Universidad de Valladolid

Escuela Técnica Superior

Ingenieros de Telecomunicación

Trabajo Fin de Máster

máster en ingeniería de telecomunicación

DISEÑO Y DESARROLLO DE EDUCAWOOD: UN SISTEMA WEB SOCIO-SEMÁNTICO PARA LA EDUCACIÓN MEDIOAMBIENTAL

Autor:

Dña. Jimena Andrade Hoz

Tutores:

Dr. D. Guillermo Vega Gorgojo
Dr. D. Juan Ignacio Asensio Pérez

Valladolid, 29 de Septiembre de 2021

Título:	**DISEÑO Y DESARROLLO DE EDUCAWOOD: UN SISTEMA WEB SOCIO-SEMÁNTICO PARA LA EDUCACIÓN MEDIOAMBIENTAL**
Autor:	**Dña. Jimena Andrade Hoz**
Tutores:	**Dr. D. Guillermo Vega Gorgojo** **Dr. D. Juan Ignacio Asensio Pérez**
Departamento:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

Tribunal

Presidente:	**Dr. D. Manuel Rodriguez Cayetano**
Vocal:	**Dr. Dña. Luisa María Regueras Santos**
Secretario:	**Dr. D. Juan Pablo de Castro Fernandez**

Fecha:	**29 de Septiembre de 2021**
Calificación:	

Resumen del TFG

En España se recogen y mantienen de forma oficial datos forestales que son publicados como datos abiertos. Estos datos abiertos pueden ser reutilizados en aplicaciones que permitan su visualización y utilización para fomentar actividades que involucren la educación medioambiental. Para favorecer la integración de estos datos se utilizan tecnologías semánticas, tarea realizada exitosamente en el proyecto europeo Cross-Forest, a partir del cual los datos del Inventario Forestal Nacional (IFN) y del Mapa Forestal Español (MFE) están disponibles como datos abiertos enlazados (LOD). Más allá del esfuerzo de las administraciones públicas, aproximaciones como la "ciencia ciudadana" permiten aumentar enormemente el número de personas involucradas en el enriquecimiento de las bases de datos forestales existentes. En el presente Trabajo Fin de Máster se presenta el diseño y desarrollo de EducaWood, un portal web socio-semántico que permite explorar la información forestal de una zona del territorio español proveniente de los datos del Cross-Forest y enriquecerla con anotaciones de árboles. EducaWood proporciona una interfaz de usuario de tipo formulario con la que pueden realizarse anotaciones semánticas, de manera que los usuarios no necesitan conocimiento de las tecnologías semánticas subyacentes (RDF y SPARQL). El principal objetivo de EducaWood es impulsar las actividades de aprendizaje medioambiental, uno de los aspectos principales de la “Educación para los Objetivos de Desarrollo Sostenible” de la UNESCO, que forma parte de la Agenda 2030 del Gobierno español. Con el uso de EducaWood, el profesorado puede proponer actividades de aprendizaje medioambiental que los estudiantes realizan de manera presencial u *online* (a través de visitas virtuales al campo). Los datos generados por los alumnos son enriquecidos con otras fuentes de datos abiertos como el MFE, el IFN o DBPedia. EducaWood ayuda al alumnado a conocer mejor su entorno, a la vez que se promociona la toma de conciencia ecológica.

Palabras clave

Educación medioambiental, conjuntos de datos forestales, datos abiertos enlazados, anotaciones semánticas, interfaz de usuario.

Abstract

Forestry data are collected and maintained officially in Spain and published as open data. This open data can be reused in applications that allow its visualisation and use to promote activities involving environmental education. Semantic technologies are used to facilitate the integration of these data, a task successfully carried out in the European Cross-Forest project, from which data from the National Forest Inventory (IFN) and the Spanish Forest Map (MFE) are available as linked open data (LOD). Beyond the efforts of public administrations, approaches such as "citizen science" allow for an enormous increase in the number of people involved in the enrichment of existing forest datasets. This Master Thesis presents the design and development of EducaWood, a socio-semantic web portal that allows to explore the forest information of an area of the Spanish territory from the Cross-Forest data and to enrich it with tree annotations. EducaWood provides a form-like user interface with which semantic annotations can be made, so that users do not need knowledge of the underlying semantic technologies (RDF and SPARQL). The main objective of EducaWood is to promote environmental learning activities, one of the main aspects of UNESCO's "Education for Sustainable Development Goals", which is part of the Spanish Government's Agenda 2030. Using EducaWood, teachers can propose environmental learning activities that students carry out in-person or online (through virtual field visits). The data generated by the students are enriched with other open data sources such as MFE, IFN or DBPedia. EducaWood helps students to learn more about their environment, while promoting ecological awareness.

Keywords

Environmental education, Forestry datasets, Linked Open Data, Semantic annotations, frontend.

Agradecimientos

A mis tutores, por apostar por mí una vez más.
A mi familia, por su apoyo incondicional.
A todo el equipo de EducaWood, por impulsar este trabajo y esta bonita experiencia.
A mis amigos, por animarme siempre en los momentos más difíciles de este camino.

Gracias.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

La educación medioambiental es uno de los pilares fundamentales de la "Educación para los Objetivos de Desarrollo Sostenible" de la Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura (UNESCO), formando parte de la "Agenda 2030[1]" del Gobierno Español. Numerosas investigaciones como [Der14] y [Che19] sugieren que la educación medioambiental mejora con aproximaciones de aprendizaje activo y construyendo el conocimiento a partir de experiencias en la vida real. Por ello, es importante disponer de herramientas que nos permitan "sacar el aula al bosque" para, por ejemplo, identificar diferentes especies arbóreas o realizar un análisis de la biodiversidad. Hoy en día contamos con el incentivo de los dispositivos móviles, a través de los cuales organizar actividades de aprendizaje ubicuo y contextualizado en el dominio medioambiental pueden resultar más atractivas y novedosas a los alumnos. El aprendizaje ubicuo [Hwa11] defiende que este no sólo ocurre en el aula, sino también en el hogar, en el lugar de juego, en la biblioteca, en el parque, en las interacciones cotidianas , etc. de manera que la vida cotidiana se convierte en espacio para nuevas pedagogías y nuevas prácticas de aprendizaje. Por ello, los profesores pueden proponer diferentes actividades fuera del aula, que serán realizadas por los alumnos. Por ejemplo, un profesor puede plantear una actividad de identificación de especies en una zona en la que existan dos muy similares. Los alumnos deberán ir al campo (aprendizaje ubicuo), localizar un árbol de cada especie de dicha zona y anotarlo. Más adelante, en el aula, se podrán utilizar esos datos para plantear nuevas actividades, como por ejemplo el cálculo del carbono fijado que generarían los árboles anotados.

Esta situación plantea dos problemas, el primero de ellos es que para permitir este tipo de actividades hace falta disponer de datos forestales de gran calidad; en el ejemplo anterior, el profesor necesitará saber en qué zona puede realizar la actividad deseada. Por otro lado, se necesita un sistema que soporte la generación de datos nuevos forestales, procedentes de las anotaciones de los estudiantes. En España, se dispone de dos repositorios abiertos de interés para este caso planteado, mantenidos por el Ministerio para la Transición Ecológica; el Inventario Forestal Nacional (IFN) y el Mapa Forestal Español (MFE). El IFN [IFN21] caracteriza los tipos de montes españoles, proporcionando información de más de un millón de árboles, mientras que el MFE [MFE21] recoge la cartografía de la situación de las masas forestales españolas. Ambos conjuntos de datos están disponibles para cualquiera que tenga interés en consultarlos, publicados como conjuntos de datos

[1] `https://www.mdsocialesa2030.gob.es/agenda2030/index.htm`

abiertos (*Open Data*, OD) [Hyl13]. Además, se generan otros repositorios de interés medioambiental en España, como son: el Inventario Nacional de Erosión de Suelos (INES), con información abierta de suelos, o los datos climatológicos de la Agencia Estatal de Meteorología (AEMET), aunque estos no son abiertos.

Publicar conjuntos de datos abiertos supone enfrentarse a barreras económicas y técnicas, debido al gasto que supone generarlos, los recursos necesarios para producirlos y el conocimiento técnico necesario para mantenerlos y reutilizarlos. El movimiento de datos abiertos [Kit14] tiene como objetivo solucionar estas barreras, abriendo los datos para una mayor reutilización y proporcionando herramientas técnicas que faciliten su acceso sin necesidad de poseer habilidades especiales para ello. A raíz de la expansión de los datos abiertos, surgieron los datos abiertos enlazados (*Linked Open Data*, LOD) [BL06], un conjunto de buenas prácticas para publicar datos abiertos interconectados entre sí. La evolución de los datos abiertos enlazados ha experimentado un crecimiento significativo en todos los dominios, hecho que podemos comprobar si observamos la evolución de la nube de LOD[2] en los últimos años, pasando de 12 conjuntos de datos en 2007 a más de 1300 en 2021. Por ello, existen cantidades masivas de datos abiertos enlazados a disposición de cualquiera que quiera utilizarlos. A pesar de los beneficios que ofrece LOD (integración, conexión y reutilización de los datos [All11]), el acceso a estos datos es complicado ya que a día de hoy las tecnologías de la Web Semántica, como SPARQL, RDF y OWL, siguen siendo desconocidas para la gran mayoría de desarrolladores web. Esto se debe en gran medida a que SPARQL tiene una curva de aprendizaje muy pronunciada y es muy propenso a errores, convirtiéndose en un trabajo tedioso incluso para expertos de la materia [Daq20]. Para evitar este problema, frecuentemente se recurre al desarrollo de APIs (*Application Programming Interfaces*) REST (*Representational State Transfer*) por encima de LOD, de manera que se abstraiga el conocimiento de las tecnologías semánticas. Esto es lo que se hizo en [MS20], a partir del cual desarrollé mi Trabajo Fin de Grado [AH20].

El problema de este trabajo previo es la falta de escalabilidad y flexibilidad que presenta, ya que la API REST fue diseñada específicamente para unos requisitos claramente definidos y concretos. Con un cambio de requisitos sería necesario redefinir toda la API, teniendo en cuenta además que dicha API estaba asociada a un único conjunto de datos. Ahora disponemos de múltiples repositorios de datos a los que acceder, además de nuevos tipos de datos forestales que soportar, por lo que se necesita una nueva estrategia que permita la integración de las fuentes, a ser posible de manera transparente. Este objetivo se pretende alcanzar con la propuesta de **CRAFTS** (*Configurable REST APIs For Triple Stores*) [VG21], una herramienta web que permite a los ingenieros de datos configurar APIs REST que utilicen datos de diferentes conjuntos de LOD de manera relativamente rápida, solucionando así ambos problemas. Los desarrolladores web pueden entonces utilizar una API CRAFTS para leer y escribir datos abiertos enlazados, sin necesidad de tener conocimiento de las tecnologías de la Web Semántica.

Teniendo en cuenta todo lo mencionado anteriormente, se plantea en el presente Trabajo Fin de Máster el diseño y desarrollo de una aplicación web de educación medioambiental de carácter semántico que utilice la herramienta de CRAFTS para solucionar las barreras de desconocimiento de tecnologías de la web semántica y permita el acceso y manipulación de varios conjuntos LOD. Los datos procederán de múltiples conjuntos de datos abiertos

[2] `https://lod-cloud.net/`

existentes y de nuevos tipos de datos que genere la propia aplicación. Por esto último, el carácter social seguirá teniendo un peso importante en la aplicación, puesto que no solo se reutilizarán los datos de las administraciones públicas, sino que será una aplicación que genere sus propios datos abiertos enlazados. La realización de este proyecto, denominado **EducaWood**, ha estado motivada por el concurso **Desafío Aporta 2020**, promovido por la iniciativa de datos abiertos del Gobierno de España (*datos.gob.es*[3]). La idea y prototipo de EducaWood fueron premiados con el tercer puesto[4].

1.2. Objetivos y alcance

El objetivo perseguido con la elaboración de este Trabajo Fin de Máster (TFM) es **el diseño e implementación de una aplicación web socio-semántica para la educación medioambiental que permita la visualización, interacción y generación de conjuntos de datos abiertos enlazados del ámbito forestal y favorezca un aprendizaje ubicuo y contextualizado para diferentes niveles educativos.** La aplicación fomentará el uso, reutilización y visualización de datos abiertos de las administraciones públicas, como son el Inventario Forestal Nacional y el Mapa Forestal Español, además de enriquecer estos conjuntos de datos con anotaciones de carácter social, en las que los usuarios de la aplicación puedan realizar anotaciones sobre árboles existentes, añadiendo, por ejemplo, fotos de los mismos, o incluso crear árboles nuevos y microhábitats. Con el sistema se intentarán conseguir tres metas fundamentales, que son: (i) hacer frente al problema existente de publicación de datos abiertos heterogéneos en la Web mediante el uso de la Web Semántica, (ii) realizar un prototipo que permita el acceso a LOD para lectura y escritura en múltiples conjuntos de datos abiertos y (iii) disponer de una aplicación que promueva la educación medioambiental de una manera novedosa e inexistente hasta el momento, para lo que será necesario desarrollar un *frontend* que favorezca tanto la visualización como la generación de LOD. Se pretende que sea una aplicación de carácter educativo, aunque el uso de la misma queda abierto a todo tipo de usuarios de la comunidad forestal y ciudadanos con intereses en este ámbito. Además, se espera que la aplicación sirva de herramienta de consulta, aprendizaje y colaboración tanto en las aulas como en el campo, de forma que pueda llegar a ser un nuevo y novedoso sistema con el que mejorar el conocimiento medioambiental y fomentar la concienciación del mantenimiento de los bosques y la biodiversidad de los mismos.

El desarrollo del proyecto se centrará, por un lado, en la visualización de los datos del IFN y del MFE en un mapa interactivo y, por otro, en la generación de nuevos datos abiertos enlazados mediante anotaciones sociales. En concreto, estas anotaciones permiten describir prácticamente al completo un árbol, de manera que se puede anotar del mismo la especie, la ubicación, medidas como diámetro y altura, el estado en el que se encuentra, imágenes de sus diferentes partes e, incluso, un análisis detallado de su biodiversidad. A partir de estas dos funcionalidades, el docente ya puede proponer numerosas actividades a sus alumnos. Por ejemplo, una de ellas podría ser buscar previamente en el mapa una zona donde abunde una especie determinada, y proponerles que identifiquen un árbol de dicha especie, o un árbol de una especie diferente a la que abunde. Esos mismos árboles pueden medirlos, y con los datos de altura y diámetro se pueden realizar otras actividades más complejas en el aula, como calcular el carbono que fijan, utilizando las ecuaciones de

[3] `https://datos.gob.es/es`

[4] `https://datos.gob.es/es/desafios-aporta/desafio-aporta-2020`

biomasa. Con EducaWood se podrán desarrollar múltiples actividades y a distintos niveles educativos, desde niveles de primaria o secundaria, hasta universitarios como el grado de ingeniería forestal u otros grados relacionados con la educación.

