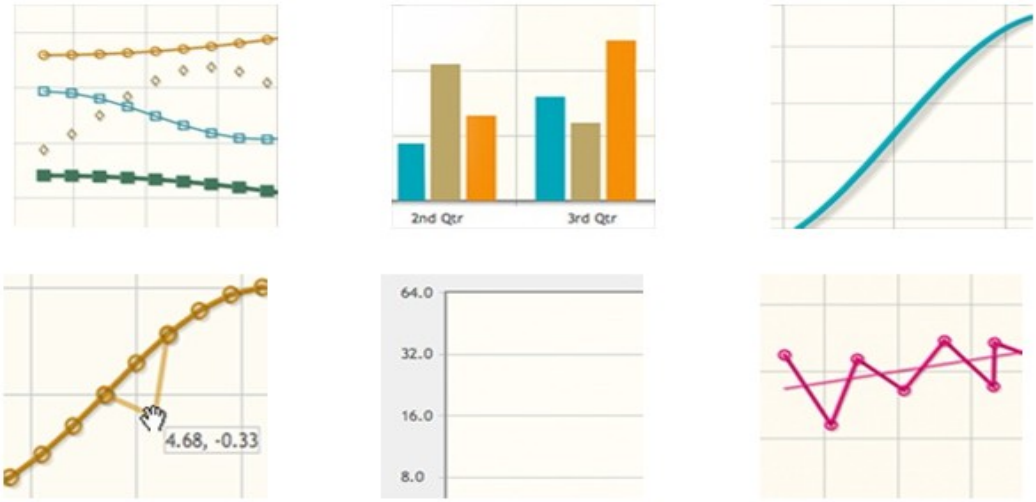


Figura B.5. Ejemplos jqPlot.js

Bibliografía

[1] Jacobson, Ivar; Booch, Grady; Rumbaugh, James. El Proceso Unificado de Desarrollo de Software. Addison Wesley, 2000.

[2] Larman, Craig. UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado. Prentice Hall, 2003

[3] Holovaty, Adrian. La guía definitiva de Django. Anaya Multimedia, cop. 2010.

[4] Martelli, Alex. Python: La guía de referencia. Anaya Multimedia, 2007

[5] ABIZTAR. Base de conocimiento de UML y arquitecturas. Milestone, 2014. http://www.milestone.com.mx/BaseConocimientoUML.htm

[6] Castro Jiménez, Eliseo. UML, Unified Modeling Language. Slideshare, 2014. http://www.slideshare.net/ecastrojimenez/uml-lenguaje-de-modelamiento-unificado-presentation

[7] Conallen, Jim. Modeling web application architectures with UML. Rational Software, 1999. http://interval.cz/podklady/1999-2008/zelenka/695/27662_webapps.pdf

[8] Python Software Foundation. Python documentation. Python Software Foundation, 2014. https://docs.python.org/2/

[9] Django Software Foundation. Django documentation. Media Temple, 2014. https://docs.djangoproject.com/en/1.5/

[10] Refsnes Data. HTML Tutorial. W3Schools, 2014. http://www.w3schools.com/html/default.asp.

[11] Refsnes Data. CSS Tutorial. W3Schools, 2014. http://www.w3schools.com/css/default.asp

[12] Refsnes Data. JavaScript Tutorial. W3Schools, 2014. http://www.w3schools.com/js/default.asp

[13] The jQuery Foundation. Jquery API . Media Temple, 2014. http://api.jquery.com

[14] The jQuery Foundation. jQuery UI API. Media Temple, 2014. http://api.jqueryui.com

[15] Moore, Jack. Colorbox, a jQuery lightbox. Jack L. Moore, 2014. http://www.jacklmoore.com/colorbox/

[16] NOAA. NetCDF Python. NOAA, 2014. http://gfesuite.noaa.gov/developer/netCDFPythonInterface.html

[17] Lancaster, Kyle. SimpleKML Python. Read The Docs, 2013.
http://simplekml.readthedocs.org/en/latest/

[18] Blech, Martin. XMLtoDict Python. Github, 2014.
https://github.com/martinblech/xmltodict

[19] PLOCAN. R3M. Estramar, 2014.
http://r3m.estramar.eu/index.php/es/

[20] NOAA. OSMC. NOAA, 2014.
http://www.osmc.noaa.gov/

[21] Ifremer. SeaDataNet. Ifremer, 2013.
http://www.seadatanet.org/Data-Access

[22] NOAA. NDBC. NOAA, 2014.
http://www.ndbc.noaa.gov/

[23] NOAA. IOOS. NOAA, 2014.
http://www.ioos.noaa.gov/catalog/welcome.html

[24] OSM. OpenStreetMap. OSM, 2014.
http://www.openstreetmap.org/

[25] Open Source Geospatial Foundation. OpenLayers. OpenLayers, 2014.
http://openlayers.org/

[26] Pelican Mapping. ReadyMap. Pelican Mapping, 2011.
http://readymap.org/index_orig.html

[27] Esri. ArcGIS. Esri, 2014.
https://www.arcgis.com/features/

[28] Yahoo. Yahoo Maps. Yahoo, 2014.
https://espanol.maps.yahoo.com/

[29] Microsoft Corporation. Bing Maps. Microsoft Corporation, 2014.
http://www.bing.com/maps/

[30] Google. Google Maps. Google, 2014.
https://maps.google.es/

[31] Google. Google Earth API. Google, 2014.
https://developers.google.com/earth/?hl=es

[32] Here. Here. Here, 2014
http://here.com/

[33] Google. Google Charts API. Google, 2014.
https://developers.google.com/chart/?hl=es

[34] Highsoft AS. Highcharts. Highsoft AS, 2014.
http://www.highcharts.com/

[35] Downie, Nick. Chart.js. Nick Downie, 2014.
http://www.chartjs.org/

[36] Smith, Olly. Morris.js. Github, 2014.
https://github.com/morrisjs/morris.js/

[37] Leonello, Chris. JsPlot. Chris Leonello, 2014.
http://www.jqplot.com/index.php