1.3. Metodología

La metodología a seguir para la consecución del Trabajo Fin de Máster ha sido la iterativa e incremental, propia del Proceso Unificado de Desarrollo de Software [Jac00]. En este marco de trabajo, el desarrollo del sistema se ha ido dividiendo en múltiples iteraciones, cada una comprendiendo las fases de Inicio, Elaboración, Construcción y Transición. Fundamentalmente ha sido un proceso iterativo, en el que cada iteración finalizaba al tener una nueva reunión con los tutores del TFM y otros miembros del equipo de EducaWood, expertos en el dominio forestal, en tecnología educativa y en Web Semántica. Cada una de las iteraciones da como resultado un incremento de la funcionalidad de la aplicación desarrollada, mejorando el sistema en desarrollo. La selección de qué funcionalidad se ha de desarrollar en cada iteración ha sido impulsada por la mitigación de los riesgos del sistema y el establecimiento de prioridades en función del horizonte temporal.

Las diferentes iteraciones realizadas han estado enfocadas en diversos aspectos de la aplicación, en función del grado de desarrollo de la misma. En las primeras iteraciones el foco principal estuvo en la planificación y análisis, mientras que, a medida que fue madurando la idea con la implementación, este análisis fue evolucionando hasta el resultado final. Para ello, se ha seguido la visión en cinco fases del Ciclo de Vida del Desarrollo de Sistemas, SDLC por sus siglas en inglés (Systems Development Life Cycle) [Val17b], que son:

- Planificación. Esta fase estuvo motivada por la primera fase del Desafío Aporta, que consistió en la descripción detalla de una idea que identificase nuevas oportunidades de captar, analizar y utilizar los datos abiertos en el desarrollo de soluciones en el ámbito educativo. En esta etapa fue fundamental la ayuda de investigadores forestales del grupo iuFor[5], en concreto de Irene Ruano, Cristóbal Ordóñez y Felipe Bravo.

- Análisis. Una vez desarrollada la idea, las siguientes iteraciones se centraron en analizar y discutir los requisitos y las funcionalidades que la aplicación debería cumplir. Esta fase fue recurrente a lo largo de todo el desarrollo del TFM.

- Diseño. Comprendió las iteraciones destinadas al diseño de la arquitectura del sistema completo, al diseño del modelo de datos para la API CRAFTS y el diseño de la interfaz de usuario.

- Implementación. Consistió en el desarrollo del prototipo de la aplicación. En las diferentes iteraciones se fueron estableciendo prioridades sobre qué funcionalidades debían implementarse primero, puesto que el horizonte temporal era limitado y la idea propuesta muy ambiciosa. Este proceso estuvo motivado por la segunda fase del Desafío Aporta, puesto que la idea presentada en la fase anterior consiguió la máxima puntuación de todos los participantes.

[5] `http://sostenible.palencia.uva.es/`

- Validación y lanzamiento. Se corresponde con las iteraciones finales del proyecto. Consistió en el despliegue en producción de la aplicación y la posterior comprobación de que los casos de uso especificados se lograban con éxito.

1.4. Estructura del documento

El documento se encuentra dividido en seis capítulos, entre los que se incluye el presente. En el capítulo 2 se realiza un estudio del estado del arte actual relacionado con los datos abiertos enlazados en el ámbito educativo, así como las diferentes propuestas sobre cómo abordar el problema de acceso a los mismos por falta de conocimiento relacionado con la Web Semántica. En el capítulo 3 se describe el proceso de planificación y análisis de la aplicación, en el que se incluyen los requisitos y los casos de uso de la misma, además de una primera aproximación de la vista de la interfaz de usuario y el modelo de datos a utilizar. El capítulo 4 se centra en el diseño de la aplicación completa, así como en el diseño de cada una de las partes individualmente. En el capítulo 5 se presenta el prototipo final y se desarrolla un escenario de uso completo de la aplicación. Para terminar, el capítulo 6 recoge las principales conclusiones extraídas con la elaboración del TFM, se proponen posibles líneas de trabajo futuro y se exponen las principales limitaciones actuales que presenta.

Capítulo 2

Estado del arte

2.1. Introducción

La Web es un medio de publicación e intercambio de información en continuo desarrollo y expansión. Hoy en día es el medio más utilizado para la publicación de datos abiertos, crecimiento que se debe en gran medida al impulso de los gobiernos de todo el mundo (como la Iniciativa Aporta[1] del Gobierno de España), la creciente publicación de datos de investigación y la recopilación, el análisis y la publicación en línea de datos de redes sociales, entre otros muchos ejemplos [Ló17]. Este hecho se ve reflejado en el ámbito forestal con la publicación de datos abiertos como el IFN o el MFE. Los datos allí recogidos pueden utilizarse para diferentes fines, siendo uno de ellos la educación medioambiental. Sin embargo, su reutilización no resulta trivial, debido a que los formatos en los que están publicados no siguen unos estándares bien definidos. Por ello, se recurre a las tecnologías de la Web Semántica para impulsar la generación de datos abiertos enlazados, de manera que sea posible la reutilización de los mismos. En el presente capítulo se presentan las ventajas que trae consigo la utilización de las tecnologías semánticas en aplicaciones que usen datos medioambientales, a la vez que los problemas que surgen debido a su utilización.

Este capítulo se estructura como sigue: la sección 2.2 presenta el concepto de datos abiertos, centrándose en los datos abiertos en el dominio medioambiental y su utilización en aplicaciones de carácter educativo. En la sección 2.3 se presenta la Web Semántica, sus fundamentos, los datos abiertos enlazados y el problema de accesibilidad de los mismos. Finalmente, se realizan unas conclusiones relacionadas con los diferentes temas tratados en el capítulo en la sección 2.4.

2.2. Datos abiertos en el dominio medioambiental

Esta sección presenta una visión global de lo que son los datos abiertos (OD, del inglés Open Data), centrando el foco en el dominio medioambiental. Desde sus inicios allá por 2010, se ha promovido su difusión y crecimiento, debido a que la apertura de datos ha resultado ser beneficiosa en múltiples sectores, como en el de la educación [RR19]. Los datos abiertos pueden contribuir a mejorar la educación digital, ya que se permiten identificar

[1] La Iniciativa Aporta surge en 2009 con el fin de promocionar la apertura de la información pública y desarrollo de servicios avanzados basados en datos. Es el elemento clave de la política de datos del gobierno de España, teniendo como objetivo principal la armonización y el aprovechamiento eficiente de las sinergias entre los proyectos de datos ya en marcha. `https://datos.gob.es/es/acerca-de-la-iniciativa-aporta`

nuevas oportunidades para captar, analizar y utilizarlos para desarrollar soluciones en el ámbito educativo [Apo20]. Por ello, en esta sección se plantean varias ideas de reutilización de los datos abiertos medioambientales en la educación.

2.2.1. Datos abiertos

El movimiento de datos abiertos [Kit14] fomenta la publicación de los datos de manera abierta en la Web para que se puedan reutilizar en otras aplicaciones y servicios. Además, facilita herramientas de investigación que eliminan la necesidad de disponer de habilidades especiales para poder reutilizarlos. Se basa en tres principios fundamentales que son: apertura, participación pública y colaboración ciudadana. Los datos abiertos persiguen el objetivo de que estos estén disponibles de forma generalizada, sin restricciones, para cualquier organización o individuo que desee consultarlos. El movimiento cobró importancia desde finales de la década de los 2000, puesto que desde entonces se han unido una gran cantidad de gobiernos e instituciones de todo el mundo que apuestan por las sociedades abiertas, la transparencia y la publicación de los datos en formatos que permitan su reutilización [Hyl14].

En España concretamente, uno de los órganos precursores de los datos abiertos es Red.es[2], una entidad pública adscrita al Ministerio de Asuntos Económicos y Transformación Digital, que trabaja a través de la Iniciativa Aporta para impulsar la cultura de la apertura de la información pública y desarrollar el mercado de la reutilización en España. El objetivo final de la iniciativa es potenciar la oferta de nuevos productos y servicios digitales, dinamizar la actividad económica y empresarial y, en última instancia, generar valor para el conjunto de la sociedad. Otro ejemplo son las Comunidades Autónomas, las cuales tienen a disposición de la ciudadanía portales web donde acceder a los datos abiertos, como por ejemplo el portal de Castilla y León[3]. Existen otros muchos organismos públicos que generan y exponen datos abiertos, como por ejemplo el Ministerio para la Transición Ecológica y el Reto Demográfico, que publica datos abiertos forestales de gran interés para el presente TFM.

2.2.2. Datos abiertos medioambientales

Se necesitan grandes cantidades de datos medioambientales para hacer gestión y planificación forestal. Por ello, no es de extrañar que cada vez sean más frecuentes iniciativas de publicación de datos abiertos relacionados con este sector. En la mayoría de países europeos, este tipo de datos abiertos se almacena en inventarios y mapas forestales. Un **inventario forestal** es la suma de la información relativa a la ubicación, extensión, naturaleza, estado y capacidad productora de las áreas forestales en la actualidad y su probable evolución en un futuro. A su vez, un **mapa forestal** representa la cartografía de la situación de las masas forestales de un territorio. Los mapas forestales se representan mediante **teselas**, del ingles *patch*, que constituyen unidades de vegetación con una estructura homogénea. Es importante resaltar que las metodologías seguidas para conseguir los resultados son muy costosas y duraderas, sin descontar que es necesario dedicar multitud de recursos económicos, personales y temporales para ello.

[2] `https://www.red.es/redes/es`
[3] https://datosabiertos.jcyl.es/web/es/datos-abiertos-castilla-leon.html

En concreto, en España, existe el Inventario Forestal Nacional (IFN), accesible a través de [IFN21] y el Mapa Forestal de España (MFE), que se puede consultar en [MFE21]. Estos datos se utilizan para la planificación y gestión forestal del país, pero también pueden ser utilizados para otros fines como, por ejemplo, la educación medioambiental. Con estos datos disponemos de información referida a más de un millón de árboles, además de información de teselas, zonas boscosas, especies, tipo de vegetación, etc. Por ello, ¿por qué no utilizar esta gran cantidad de información en beneficio de la educación medioambiental? Las actividades que se podrían realizar utilizando estos datos son numerosas, dependiendo de la imaginación del docente y en función del nivel educativo. En la Tabla 2.1 se muestran algunos de los ejemplos de actividades que se podrían realizar gracias a la reutilización de los datos. Estas ideas fueron impulsadas por el grupo de docentes en ingeniería forestal que forma parte del equipo de EducaWood. Los intereses de las actividades educativas propuestas son muy variados: (i) basándonos en características básicas como la forma de la hoja o el color de la corteza los alumnos pueden aprender a diferenciar grandes grupos de especies forestales, (ii) con la salida al campo podrán descubrir la variabilidad de plantas y árboles que viven en un bosque y aumentará su interés por la naturaleza, además, (iii) trabajarán otras materias como las matemáticas para calcular los índices de diversidad estructural o el carbono fijado, y por último, (iv) trabajarán en grupo. Además, los datos del IFN y del MFE podrían ser enriquecidos con otros conjuntos de datos abiertos como los de la Wikipedia[4].

2.2.3. Aplicaciones que usan datos abiertos medioambientales

En esta última subsección se analizan dos casos particulares de aplicaciones del ámbito medioambiental que podrían utilizarse como herramientas educativas, estas son ***Observation.org*** y ***Integrate Tree Microhabitat App***. Ambas resultan de interés debido a que exponen datos abiertos forestales. Además, la primera de ellas se apoya en un modelo colaborativo que difunde el conocimiento a través de una herramienta web al alcance de toda la ciudadanía, factor positivo que ayuda a generar una mayor cantidad de datos que los generados por las instituciones públicas.

Observation.org

Observation.org [Obs21] es una aplicación web que forma parte de la *Observation International Foundation*, siendo la plataforma de observación de la naturaleza más importante de Europa. Consiste en una aplicación en la que miles de usuarios recogen información de biodiversidad alrededor de todo el mundo. Esta información se incorpora a la base de datos del proyecto, que es compartida y sus datos expuestos para todo el público. Por ello, Observation.org es una poderosa herramienta para la conservación, investigación, gestión, educación y experiencia del medio ambiente. La filosofía de datos abiertos es uno de sus pilares fundamentales. Su objetivo es compartir datos de biodiversidad general, tanto los registrados en el pasado como en el presente, para que sean fuente de conocimiento para el futuro. Uno de los proyectos más relevantes en los que participa esta aplicación es el proyecto Cambio Climático en CyL [CyL21], que consiste en la evaluación de la vulnerabilidad de los ecosistemas de Castilla y León frente al cambio climático. Con la información almacenada en esta aplicación, los expertos en este ámbito hará un análisis integrado para los ecosistemas naturales y ayudarán a definir una serie de medidas de aplicación y adaptación.

[4]https://es.wikipedia.org/wiki/Wikipedia:Portada

Tabla 2.1: Ejemplos de actividades con LOD forestales

Nombre	Datos	Actividad	Nivel Educativo
Identificación de especies	MFE, IFN	Encontrar un árbol de la especie dominante de la zona	Primaria, secundaria y bachillerato
	MFE, IFN	Encontrar un árbol de una especie diferente a la especie dominante de la zona	Primaria, secundaria y bachillerato
	MFE, IFN	Realización de un herbario forestal identificando la especie y fotografiando sus hojas y sus frutos	Grado: asignatura de botánica
Diversidad estructural	MFE	Estimación de la diversidad estructural en grupo midiendo todos los árboles y la distancia a sus tres vecinos, cubriendo entre todos una zona delimitada por el profesor	Bachillerato y grado
Estimación de carbono fijado	MFE	Identificar un árbol. Medir su diámetro y su altura e identificar su especie. Utilizar las ecuaciones de biomasa para calcular cuánto carbono fija el árbol	Secundaria y bachillerato
Análisis de biodiversidad	MFE	Identificación de microhábitats en una zona o en un número de árboles	Secundaria y bachillerato
	MFE	Identificación de madera muerta en una zona	Secundaria y bachillerato
Análisis de enfermedades	MFE	Localización de árboles con muestras de enfermedad o señales de plagas. Identificación de la plaga o enfermedad	Grado: asignatura de plagas y enfermedades forestales

Integrate Tree Microhabitat App

La aplicación *Integrate Tree Microhabitat* [ITM21] se desarrolló como herramienta de apoyo de campo para identificar estructuras de microhábitats arbóreos relevantes para la biodiversidad. Para ello, utiliza el catálogo de datos *Tree Microhabitats* elaborado por un grupo de expertos de alto nivel del grupo *Integrate Network*[5], una alianza de representantes de diferentes países europeos que promueve la integración de la conservación de la naturaleza en la gestión forestal sostenible a nivel político, práctico y de investigación.

2.3. Acceso a datos abiertos enlazados

La Web es el principal medio de publicación de datos abiertos y el más extendido, ya que permite gran variedad de formas de representar los datos y acceder a ellos. Sin embargo, este hecho trae consigo varios problemas, como son la búsqueda ineficiente de contenidos o la complicada reutilización e integración de los datos publicados. Como solución para

[5]https://integratenetwork.org/about-us/

mitigar estos problemas, se propone la Web Semántica [BL01], una evolución de la Web clásica en la que los datos presentados poseen un formato único y procesable por máquinas. La principal motivación del uso de tecnologías semánticas es la integración de datos, lo que favorece posteriormente su acceso. En este capítulo se repasan los conceptos básicos de la Web Semántica y uno de sus resultados más relevantes, el movimiento de datos abiertos enlazados (LOD), que persigue la generación de datos conectados entre sí, disponibles para todos los ciudadanos y reutilizables para diferentes aplicaciones y fines [Biz11]. Además, se describe un problema importante presente a la hora de utilizar LOD en aplicaciones web: el acceso a repositorios de triplas no es sencillo en absoluto, un factor que imposibilita en muchas ocasiones la integración de LOD por parte de los desarrolladores web. El fin último del capítulo es identificar las diferentes posibilidades existentes para generar, visualizar y reutilizar LOD, de manera que se llegue a la solución más adecuada para utilizar en la implementación de EducaWood.

2.3.1. Web Semántica

La Web Semántica [BL01] fue concebida para ayudar a la evolución del conocimiento humano como un todo, permitiendo a las máquinas comprender los documentos publicados en la Web. Persigue la transformación de Internet de un contenido no estructurado a uno estructurado y semántico, con la posibilidad de conectar datos de diferentes fuentes y de tal manera que los ordenadores puedan interpretar y manipular los diferentes conjuntos de datos en un formato estándar, ayudando a los humanos a dar sentido a la información. La Web Semántica no es una web aparte, sino una ampliación de la actual, en la que la información adquiere un significado bien definido, lo que permite a los ordenadores y a las personas trabajar mejor en cooperación. El principal objetivo es que la Web deje de ser una red de documentos para convertirse en una red de datos enlazados. Se trata del desarrollo de formalismos y tecnologías que faciliten la creación, el intercambio y la consulta de datos enlazados utilizando ontologías compartidas para establecer interpretaciones comunes. Por esta razón, un nombre alternativo para la Web Semántica es la Web de datos [O'H12].

Para que la Web Semántica funcione, los ordenadores deben tener acceso a colecciones estructuradas de información y conjuntos de reglas de inferencia que puedan utilizar para realizar razonamientos automatizados [BL01]. Las tecnologías utilizadas para ello son RDF y SPARQL. Los datos siguen un modelo de datos llamado **RDF** (*Resource Description Framework*) [Cyg14] el cual constituye la base para publicar y enlazar los datos. RDF codifica los datos en conjuntos de triplas, cada una de las cuales se asemeja al sujeto, verbo y objeto de una frase elemental. La **tripla** es el elemento de construcción básico de RDF, la cual consta de dos entidades (sujeto y objeto) y una relación binaria entre ellas (predicado). Se puede ver un ejemplo en la Tabla 2.2. El sujeto y el objeto se identifican mediante un identificador universal de recursos (URI), tal y como se utiliza en un enlace de una página web. Los verbos también se identifican mediante URIs, lo que permite a cualquiera definir un nuevo concepto o un nuevo verbo definiendo un URI para él. Por otro lado, **SPARQL** [Har13] constituye el lenguaje de consultas estándar de la Web Semántica [All11].

Por otro lado, es importante tener en cuenta que dos bases de datos pueden utilizar diferentes identificadores para referirse al mismo concepto. Un programa que quiera comparar o combinar información en las dos bases de datos tiene que saber que estos dos términos se están utilizando para referirse lo mismo. Para evitar este tipo de ambigüeda-

Tabla 2.2: Ejemplos de triplas RDF

Sujeto	**Predicado**	**Objeto**
`Tree`	`hasAnnotation`	`Annotation`
`Tree`	`createdBy`	`IFN`
`Annotation`	`latitude`	`3.4`
`Annotation`	`status`	`alive`

des existen las ontologías. Una **ontología** [BL01] es un documento o archivo que define formalmente las relaciones entre diferentes términos. El tipo más típico de ontología para la Web tiene una taxonomía y un conjunto de reglas de inferencia. La **taxonomía** define clases de objetos y relaciones entre ellos. Por ejemplo, un `árbol` podría definirse como un tipo de `planta`. Las relaciones entre las clases se establecen mediante la asignación de propiedades y permitiendo que las subclases hereden dichas propiedades. Siguiendo el ejemplo anterior, se podrían relacionar las clases `árbol` y `tronco` mediante la propiedad tiene (`Un árbol tiene un tronco`).

2.3.2. Datos abiertos enlazados

Un concepto muy ligado a la Web Semántica es el de datos enlazados (LD, del inglés *Linked Data*), que se definen formalmente según la W3C [Hyl13] como un patrón para hiperenlazar conjuntos de datos que puedan ser leídos por máquinas, utilizando técnicas de la Web Semántica, especialmente RDF y URI. Además, LD permite realizar consultas mediante SPARQL. Cuando estos datos se publican en la Web pública, se denominan generalmente datos abiertos enlazados (LOD, del inglés *Linked Open Data*). Tim Berners-Lee, inventor de la Web y propulsor del proyecto de LD, sugirió un esquema de 5 estrellas acumulativo para LOD, en el que cada estrella supone que los datos cumplen un paso adicional, además de todos los anteriores. Estos son:

1. Publicar datos en la web en cualquier formato (por ejemplo, PDF, JPEG) acompañados de una licencia abierta explícita

2. Publicar datos estructurados en la web en un formato legible por máquina (por ejemplo, XML)

3. Publicar datos estructurados en la web en un formato de datos documentado y no propietario (por ejemplo, CSV, KML).

4. Publicar datos estructurados en la web como RDF (por ejemplo, Turtle, RDFa, JSON-LD, SPARQL)

5. En el RDF, hacer que los identificadores sean enlaces (URL) a fuentes de datos útiles

El resultado más satisfactorio de LOD es la nube de datos abiertos enlazados (*The Linked Open Data Cloud*[6]), la cual recoge todas las fuentes de datos publicados siguiendo los principios de LOD. En la Figura 2.1 se muestra el conjunto de datos actualizado a mayo de 2021, con un total de 1301 conjuntos de LOD.

[6] `https://lod-cloud.net/`

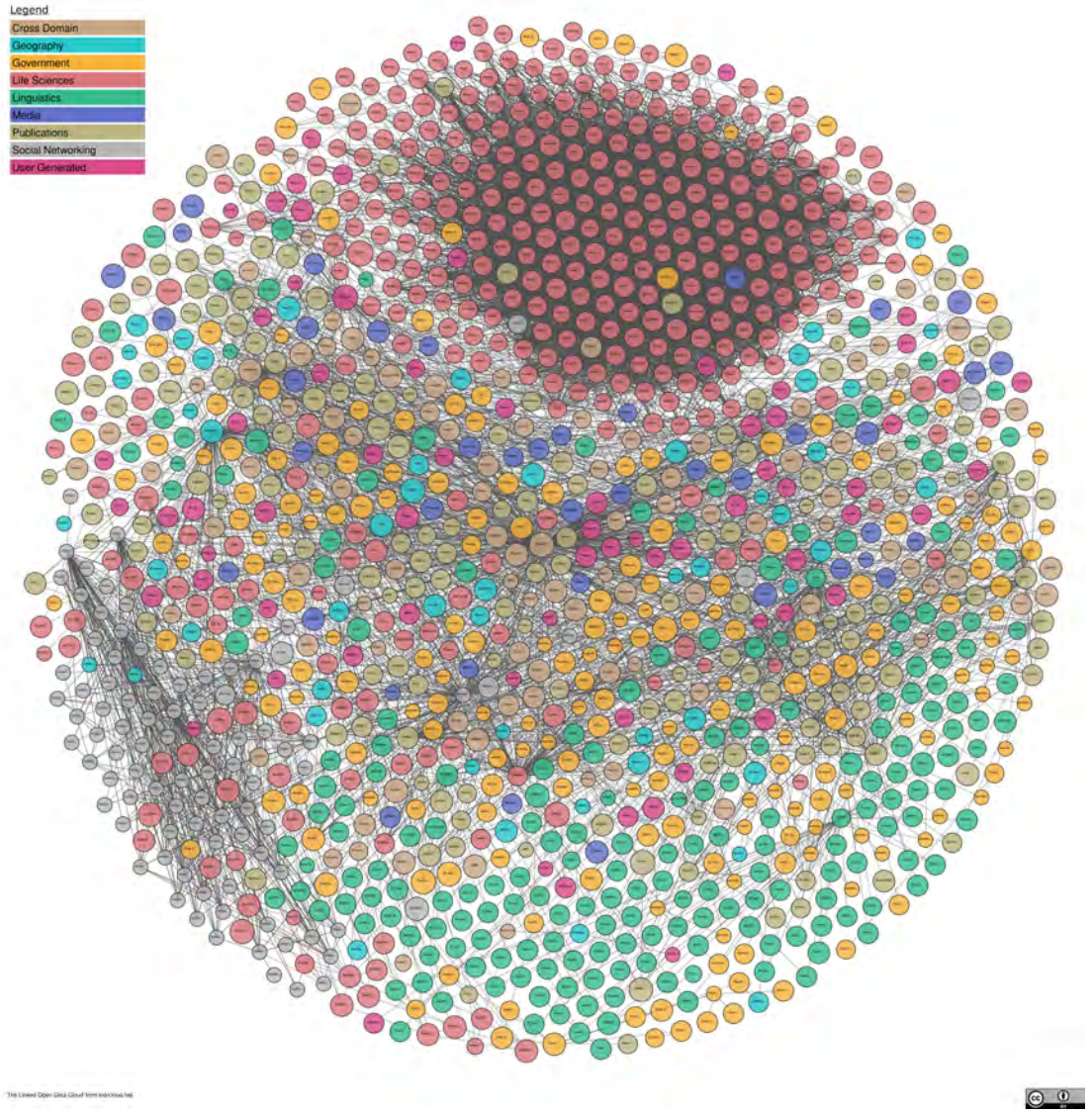


Figura 2.1: Diagrama de la nube de *Linked Open Data*. `https://lod-cloud.net/clouds/lod-cloud.svg`

Para el caso que nos ocupa, cabe resaltar el proyecto **Cross-Forest** [CF20], el cual nace con el objetivo de desarrollar servicios de Infraestructuras de Servicios Digitales orientados al control y predicción de incendios forestales utilizando datos abiertos enlazados. Para ello, se crearon un total de cinco ontologías que han permitido modelar inventarios forestales y mapas de España y Portugal, así como posiciones y medidas. El objetivo de estas ontologías es que puedan ser utilizadas a nivel internacional para representar y publicar los datos de inventarios y mapas forestales de cualquier país en un formato abierto y estándar. En el caso concreto de España, se han transformado y publicado como LOD los datos del IFN y del MFE. Posteriormente ha sido posible enlazarlos a otras fuentes como DBpedia[7] y Wikidata[8].

[7] `https://www.dbpedia.org/`
[8] `https://www.wikidata.org/wiki/Wikidata:Main_Page`

2.3.3. Acceso a datos abiertos enlazados

Antes de la iniciativa LOD, la Web Semántica sufría la "situación del huevo o la gallina", ya que no había aplicaciones porque no existían datos y, a su vez, no había datos porque ninguna aplicación los utilizaba [Ver16]. Gracias a su desarrollo y extensión, hoy en día existen infinidad de triplas de datos abiertos enlazados en miles de grafos de conocimiento RDF en la Web, como hemos podido comprobar en la nube de datos abiertos enlazados. Sin embargo, muy pocos de estos grafos pueden realmente consultarse en tiempo real desde las aplicaciones web. Sólo un número limitado de grafos está disponible en una interfaz de consulta, y las interfaces existentes pueden ser costosas de alojar y difíciles de manejar. Dada esta enorme cantidad de datos abiertos enlazados, deberíamos ser capaces de construir aplicaciones que aprovechen la inteligencia de estos datos, las llamadas aplicaciones semánticas. Sin embargo, contamos con ciertas limitaciones que imposibilitan su desarrollo.

La primera de ellas se debe a la baja disponibilidad de repositorios de datos abiertos enlazados, causado por un problema de dos caras al que se enfrenta actualmente la Web Semántica: (i) la mayoría de grafos de conocimiento no se publican de forma que puedan ser consultados y (ii) los grafos que se publican en un *endpoint* de SPARQL sufren frecuentes caídas [BA13]. Esta falta de disponibilidad se vuelve aún más problemática si se consideran las consultas sobre múltiples conjuntos de datos distribuidos. Por ello, es de esperar que muchos publicadores de datos eviten la responsabilidad de alojar un *endpoint* SPARQL, ofreciendo en su lugar archivos de datos (*data dumps*). Esto no nos acerca a la Web Semántica, ya que estos archivos de datos deben descargarse y almacenarse localmente para que la consulta real pueda realizarse sin conexión. Además, su utilización sólo es posible en máquinas suficientemente potentes, no en dispositivos móviles, cuya popularidad no deja de aumentar, y su puesta en marcha requiere conocimientos técnicos. La otra gran limitación se debe a que tan solo un número muy reducido de desarrolladores de aplicaciones web son expertos en Web Semántica y sus tecnologías, lo que supone un gran problema a la hora de implementar software de servidor (*bakend*) que realice consultas SPARQL a los diferentes repositorios de triplas. Por todo ello, una cantidad importante de grafos de conocimiento LOD no se pueden consultar de forma fiable ni son fácilmente accesibles en la Web [Ver16].

Para mitigar este problema de acceso a los datos abiertos enlazados, la pelota se encuentra ahora en el tejado de los expertos en Web Semántica. Si se quiere conseguir que las aplicaciones semánticas se conviertan en una realidad, es necesario considerar las opciones existentes en cuanto a la publicación y consumo de LOD en la Web. Entre los dos extremos de *data dumps* y *endpoints* SPARQL, se encuentra todo un espectro de posibles interfaces web, que han permanecido durante mucho tiempo inexploradas [Ver16]. Desde sus inicios, la comunidad de expertos en Web Semántica ha propuesto varios enfoques para facilitar el acceso a los almacenes de triplas, como son la interfaz *Linked Data Platform* [Spe15] y *Triple Pattern Fragments* [Ver16], aunque es cierto que estas soluciones quedan muy limitadas a la comunidad de expertos de la Web Semántica. Como solución, existen otros enfoques que proponen nuevas serializaciones de los resultados de LOD y SPARQL al formato JSON, como son JSON-LD [Spo20] y SPARQL Transformer [Lis19]. Además, existen otras opciones cuyas funcionalidades se encuentran más próximas a lo requerido para el desarrollo del presente TFM, puesto que buscan la creación de APIs sobre puntos SPARQL, como son RAMOSE [Daq20], grlc [MP16], OBA [Gar20] y CRAFTS [VG21].

RAMOSE

RAMOSE [Daq20] es una herramienta genérica desarrollada en Python para crear APIs REST sobre *endpoints* SPARQL. Para ello, es necesaria la creación de archivos de configuración textual que permiten la consulta a *endpoints* SPARQL mediante simples llamadas a APIs web REST. La respuesta incluye los datos consultados en formato JSON o CSV, ocultando así todas las complejidades intrínsecas de SPARQL a los usuarios desconocedores de tecnologías semánticas. Dado que RAMOSE es genérico y puede utilizarse con cualquier *endpoint* SPARQL, representa una solución técnica sencilla para los desarrolladores de aplicaciones web que deseen crear un servicio API para acceder a LOD almacenado como RDF en repositorios de triplas convencionales.

grlc

grlc[9] [MP16] consiste en un servidor ligero que toma consultas SPARQL curadas en repositorios de GitHub[10] y las traduce a APIs RESTful de LOD automáticamente. Para ello, es necesario que el desarrollador disponga de un repositorio de GitHub en el que alojar las consultas SPARQL. Esto requiere de conocimientos del lenguaje SPARQL por parte del desarrollador, lo que es una clara desventaja.

OBA

OBA [Gar20] es una propuesta reciente para crear APIs REST sobre repositorios de triplas. Partiendo de una ontología, OBA permite generar automáticamente una API siguiendo las mejores prácticas de las APIs REST. Para ello, utiliza tecnologías estándar conocidas por los desarrolladores web (como *OpenAPI Specification* [Mil20] o JSON) y las combina con los estándares de la W3C (OWL, marcos JSON-LD y SPARQL) para crear las APIs. Estas APIs presentan una documentación para su mantenimiento, pruebas de unidades y validación automatizada de recursos y clientes (en Python, Javascript, etc.). De esta manera, los no expertos en la Web Semántica pueden acceder al contenido de un grafo de conocimiento de manera sencilla. OBA emplea una serie de plantillas de consulta predefinidas para convertir las llamadas a la API en consultas SPARQL. La limitación más importante que presenta esta herramienta es que genera las APIs basándose en una ontología, lo que es atractivo si pensamos en automatizar la creación de una API, pero poco práctico si queremos cierta personalización de los datos expuestos. Por ejemplo, se puede querer una API que nos devuelva solo cierta información de una ontología y no toda, resultando ineficiente que la API cargue toda la información innecesariamente.

CRAFTS

Configurable REST APIs For Triple Stores (CRAFTS) [VG21] es una herramienta que permite configurar APIs REST sobre múltiples almacenes de triplas, de manera que los desarrolladores web puedan utilizarla para leer y escribir LOD. CRAFTS maneja automáticamente la traducción de las llamadas a la API en consultas SPARQL, entregando los resultados en formato JSON. La API de CRAFTS es uniforme, independiente del dominio y se describe con la especificación OpenAPI. Todas las herramientas anteriormente mencionadas proporcionan una funcionalidad similar a las consultas SPARQL de CRAFTS: una vez definidas las consultas que se van a admitir, los usuarios de LOD harán llamadas a

[9] http://grlc.io/
[10] https://github.com/

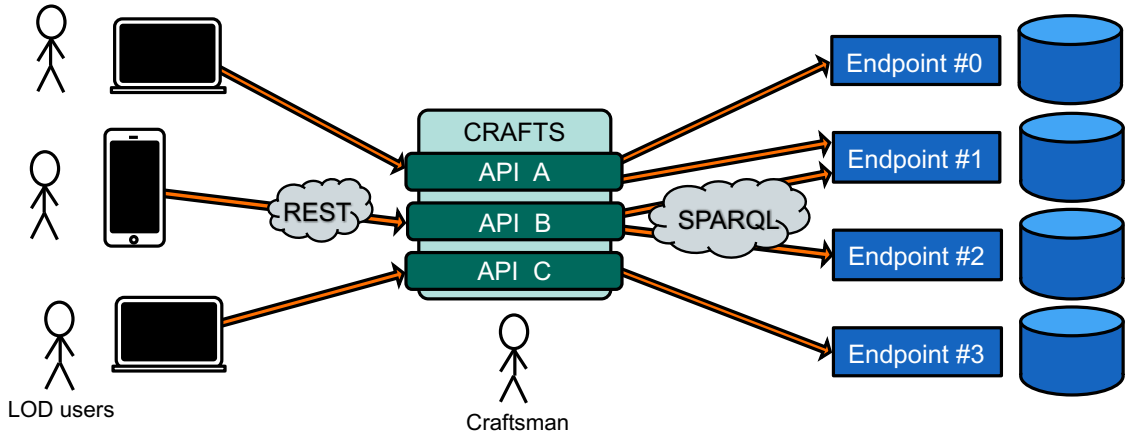


Figura 2.2: Visión general de CRAFTS (Fuente: [VG21])

la API en lugar de crear consultas SPARQL. Una diferencia en CRAFTS es el uso de Mustache[11] como mecanismo conocido de plantillas para consultas SPARQL parametrizadas, mientras que las herramientas mencionadas emplean sus propias convenciones personalizadas. Además, ninguna de ellas (a excepción de OBA) soporta operaciones sobre recursos LOD, una de las capacidades más distintivas de CRAFTS. Siguiendo con otras ventajas destacables, permite la integración de varios *endpoints* SPARQL y la configuración del modelo de una API no es tan exigente comparado con la creación de consultas SPARQL. CRAFTS se explica en profundidad en la siguiente sección (2.3.4).

2.3.4. *Configurable REST APIs For Triple Stores*

CRAFTS[12] es una herramienta para acceder a LOD a través de APIs REST. En la Figura 2.2 se muestra una visión general del sistema, en la que se pueden distinguir dos tipos de usuarios: los *craftsmen* y los usuarios de LOD (*LOD users*). Los *craftsmen* se corresponden con los ingenieros expertos en Web Semántica, encargados de configurar las APIs. Estas APIs proporcionan a los usuarios de LOD acceso trasparente a uno o varios *endpoints* SPARQL. CRAFTS maneja automáticamente la traducción de las llamadas a la API en consultas SPARQL, entregando el resultado a los usuarios de LOD en formato JSON. Así, estos usuarios pueden hacer uso de una API CRAFTS a través de llamadas REST sin requerir conocimientos de RDF, OWL, o SPARQL.

En las siguientes secciones se profundiza en los conocimientos de la API de CRAFTS y se presenta su arquitectura, puesto que será la herramienta a utilizar para la implementación del TFM. Los ejemplos de uso concretos se mencionarán en capítulos posteriores.

La API de CRAFTS

CRAFTS proporciona una API uniforme (resumida en la Tabla 2.3) que está documentada con la especificación OpenAPI [Mil20]. Se puede acceder a la documentación a través de la URL siguiente: `https://crafts.gsic.uva.es/docs/`. Una API REST soporta diferentes llamadas, en las que cada una de ellas se incluye la ruta (*path*), que identifica

[11] https://mustache.github.io/
[12] https://crafts.gsic.uva.es/

a un recurso expuesto por la API y la operación a realizar, que se corresponde con un método propio del protocolo HTTP (GET, PUT, DELETE, etc.) [Fie99]. Adicionalmente, se pueden añadir parámetros de consulta al final del *path*, especificados con un símbolo de interrogación (`?`) y diferentes pares de `nombre=valor` separados con el símbolo ampersand (`&`). Las operaciones de escritura (PUT, PATCH y POST) suelen incluir un cuerpo de petición que contiene la representación del recurso web que se va a crear o actualizar.

Tabla 2.3: Resumen de especificaciones de la API expuesta por CRAFTS

ID	**Op**	**Ruta**	**Query param.**	**Descripción**
C0	`GET`	`/apis`	-	Lista las APIs disponibles
C1	`GET`	`/apis/{apiId}`	-	Obtiene la API `apiId`
C2	`PUT`	`/apis/{apiId}`	-	Crea o reemplaza la API `apiId`
C3	`DELETE`	`/apis/{apiId}`	-	Elimina la API `apiId`
C4	`GET`	`/apis/{apiId}/resource`	`id*, iri*`	Obtiene el recurso con IRI `iri` de la API `apiId` usando el `id` del modelo
C5	`PUT`	`/apis/{apiId}/resource`	`id*, iri*`	Crea o reemplaza el recurso con IRI `iri` de la API `apiId` usando el `id` del modelo
C6	`PATCH`	`/apis/{apiId}/resource`	`id*, iri*`	Actualiza el recurso con IRI `iri` de la API `apiId` usando el `id` del modelo
C7	`DELETE`	`/apis/{apiId}/resource`	`id*, iri*`	Elimina el recurso con IRI `iri` de la API `apiId` usando el `id` del modelo
C8	`GET`	`/apis/{apiId}/resources`	`id*, iris*, ns, nspref`	Obtiene un conjunto de recursos con IRIs `iris` de la API `apiId` usando el `id` del modelo. Los IRIs pueden ser abreviados utilizando el prefijo `nspref` y el espacio de nombres `ns`
C9	`GET`	`/apis/{apiId}/query`	`id*`	Envía una consulta SPARQL parametrizada utilizando el `id` de la plantilla de la API `apiId` con los valores de los parámetros deseados

Fijándonos en la Tabla 2.3, las primeras llamadas (C0-3) se utilizan para la configuración de CRAFTS, desde listar todas las APIS disponibles (C0), hasta crear o modificar APIs existentes. Para crear una nueva API, es necesario mandar una configuración en el cuerpo de la petición C2, que sigue el esquema especificado en el Anexo A. Concretamente, la configuración es un fichero en formato JSON con las siguientes claves y valores:

- `apiId`: se corresponde con el identificador de la API. Tiene que ser único en todo el sistema.

- `endpoints`: consiste en un *array* de *endpoints* SPARQL que proporcionarán los datos abiertos enlazados a la API.
- `model`: conjunto de modelos de recursos que sirven para guiar los intercambios de datos con los *endpoints*. Cada solicitud de un recurso realizada a la API (llamadas C4-8) debe hacer referencia a un recurso concreto del modelo con un identificador `id`. El modelo de datos define un mapeo de datos RDF a un objeto JSON. Los diferentes campos sirven para mapear propiedades de tipo `datatype`, propiedades `object type` y pertenencia a clases (`rdf:type`) de RDF a objetos JSON
- `queryTemplates`: se corresponde con un *array* de plantillas de consulta SPARQL. Cada una de las plantillas contiene un `id`, un `endpoint` previamente configurado, un *array* de parámetros y de variables y, por último, una plantilla, que se corresponde con la consulta SPARQL parametrizada descrita con Mustache.

Siguiendo con la especificación de la Tabla 2.3, las llamadas C4-8 sirven para acceder fácilmente a los datos abiertos enlazados de los *endpoints* seleccionados. Todas ellas siguen una estructura general, válida para cualquier API CRAFTS: el parámetro `apiId` identifica la API seleccionada, el parámetro `id` especifica un modelo de recurso y, el parámetro `iri`, identifica el recurso RDF de destino. Dependiendo el tipo de operación escogida, se estará recuperando (GET), creando o reemplazando (PUT), actualizando (PATCH) o borrando (DELETE) el recurso. Las representaciones de los recursos serán incluidas en el cuerpo de la petición o de la respuesta en función del tipo de operación. Estas representaciones tendrán el formato especificado en el modelo de recursos al crear la API. Para el caso concreto de las actualizaciones parciales de recursos, se utiliza JSON Patch [Bry13a]. JSON Patch es un formato (identificado por el tipo de medio `application/json-patch+json`) para expresar una secuencia de operaciones que se aplicarán a un documento JSON; es idóneo para su uso con el método HTTP PATCH [Dus10], el cual extiende el protocolo HTTP con un método para realizar modificaciones parciales en los recursos. Un documento de tipo JSON PATCH está formado por un *array* de objetos JSON, en los que cada objeto representa una operación a realizar en el documento JSON especificado. Cada objeto presenta los siguientes campos:

- `op`: cuyo valor indica la operación a realizar. Existen seis tipos, que son: `add`, `remove`, `replace`, `move`, `copy` y `text`.
- `path`: cuyo valor es una cadena que contiene un *JSON-pointer* [Bry13b], que hace referencia a una ubicación dentro del documento JSON especificado donde se realizará la operación.
- `value`: en él se especifica el contenido que se desea añadir o reemplazar en el `path` especificado.

C8 permite la recuperación de un conjunto de recursos RDF en una sola llamada, por lo que puede emplearse para reducir el número de llamadas a C4. Por último, C9 está pensado para enviar fácilmente consultas SPARQL parametrizadas, así como para descubrir IRIs de recursos RDF que luego se utilizarán en las llamadas C4-8. Se verán ejemplos concretos de todas estas llamadas en el capítulo 4.

Arquitectura de CRAFTS

En la Figura 2.3 se muestra la arquitectura lógica de CRAFTS. Cuando se realiza una llamada a la API, lo primero que se hace es comprobar si cumple con la especificación de la

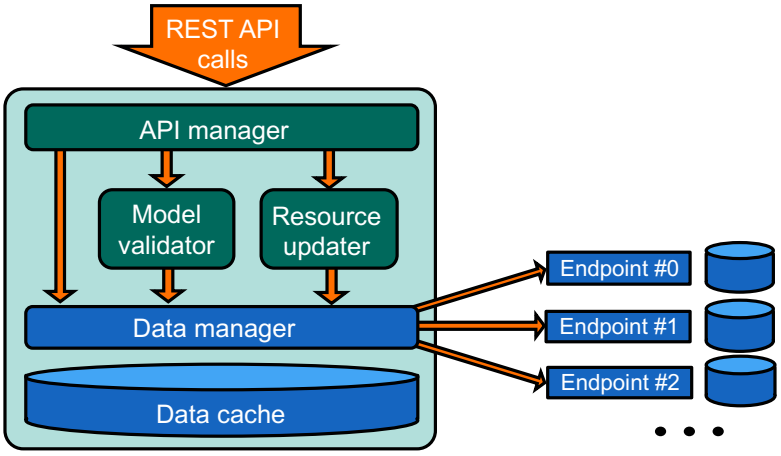


Figura 2.3: Arquitectura lógica de CRAFTS (Fuente: [VG21])

Tabla 2.3. Si es válida, se manda al gestor de la API (*API manager*). Dependiendo del tipo de llamada, el gestor puede depender de otros componentes (*Model validator* y *Resource updater*) para generar una respuesta. El *Model validator* se encarga de detectar errores en los cuerpos de las solicitudes de las llamadas recibidas, mientras que el *Resource updater* maneja las llamadas de escritura de recursos. Por último, el *Data manager* se encarga de gestionar todos los intercambios de datos con los *endpoints* SPARQL.

2.4. Discusión y conclusiones

El objetivo último del TFM es desarrollar una aplicación para la educación medioambiental. En el presente capítulo, se han analizado un par de aplicaciones afines a este objetivo, como son *Observation.org* e *Integrate Tree Microhabitat App*. Sin embargo, los datos que exponen ambas herramientas no siguen un modelo RDF, por lo que su futura integración en otras aplicaciones que hagan uso de datos abiertos enlazados no sería posible. Por ello, se ha presentado la posibilidad de usar las tecnologías de la Web Semántica para estructurar los datos de manera que estos puedan ser reutilizados. Como resultado, se cuenta con LOD forestales de gran interés para el desarrollo de EducaWood, como son los conjuntos de datos del IFN y del MFE, facilitados gracias al desarrollo del proyecto Cross-Forest.

Tras la decisión del uso de LOD para el proyecto, se hizo visible la necesidad de buscar nuevas estrategias que permitieran crear un puente entre las tecnologías semánticas y su uso sin necesidad de tener conocimiento de las mismas. La salida basada en triplas de los *endpoints* SPARQL puede ser una barrera para los desarrolladores que quieren integrar LOD en sus aplicaciones. Por ello, resulta de gran importancia para los proveedores de LOD disponer de un servicio de acceso a APIs alternativo, que permita su uso por parte de usuarios sin (o con poca) experiencia en tecnologías de la Web Semántica y en el lenguaje de consulta SPARQL. Se han investigado varias aproximaciones para conseguir APIs sobre puntos SPARQL, de las cuales finalmente CRAFTS fue la herramienta escogida para desarrollar Educawood, ya que solventa todos los problemas que el resto de opciones presentaban.

Capítulo 3

Planificación y Análisis del sistema

3.1. Introducción

Una vez abordados los problemas y conceptos teóricos que sirven de base para este TFM, se puede proceder a desarrollar el contenido referente a las fases iniciales del Ciclo de Vida del Desarrollo de Sistemas descritas en la sección 1.3 del Capítulo 1. Estas son: Planificación y Análisis. Para ello, se presenta en la Figura 3.1 una visión de la aplicación **EducaWood** completa. Se plantea una arquitectura multicapa compuesta por un *frontend* y un *backend*. El *backend* estará compuesto por herramientas ya implementadas de las que se hablará posteriormente, por lo que el análisis del presente capítulo estará centrado en el *frontend* de la aplicación web, siendo este al que nos referimos al hablar de "el sistema".

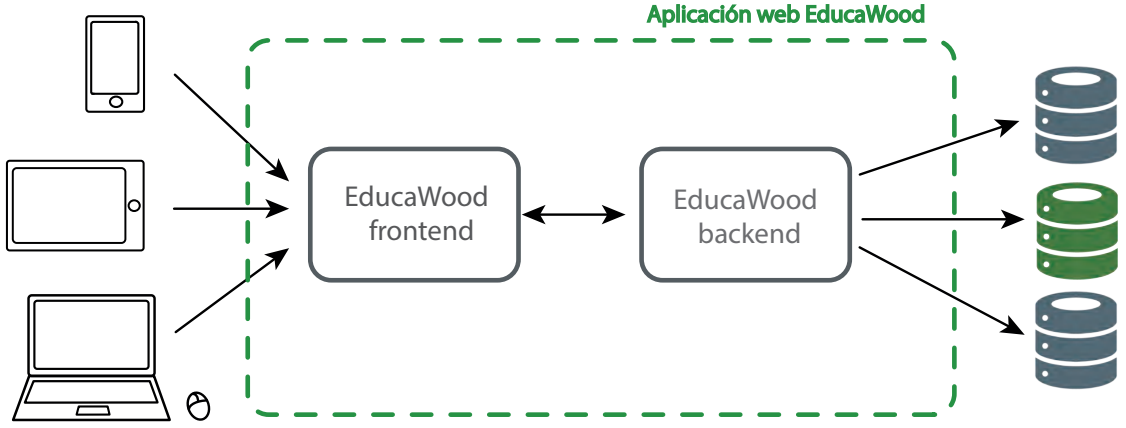


Figura 3.1: Visión de la aplicación completa

La **Planificación** constituye la fase inicial del SDLC. Principalmente abarca las actividades de identificación y selección de los objetivos que se pretenden conseguir, se propone un primer planteamiento del problema a resolver y se responde a las necesidades principales que debe cubrir el sistema [Val17a]. Como resultado de esta fase se ha elaborado la sección 3.2, la visión del sistema. Esta primera entrega se llevó a cabo a través de una lluvia de ideas entre todos los participantes del proyecto. Las ideas finales se recogen en un boceto esquematizado en la Figura 3.2, las cuales fueron motivadas por los aspectos mencionados en el Capítulo 2 y por las bases del concurso Desafío Aporta. Como resultado de este trabajo, se creó un vídeo promocional disponible en YouTube[1].

[1]https://www.youtube.com/watch?v=VlcjIYmq1Ao&t=5s

En las secciones sucesivas se recoge lo referente a la fase de **Análisis**, cuyo propósito es determinar qué información y qué servicios de procesamiento de la información son necesarios para cumplir con los objetivos y las funciones seleccionadas [Val17c]. En la sección 3.3 se recogen los requisitos del sistema, que consisten en la recopilación de las características que debe ofrecer el mismo. En la sección 3.4 se especifican los casos de uso, que sirven de ayuda para capturar las funcionalidades del sistema. El prototipo de la interfaz de usuario se muestra en la sección 3.5. En la sección 3.6 se presenta el modelo de datos del dominio de la aplicación. Para terminar el presente capítulo, en la sección 3.7 se realizan unas pequeñas conclusiones a cerca de la fase de análisis.

3.2. Visión del sistema

EducaWood pretende ser una aplicación web de carácter educativo, social y semántico, que facilite el intercambio de información relativa a los árboles entre los usuarios de la plataforma, siendo estos datos parte de la nube de datos abiertos enlazados. Los usuarios de la aplicación podrán navegar por un mapa interactivo que dará información relativa al uso del suelo y los árboles existentes en una zona. Además, la aplicación permitirá realizar anotaciones de árboles, con todas sus características: localización, especie, altura, diámetro, estado en el que se encuentra, imágenes, microhábitats, etc. EducaWood está pensada principalmente para uso educativo, aunque su uso divulgativo también estará cubierto. Los usuarios podrán registrarse si así lo desean. A los usuarios registrados en EducaWood se les denominará *Woodies*. De manera trasparente al usuario, la herramienta organizará y estructurará los datos manejados con vocabularios y herramientas de la Web Semántica. Sus aspectos más relevantes se recogen a continuación (ver Figura 3.2):

1. EducaWood está pensada para ser una herramienta educativa, que ayude a fomentar la educación medioambiental en prácticamente cualquier nivel educativo (principalmente desde primaria hasta grado universitario).

2. EducaWood sacará el aula al bosque. Se basa en principios de aprendizaje activo, con el que se construye conocimiento a partir de experiencias en la vida real, reflexiones sobre esas experiencias y su posterior experimentación activa (en el bosque o en el aula).

3. EducaWood utilizará fuentes de datos abiertos enlazados como base del aprendizaje medioambiental. Concretamente, datos del Mapa Forestal Español, que describe la cartografía de las masas forestales del territorio, datos del Inventario Forestal Español, el cual muestrea las especies arbustivas y arbóreas de todo el país y, por último, datos de la DBpedia, para aportar información enciclopédica.

4. EducaWood fomentará el uso de las TIC (Tecnologías de la Información y la Comunicación) en la educación.

5. EducaWood permitirá la generación de más datos medioambientales (nuevos árboles, microhábitats, madera muerta, etc.), de manera que se enriquecerán las fuentes de datos abiertos enlazados utilizadas con nuevas anotaciones sociales.

6. EducaWood permitirá la visualización de manera interactiva de los diferentes conjuntos de datos abiertos enlazados utilizados.

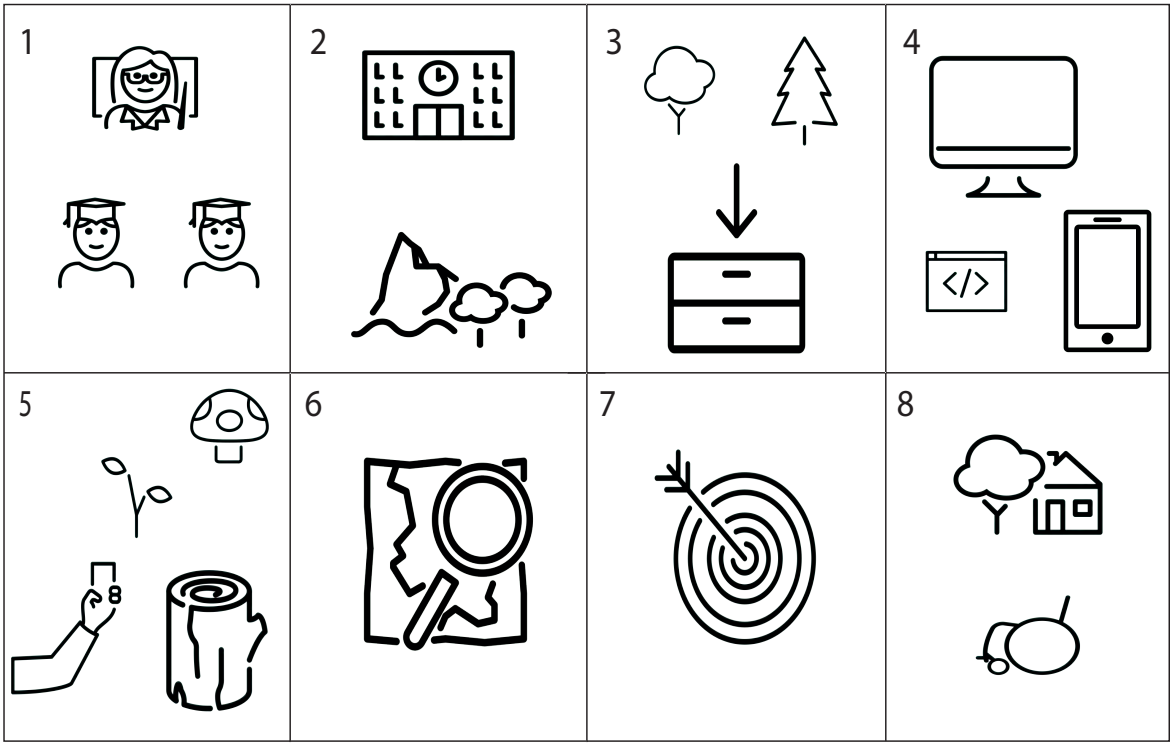


Figura 3.2: Lluvia de ideas inicial de EducaWood

7. La educación ambiental es una buena herramienta no sólo para fomentar el aprendizaje sobre el funcionamiento de la naturaleza, sino también para promocionar la toma de conciencia ecológica. Con EducaWood se pretende fomentar la toma de conciencia del papel mitigador que tienen los bosques ante el cambio climático y su importancia para la biodiversidad, en línea con el Objetivo 15[2] de la Agenda 2030.

8. EducaWood fomentará la calidad de vida de colectivos vulnerables. Estará prevista para su uso en cualquier dispositivo con un navegador web (incluidos móviles), por lo que los requisitos de uso son bastante modestos y asumibles por una gran parte de la ciudadanía. Además, estudiantes con movilidad reducida o de zonas aisladas podrán realizar visitas virtuales del bosque y ver las anotaciones que hayan hecho otros estudiantes de los árboles de su entorno más próximo (o lejano). Las actividades podrán realizarse de manera remota aprovechando las anotaciones creadas por la comunidad. Estas actividades se adaptarán adecuadamente a escenarios como el que actualmente tenemos como consecuencia de la COVID, en los que las actividades de educación presencial puedan no ser factibles.

Dado el carácter social de EducaWood, la herramienta presenta un sistema básico para curar los datos publicados en el que existen **anotaciones primarias**. Todos los árboles publicados por *Woodies* tendrán una anotación de cada tipo (especie, ubicación, altura,

[2]https://www.mdsocialesa2030.gob.es/agenda2030/index.htm

diámetro y estado) considerada la correcta, la anotación primaria. Esto supone que, inicialmente, un *Woodie* puede crear un árbol con sus diferentes anotaciones y estas pasarán a ser automáticamente las anotaciones primarias. A continuación, puede que otro *Woodie* se de cuenta de que una de esas anotaciones (de especie, por ejemplo), es incorrecta, por lo que este segundo usuario añadirá una nueva anotación de especie. El sistema estará configurado para que la última anotación añadida pase a ser la anotación primaria, de manera que las anotaciones que sean incorrectas se corregirán cuando algún usuario se de cuenta del error y corrija la anotación añadiendo una nueva.

La aplicación será accesible a través de un dispositivo con acceso a Internet y un navegador moderno. La interfaz de usuario será adaptativa, por lo que se ajustará de manera automática a la pantalla del dispositivo utilizado. Su prototipo se especifica en la sección 3.5.

3.3. Definición de requisitos

Como resultado de las primeras reuniones del equipo de EducaWood, se recopiló toda la información relacionada con las necesidades y las características que debía tener la aplicación propuesta. Con toda esta información, se especifican los Requisitos Funcionales (FRQ, *functional requirement*) y los Requisitos No Funcionales (NFRQ, *non-functional requirement*) del sistema. Ambos tipos de requisitos responden a las necesidades que tiene el sistema en torno a su funcionalidad, fiabilidad, rendimiento, soporte, diseño, implementación, integración y hardware (características FURPS+) [Ste15].

3.3.1. Requisitos Funcionales

Los requisitos funcionales son características del sistema que expresan una funcionalidad del mismo [Ste15]. En primer lugar, se plantean en forma de tabla para agrupar todas las funcionalidades que debe cumplir el sistema. A continuación, se tratan en profundidad con los casos de uso en la sección 3.4. Dentro de la aplicación, se pueden distinguir varios tipos de funcionalidades, que se han agrupado por tablas. En primer lugar, se tratan las características relacionadas con la gestión de usuarios (ver Tabla 3.1), las relacionadas con la gestión del mapa se recogen en la Tabla 3.2 y, por último, las relacionadas con la gestión de anotaciones se pueden consultar en la Tabla 3.3.

Tabla 3.1: Requisitos Funcionales del sistema: gestión de usuarios

Requisitos Funcionales	
ID	**Descripción**
FRQ001	El sistema deberá permitir a los usuarios registrarse en la aplicación, de manera que pasen a ser *Woodies*[3]. La información necesaria para darse de alta será el nombre de usuario, el correo electrónico y una contraseña. El sistema deberá mandar estos datos al servicio de autenticación para realizar el registro.
FRQ002	El sistema utilizará el servicio de autenticación para comprobar el correo electrónico del usuario mediante el envío de un correo de confirmación.

[3]Nombre que adquieren los usuarios registrados en EducaWood.

Continuación de la Tabla 3.1.	
ID	**Descripción**
FRQ003	El sistema deberá mandar los datos personales del usuario al servicio de autenticación para su almacenamiento.
FRQ004	El sistema deberá impedir la creación de más de una cuenta asociada a un mismo correo electrónico, indicando al usuario que ya existe una cuenta registrada con ese correo.
FRQ005	El sistema deberá permitir a los usuarios iniciar sesión en la aplicación introduciendo el correo electrónico y la contraseña. Los datos introducidos se mandarán al servicio de autenticación para comprobar si son correctos.
FRQ006	El sistema deberá permitir a los usuarios *Woodies* cerrar sesión.
FRQ007	El sistema deberá permitir a los usuarios de EducaWood darse de baja a través del servicio de autenticación. En tal caso, las anotaciones realizadas por dicho usuario permanecerían en la aplicación.
FRQ008	El sistema deberá permitir consultar los datos personales a los *Woodies*, estos serán: nombre de usuario, correo electrónico y foto. El sistema deberá obtenerlo del servicio de autenticación de EducaWood.
FRQ009	El sistema deberá permitir modificar los datos personales a los *Woodies* y mandarlos al servicio de autenticación para que sean actualizados.
FRQ010	El sistema deberá permitir cambiar la contraseña a los usuarios *Woodies*. Para ello deberá solicitar la contraseña actual, una nueva y mandarlo al servicio de autenticación para su comprobación y posterior actualización.
FRQ011	El sistema deberá facilitar a los usuarios *Woodies* recuperar su contraseña en caso de olvido, a través del servicio de autenticación. Se deberá mandar un correo electrónico a la cuenta del usuario para que pueda crear una nueva.

Tabla 3.2: Requisitos Funcionales del sistema: gestión del mapa

Requisitos Funcionales	
ID	**Descripción**
FRQ012	El sistema deberá obtener de CRAFTS las teselas de una zona específica y mostrarlas en un mapa.
FRQ013	El sistema deberá obtener de CRAFTS la información completa de una tesela específica y permitir visualizarla.
FRQ014	El sistema deberá obtener de CRAFTS los árboles existentes en el IFN de una zona específica y visualizarlos en un mapa.
FRQ015	El sistema deberá obtener de CRAFTS la información completa de un árbol concreto del IFN y mostrárselo al usuario.
FRQ016	El sistema deberá obtener de CRAFTS los árboles anotados por los *Woodies* de una zona específica y visualizarlos en un mapa.

Continuación de la Tabla 3.2.	
ID	**Descripción**
FRQ017	El sistema deberá obtener de CRAFTS la información completa de un árbol anotado por un *Woodie* y mostrárselo al usuario.

Tabla 3.3: Requisitos Funcionales: gestión de anotaciones.

Requisitos Funcionales	
ID	**Descripción**
FRQ018	El sistema deberá facilitar a los *Woodies* dar de alta un nuevo árbol en la aplicación y mandar los datos a CRAFTS para su almacenamiento, siendo necesario únicamente añadir una anotación de ubicación del árbol. El resto de atributos (ej. especie, altura, estado, etc.) son opcionales. Las imágenes correspondientes a las anotaciones de imagen se guardarán en el servicio de almacenamiento.
FRQ019	El sistema deberá permitir a un usuario *Woodie* anotar ubicaciones (coordenadas de latitud y longitud) de árboles (nuevos o existentes) y mandarlas a CRAFTS para su almacenamiento.
FRQ020	El sistema deberá permitir a un usuario *Woodie* anotar especies de árboles (nuevos o existentes) a partir de la taxonomía cargada en el sistema proveniente de la ontología del Cross-Forest[4] y mandar estas anotaciones a CRAFTS para su almacenamiento.
FRQ021	El sistema deberá permitir a un usuario *Woodie* añadir imágenes de árboles (nuevos o existentes), de partes de estos o de microhábitats, anotarlas y mandarlas a CRAFTS y al servicio de almacenamiento para su registro.
FRQ022	El sistema deberá permitir a un usuario *Woodie* anotar las medidas de un árbol (nuevos o existentes) y mandarlas a CRAFTS para su almacenamiento. Estas medidas son el diámetro y la altura.
FRQ023	El sistema deberá permitir a un usuario *Woodie* anotar el estado en el que se encuentra un árbol (nuevo o existente) a partir de los estados disponibles en la ontología del Anexo B y mandar la información a CRAFTS para su almacenamiento.
FRQ024	El sistema deberá permitir a un usuario *Woodie* anotar los microhábitats de un árbol (nuevo o existente) a partir de los diferentes tipos disponibles en la ontología del Anexo B, y mandar la información a CRAFTS para su almacenamiento.
FRQ025	El sistema deberá permitir a los usuarios visualizar las últimas anotaciones de árboles realizadas en la aplicación. Dicha información deberá solicitarse a CRAFTS.

[4] `https://github.com/Cross-Forest/Ontologies`

Continuación de la Tabla 3.3.	
ID	**Descripción**
FRQ026	El sistema deberá permitir a cualquier usuario de la aplicación completar o corregir la información de un árbol existente, añadiendo nuevas anotaciones. Tanto si se añade una anotación sobre una característica que no esté anotada previamente, como si se añade una anotación de algo ya anotado, esta última pasará a ser la anotación primaria, aquella que se considera válida.
FRQ027	El sistema deberá permitir a cualquier usuario de la aplicación escoger un árbol y ver su información completa. Esta información incluye las anotaciones primarias de ubicación, especie, diámetro, altura y estado (si existen), además del resto de anotaciones asociadas al mismo. El sistema deberá obtener dicha información de CRAFTS.
FRQ028	El sistema deberá transformar los datos de entrada del usuario con información relativa a los árboles en llamadas a la API de CRAFTS para que puedan ser almacenados correctamente como datos abiertos enlazados.

3.3.2. Requisitos no Funcionales

Un Requisito no Funcional es una descripción sobre la calidad del comportamiento del sistema o sobre sus restricciones a la hora de producir un resultado. Con carácter general, se centran en las características de rendimiento, fiabilidad y seguridad del sistema analizado [Ste15]. En la Tabla 3.4 se recopilan los requisitos no funcionales del sistema.

Tabla 3.4: Descripción de los Requisitos no Funcionales del sistema.

Requisitos no Funcionales	
ID	**Descripción**
NFRQ001	El sistema deberá permitir anotaciones semánticas de árboles.
NFRQ002	El sistema deberá controlar que los usuarios solo puedan visualizar los datos públicos de otros usuarios registrados.
NFRQ003	El sistema deberá controlar que los usuarios registrados solo puedan modificar datos de usuario de su propia cuenta.
NFRQ004	El sistema deberá permitir el uso de posiciones absolutas y definidas en coordenadas del sistema de coordenadas WGS84 [NGA].
NFRQ005	El sistema deberá intercambiar la información con una API de CRAFTS en formato JSON.
NFRQ006	El sistema deberá presentar una interfaz amigable y fácil de usar, de manera que ofrecerá una interfaz web basada en formularios y no requerirá conocimientos de Web Semántica ni de bases de datos.
NFRQ007	El sistema deberá funcionar y visualizarse correctamente en la última versión de los navegadores Mozilla Firefox, Google Chrome y Safari.

Continuación de la Tabla 3.4.	
ID	**Descripción**
NFRQ008	La interfaz de usuario deberá ser adaptable a cualquier dispositivo utilizado, desde móviles (principalmente) hasta tabletas u ordenadores de sobremesa.
NFRQ009	El sistema deberá presentar la interfaz de usuario en español.
NFRQ010	El sistema deberá solucionar localmente los fallos que puedan deberse a algún servicio externo del *backend* (CRAFTS, servicio de almacenamiento y servicio de autenticación) e informar adecuadamente al usuario de lo ocurrido.
NFRQ011	El sistema deberá resolver peticiones de lectura en tiempos inferiores a 1 segundo para garantizar la fluidez de la aplicación..
NFRQ012	El sistema deberá resolver peticiones de creación en tiempos inferiores a 5 segundos para garantizar la fluidez de la aplicación.
NFRQ013	El sistema deberá permitir una navegación fluida por el mapa. La latencia máxima entre interacciones no deberá ser superior a 1 segundo.
NFRQ014	El sistema deberá presentar un manual de usuario que explique el funcionamiento de la aplicación.
NFRQ015	El sistema requerirá conexión a Internet.

3.4. Casos de uso

Tras especificar los requisitos del sistema, se documentan los diferentes casos de uso del mismo, desde un punto de vista brevemente descriptivo. Un Caso de Uso [Ste15] describe el comportamiento del sistema al afrontar un requisito funcional, enfatizando el valor proporcionado por el sistema a sus actores y la relación entre ellos. La descripción de los mismos viene acompañada de un Diagrama de Casos de Uso, que ayuda a representar la manera en la que los diferentes actores interactúan con el sistema. Los diagramas de casos de uso se han agrupado siguiendo la división de funcionalidades de la sección anterior. En primer lugar se tratan los casos de uso relacionados con la gestión de usuarios (ver sección 3.4.1), la gestión de funcionalidades del mapa se trata en la sección 3.4.2 y lo relacionado con la gestión de anotaciones en la sección 3.4.3. Finalmente, en la sección 3.4.4, se puede observar el diagrama de casos de uso de la gestión de actividades. Las funcionalidades del sistema allí descritas no se recogen en los requisitos del sistema, puesto que no forman parte del ámbito del TFM. Sin embargo, se consideran relevantes puesto que puede que tengan cabida en la futura continuación de la aplicación.

3.4.1. Gestión de usuarios

En la Figura 3.3 se muestra el diagrama de casos de uso relacionado con la gestión de los usuarios que utilizan EducaWood. Los casos de uso aquí descritos se corresponden con los requisitos funcionales FRQ001 al FRQ011 de la sección 3.3.1. La descripción de los mismos se detalla a continuación:

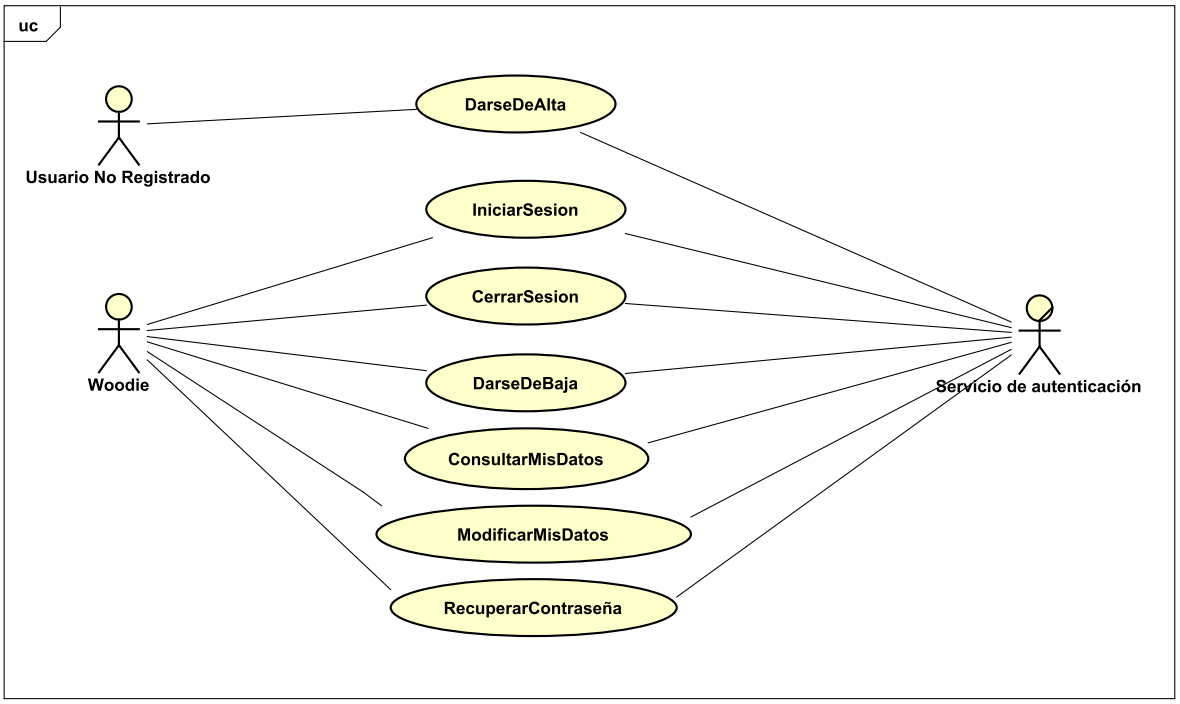


Figura 3.3: Diagrama de Casos de Uso: gestión de usuarios

CU001: *DarseDeAlta*

- **Descripción breve.** Un usuario se registra por primera vez en EducaWood.
- **Precondición.** Ninguna.
- **Descripción paso a paso.**
  1. El usuario no registrado selecciona la opción "Registrarse" del menú de navegación.
  2. El sistema solicita el nombre de usuario, el correo electrónico y la contraseña.
  3. El usuario introduce los datos solicitados.
  4. El sistema manda los datos al servicio de autenticación.
  5. El servicio de autenticación comprueba que no existe ya ningún usuario con ese correo electrónico y, en caso afirmativo, crea la cuenta de usuario y manda un correo electrónico de verificación. En caso contrario, solicita al sistema que el usuario introduzca unos nuevos datos.
  6. El sistema traslada al usuario a la página de espera de verificación de la cuenta.
  7. El usuario confirma su cuenta en su correo electrónico.
- **Postcondición.** El usuario ha sido registrado en la aplicación.

CU002: *IniciarSesion*

- **Descripción breve.** Un usuario valida su identidad en la aplicación.
- **Precondición.** El usuario está registrado en el sistema.

- **Descripción paso a paso.**

  1. El usuario selecciona la opción “Iniciar Sesión” del menú de navegación.
  2. El sistema solicita el correo electrónico y la contraseña.
  3. El usuario introduce los datos solicitados.
  4. El sistema envía los datos introducidos al servicio de autenticación para comprobar que son correctos.
  5. El servicio de autenticación manda la confirmación o negación del acceso con los datos del usuario introducidos.
  6. El sistema, en caso afirmativo, traslada al usuario a la página principal de EducaWood. En caso contrario, solicita al usuario volver a introducir los datos.

- **Postcondición.** El usuario ha iniciado sesión.

CU003: *CerrarSesion*

- **Descripción breve.** Un usuario quiere cerrar sesión en EducaWood.
- **Precondición.** El usuario está registrado en el sistema y ha iniciado sesión.
- **Descripción paso a paso.**

  1. El usuario selecciona la opción “Cerrar Sesión” del menú de navegación.
  2. El sistema cierra la sesión actual y traslada al usuario a la página de inicio de la aplicación.

- **Postcondición.** El usuario ha cerrado su sesión.

CU004: *DarseDeBaja*

- **Descripción breve.** Un usuario desea darse de baja de EducaWood.
- **Precondición.** El usuario está registrado en el sistema y ha iniciado sesión.
- **Descripción paso a paso.**

  1. El usuario selecciona la opción “Darse de baja del sistema” dentro del apartado de Ajustes.
  2. El sistema pregunta al usuario si está seguro que quiere darse de baja, indicándole que toda su información personal se borrará de la aplicación.
  3. El usuario confirma la acción.
  4. El sistema solicita al servicio de autenticación borrar al usuario del registro de usuarios.
  5. El servicio de autenticación elimina la cuenta de usuario.

- **Postcondición.** El usuario y toda su información personal han sido eliminadas de la aplicación.

CU005: *ConsultarMisDatos*

- **Descripción breve.** Un usuario desea consultar sus datos personales.
- **Precondición.** El usuario está registrado en el sistema y ha iniciado sesión.
- **Descripción paso a paso.**
  1. El usuario accede al apartado de "Ajustes" de la aplicación.
  2. El sistema solicita al servicio de autenticación los datos personales del usuario.
  3. El servicio de autenticación proporciona los datos solicitados.
  4. El sistema muestra los datos personales del usuario: nombre de usuario y correo electrónico.
- **Postcondición.** Ninguna.

CU006: *ModificarMisDatos*

- **Descripción breve.** Un usuario desea modificar sus datos personales guardados en la aplicación.
- **Precondición.** El usuario está registrado en el sistema y ha iniciado sesión.
- **Descripción paso a paso.**
  1. El usuario accede al apartado de "Editar Perfil", dentro del menú de Ajustes.
  2. El sistema solicita al servicio de autenticación los datos personales del usuario.
  3. El servicio de autenticación proporciona los datos solicitados.
  4. El sistema muestra los datos devueltos dentro del apartado "Datos Personales".
  5. El usuario modifica los datos que desea (nombre de usuario, correo electrónico y/o contraseña) y pulsa "Modificar datos".
  6. El sistema recoge las modificaciones realizadas y las manda al servicio de autenticación de EducaWood.
  7. El servicio de autenticación registra los datos modificados.
  8. El sistema confirma la acción y muestra por pantalla los datos actualizados.
- **Postcondición.** El usuario ha podido actualizar su información personal registrada en la aplicación.

CU007: *RecuperarContraseña*

- **Descripción breve.** Un usuario desea recuperar su contraseña para poder iniciar sesión en la aplicación.
- **Precondición.** El usuario está registrado en el sistema.
- **Descripción paso a paso.**
  1. El usuario accede al apartado de "He olvidado mi contraseña", dentro de la página de Inicio de Sesión.
  2. El sistema solicita el correo electrónico del usuario.

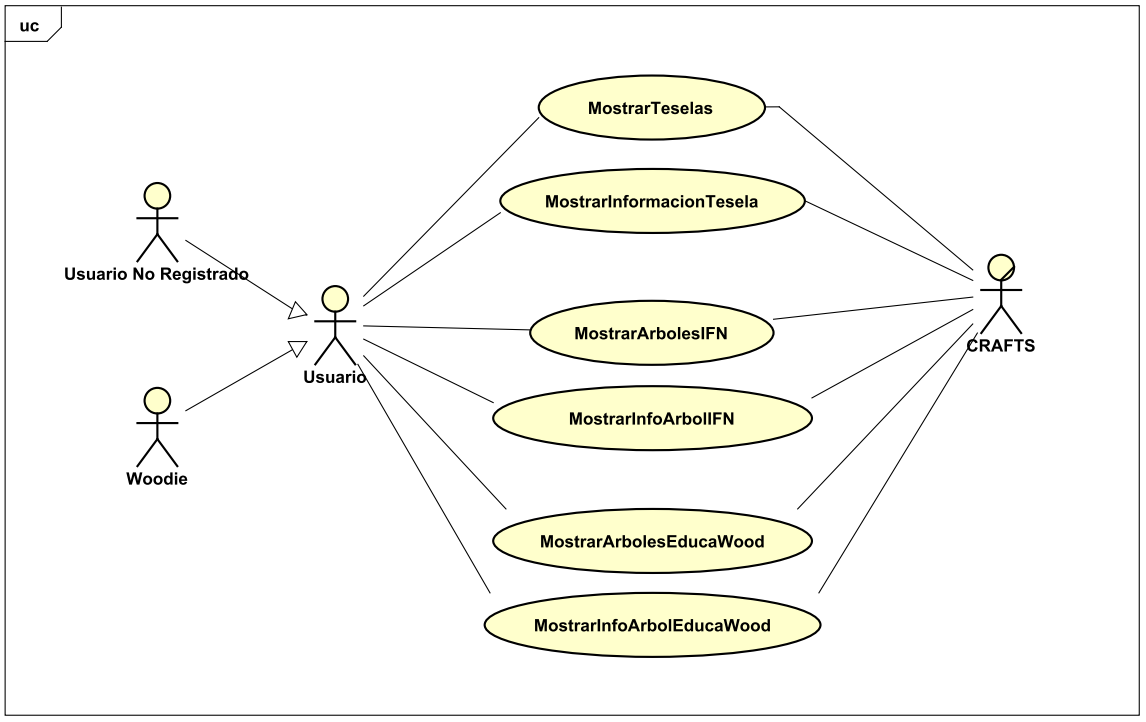


Figura 3.4: Diagrama de Casos de Uso: gestión del mapa

3. El usuario introduce su correo electrónico y selecciona “Recuperar contraseña”.
4. El sistema envía los datos al servicio de autenticación.
5. El servicio de autenticación comprueba que hay una cuenta de usuario asignada a ese correo electrónico y, en caso afirmativo, envía un correo electrónico a dicha cuenta con un enlace para poder crear una nueva contraseña.
6. El sistema indica por pantalla al usuario de que el correo se ha enviado con éxito.
7. El usuario accede a su correo electrónico y restablece su contraseña a través del enlace adjunto en el correo.
8. El servicio de autenticación actualiza la contraseña del usuario.

- **Postcondición.** El usuario ha podido restablecer la contraseña de su cuenta en EducaWood.

3.4.2. Gestión del mapa

La Figura 3.4 muestra el diagrama de casos de uso relacionado con la gestión de las actividades que se pueden realizar en el mapa interactivo de la aplicación. Los casos de uso aquí descritos se corresponden con los requisitos funcionales FRQ012 al FRQ017 de la sección 3.3.1. La descripción de los mismos se detalla a continuación:

CU008: *MostrarTeselas*

- **Descripción breve.** Un usuario desea consultar las teselas de una zona.

- **Precondición.** Ninguna.
- **Descripción paso a paso.**
  1. El usuario navega por el mapa hasta encontrar la zona de interés y pulsa el botón de "Cargar Teselas".
  2. El sistema solicita a CRAFTS las teselas de la zona, proporcionándole las coordenadas en las que se encuentra el mapa.
  3. CRAFTS devuelve las teselas existentes en la zona del mapa seleccionada.
  4. El sistema pinta las teselas en el mapa, siguiendo un código de colores específico para cada tipo de tesela y almacena la información de cada una localmente.
- **Postcondición.** El sistema tiene almacenada la información correspondiente a las teselas de la zona seleccionada.

CU009: *MostrarInformacionTesela*

- **Descripción breve.** Un usuario desea consultar la información de una tesela.
- **Precondición.** El caso de uso *MostrarTeselas* ha finalizado con éxito.
- **Descripción paso a paso.**
  1. El sistema muestra una zona del mapa con las teselas existentes.
  2. El usuario selecciona la tesela que desea consultar.
  3. El sistema muestra la información de la tesela seleccionada.
- **Postcondición.** Ninguna.

CU010: *MostrarArbolesIFN*

- **Descripción breve.** Un usuario desea consultar los árboles existentes en el IFN de una zona determinada.
- **Precondición.** Ninguna.
- **Descripción paso a paso.**
  1. El usuario navega por el mapa hasta encontrar la zona de interés y pulsa el botón de "Cargar árboles IFN".
  2. El sistema solicita a CRAFTS los árboles existentes en dicha zona, proporcionándole las coordenadas en las que se encuentra el mapa.
  3. CRAFTS devuelve los árboles solicitados.
  4. El sistema pinta de color verde los árboles en el mapa y almacena la información de cada uno localmente.
- **Postcondición.** El sistema tiene almacenada la información correspondiente a los árboles del IFN de la zona seleccionada.

CU011: *MostrarInfoArbolIFN*

- **Descripción breve.** Un usuario desea consultar la información de un árbol del IFN específico.
- **Precondición.** El caso de uso *MostrarArbolesIFN* ha finalizado con éxito.
- **Descripción paso a paso.**
  1. El sistema muestra una zona del mapa con las teselas y los árboles del IFN existentes.
  2. El usuario selecciona el árbol del IFN que desea consultar.
  3. El sistema muestra la información del árbol seleccionado.
- **Postcondición.** Ninguna.

CU012: *MostrarArbolesEducaWood*

- **Descripción breve.** Un usuario desea consultar los árboles que han sido creados en EducaWood de una zona determinada.
- **Precondición.** Ninguna.
- **Descripción paso a paso.**
  1. El usuario navega por el mapa hasta encontrar la zona de interés y pulsa el botón de "Cargar árboles EducaWood".
  2. El sistema solicita a CRAFTS los árboles existentes en dicha zona, proporcionándole las coordenadas en las que se encuentra el mapa.
  3. CRAFTS devuelve los árboles solicitados.
  4. El sistema pinta de color morado los árboles en el mapa y almacena la información de cada uno localmente.
- **Postcondición.** El sistema tiene almacenada la información correspondiente a los árboles de tipo EducaWood de la zona seleccionada.

CU013: *MostrarInfoArbolEducaWood*

- **Descripción breve.** Un usuario desea consultar la información de un árbol de tipo EducaWood específico.
- **Precondición.** El caso de uso *MostrarArbolesEducaWood* ha finalizado con éxito.
- **Descripción paso a paso.**
  1. El sistema muestra una zona del mapa con las teselas y los árboles de tipo EducaWood existentes.
  2. El usuario selecciona desde el mapa el árbol que desea consultar.
  3. El sistema muestra la información del árbol seleccionado.
- **Postcondición.** Ninguna

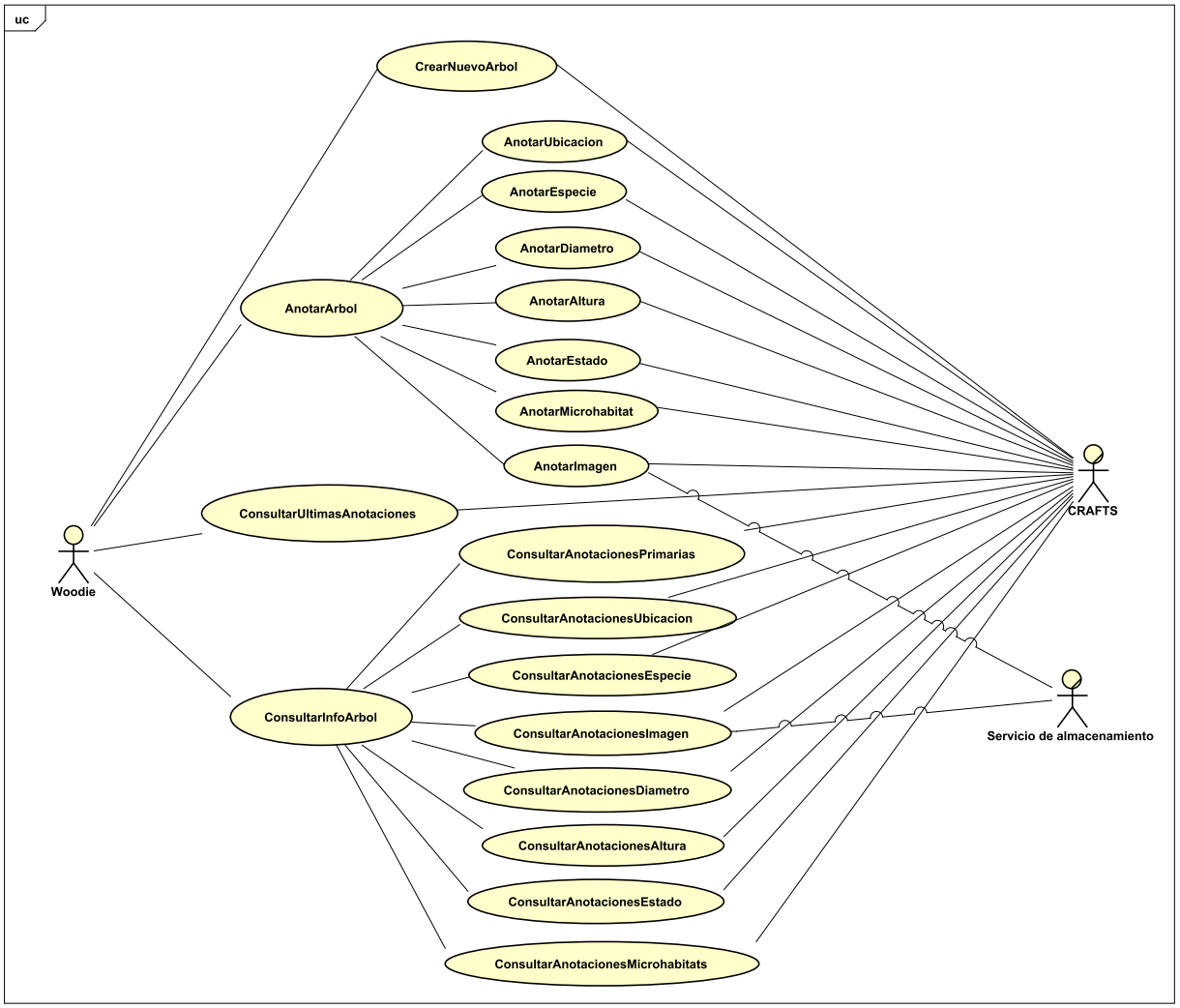


Figura 3.5: Diagrama de Casos de Uso: gestión de anotaciones

3.4.3. Gestión de anotaciones

En la Figura 3.5 se muestra el diagrama de casos de uso relacionado con la gestión de las anotaciones realizadas en EducaWood. Los casos de uso aquí descritos se corresponden con los requisitos funcionales FRQ018 al FRQ028 de la sección 3.3.1. La descripción de los mismos se detalla a continuación:

CU014: *CrearNuevoArbol*

- **Descripción breve.** Un usuario Woodie desea añadir un nuevo árbol en la aplicación.
- **Precondición.** El usuario ha iniciado sesión en EducaWood.
- **Descripción paso a paso.**

  1. El usuario selecciona la opción “Crear nuevo árbol” dentro de la barra del menú de navegación.

2. El sistema solicita la ubicación del árbol como dato obligatorio para su creación. Adicionalmente se pueden añadir otro tipo de anotaciones explicadas en el caso de uso *AnotarArbol*.
3. El usuario introduce las coordenadas de latitud y longitud (y el resto de campos opcionalmente).
4. El sistema comprueba que se ha introducido la ubicación correctamente y, en caso afirmativo, adecúa los datos y los envía a CRAFTS para su almacenamiento.
5. CRAFTS confirma que la acción se ha completado con éxito.
6. El sistema traslada al usuario a la página que muestra los datos del nuevo árbol creado.

- **Postcondición.** Un nuevo árbol de tipo EducaWood ha sido registrado en el sistema.

CU015-21: *AnotarArbol*

- **Descripción breve.** El usuario desea añadir una nueva anotación a un árbol existente o a uno que se va a crear nuevo.
- **Precondición.** El usuario ha iniciado sesión en EducaWood.
- **Descripción paso a paso.**

1. El usuario está visualizando la información completa de un árbol o creando uno nuevo y selecciona la opción "Añadir anotación".
2. El sistema solicita los datos necesarios en función de la anotación:
   a) (CU016: *AnotarUbicacion*) Para la anotación de ubicación, solicita las coordenadas geográficas: latitud y longitud.
   b) (CU017: *AnotarEspecie*) Para la anotación de especie, muestra una lista con todas las especies disponibles en el sistema.
   c) (CU018: *AnotarImagen*) Para la anotación de imagen, solicita la imagen y, adicionalmente, una descripción de la misma, un título y la parte del árbol que muestra.
   d) (CU019: *AnotarDiamentro*) Para la anotación de diámetro, solicita un número decimal en centímetros.
   e) (CU020: *AnotarAltura*) Para la anotación de altura, solicita un número decimal en metros.
   f) (CU021: *AnotarEstado*) Para la anotación de estado, muestra una lista con todos los estados disponibles descritos en la ontología.
   g) (CU022: *AnotarMicrohabitat*) Para la anotación de microhábitat, muestra una lista con todos los tipos disponibles en la ontología y solicita una imagen del mismo.
3. El usuario introduce los campos solicitados y pulsa a "Confirmar anotación".
4. El sistema adecúa los datos introducidos y los envía a CRAFTS para su almacenamiento. En el CU018, el sistema deberá mandar la imagen al servicio de almacenamiento.
5. El *backend* confirma que la acción se ha completado con éxito.

6. El sistema confirma la acción por pantalla al usuario.

- **Postcondición.** Se ha registrado una nueva anotación a un árbol del sistema.

CU022: *ConsultarUltimasAnotaciones*

- **Descripción breve.** Un usuario desea ver las últimas anotaciones de árboles realizadas en la aplicación.
- **Precondición.** Ninguna.
- **Descripción paso a paso.**

1. El usuario accede al apartado de "Últimas anotaciones".
2. El sistema solicita a CRAFTS los últimos 10 árboles añadidos en la aplicación.
3. CRAFTS devuelve la información solicitada.
4. El sistema muestra al usuario los árboles devueltos y le da la opción de ver más.
5. En el caso en el que el usuario desee ver más, seleccionará la opción "Ver más" y se volverá al paso 2, modificando el *offset* de la consulta realizada CRAFTS.

- **Postcondición.** Ninguna.

CU023-30: *ConsultarInfoArbol*

- **Descripción breve.** Un usuario *Woodie* desea ver la información completa de un árbol del sistema.
- **Precondición.** El usuario ha iniciado sesión en EducaWood.
- **Descripción paso a paso.**

1. El usuario accede al apartado "Información del árbol" a través del mapa o bien a través de la URL con el identificador del árbol.
2. El sistema solicita al CRAFTS toda la información del árbol.
   a) El CU024 (*ConsultarAnotacionesPrimarias*) para obtener las anotaciones primarias de ubicación, especie, estado, altura y diámetro.
   b) El CU025 (*ConsultarAnotacionesUbicacion*) para obtener todas las anotaciones de ubicación del árbol que se hayan añadido.
   c) El CU026 (*ConsultarAnotacionesEspecie*) para obtener todas las anotaciones de especie del árbol que se hayan añadido.
   d) El CU027 (*ConsultarAnotacionesImagen*) para obtener todas las anotaciones de imágenes del árbol que se hayan añadido. El sistema deberá adicionalmente consultar el servicio de almacenamiento para cargar las imágenes.
   e) El CU028 (*ConsultarAnotacionesDiametro*) para obtener todas las anotaciones de diámetro del árbol que se hayan añadido.
   f) El CU029 (*ConsultarAnotacionesAltura*) para obtener todas las anotaciones de altura del árbol que se hayan añadido.
   g) El CU030 (*ConsultarAnotacionesEstado*) para obtener todas las anotaciones de estado del árbol que se hayan añadido.

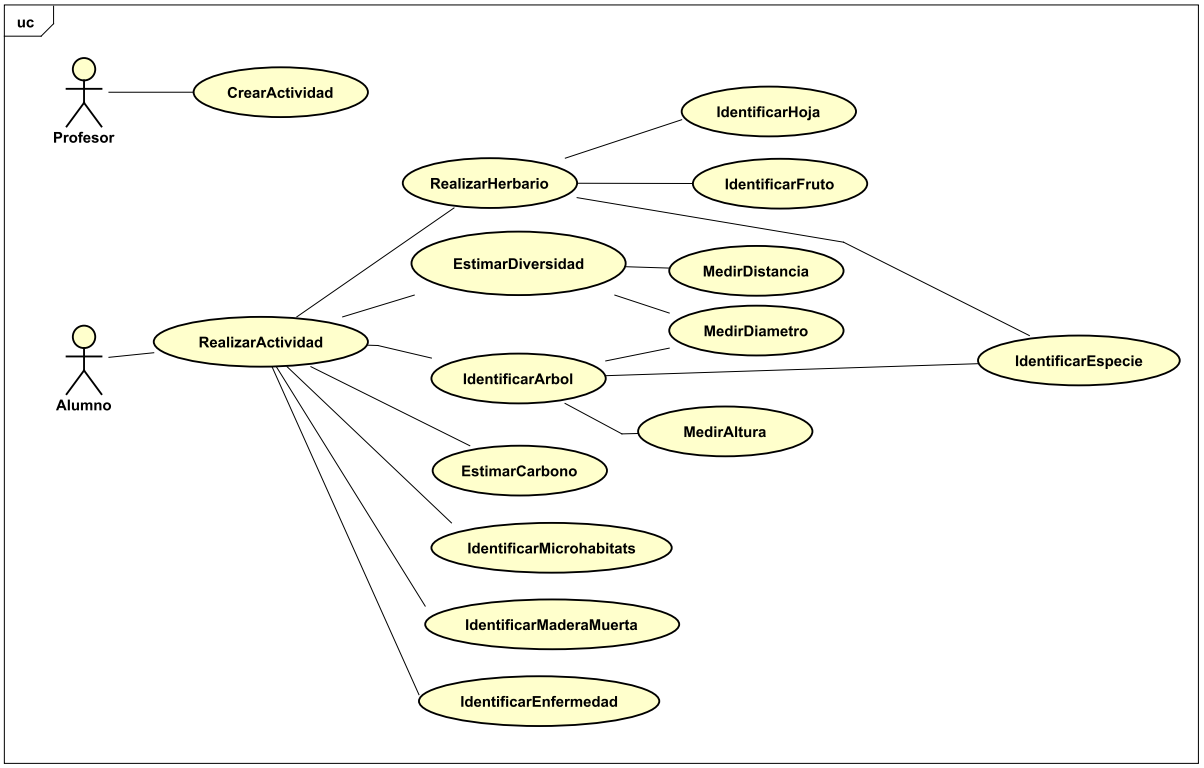


Figura 3.6: Diagrama de Casos de Uso: gestión de actividades

h) El CU031 (*ConsultarAnotacionesMicrohabitats*) para obtener todas las anotaciones de microhábitats del árbol que se hayan añadido.

3. CRAFTS y el servicio de almacenamiento devuelven toda la información del árbol solicitado que haya disponible.
4. El sistema muestra al usuario las diferentes anotaciones del árbol.

- **Postcondición.** Ninguna.

3.4.4. Gestión de actividades

En la Figura 3.6 se muestra el diagrama de casos de uso relacionado con la gestión de actividades. Estas actividades fueron ideadas y propuestas en el Desafío Aporta, aunque sus casos de uso no han sido implementados en esta primera versión de la aplicación. Se han incluido puesto que forman parte del trabajo futuro.

3.5. Prototipo de la interfaz de usuario

Existen numerosos métodos que se pueden utilizar en la fase de análisis para determinar los requisitos del sistema. El núcleo principal de esta fase es la recopilación de información [Val17a]. Una de las mejores maneras de conseguir este objetivo, es hablando con las personas que están directa o indirectamente relacionadas con la herramienta a desarrollar: potenciales usuarios, gestores, financiadores, expertos del dominio, etc. Otra forma de conocer el sistema actual es obteniendo copias de la documentación relativa a los sistemas y procesos empresariales actuales relacionados con el ámbito de desarrollo. A

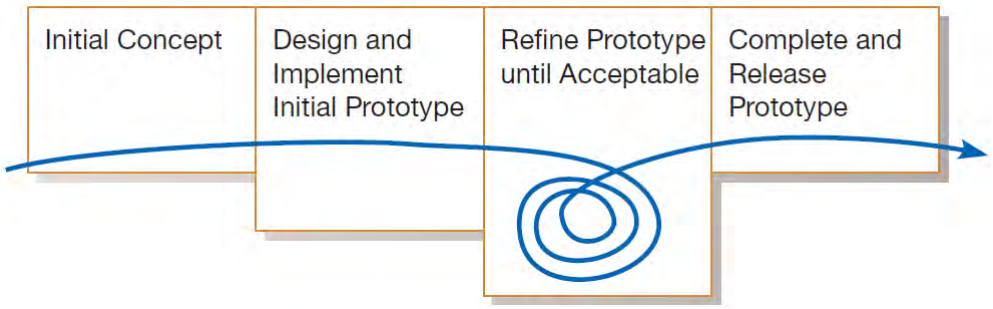


Figura 3.7: Modelo del prototipo evolutivo. (Fuente: [Val17a])

pesar de que estos métodos son tradicionales, siguen usándose a día de hoy para recopilar información debido a su gran utilidad. Sin embargo, existen otros métodos que han surgido posteriormente que también resultan ser una manera interesante y distinta de reunir información. Este es el caso de los prototipos. Crear prototipos [Val17a] de forma iterativa permite perfeccionar la comprensión de los requisitos del sistema en términos concretos, mostrando versiones de funcionamiento de las características del mismo. En el caso concreto de EducaWood, se han utilizado conjuntamente todos estos métodos para elaborar la fase de análisis. Los métodos tradicionales se usaron principalmente con los expertos del dominio forestal, puesto que su experiencia propia nos sirvió de gran ayuda para diseñar las diferentes funcionalidades del sistema. Por otro lado, se usó la creación de prototipos para madurar las ideas.

La creación de prototipos es un método iterativo y evolutivo en el que se empieza por modelar partes del sistema objetivo y, si el proceso tiene éxito, se hace evolucionar el sistema incluyendo lo nuevo. En la Figura 3.7 se ilustra este proceso. Un aspecto clave de este enfoque es que el prototipo se convierte en el sistema de producción real. Por ello, se optó por emplear una herramienta que pudiera reflejar prácticamente de manera efectiva cómo sería el aspecto de la aplicación tras su implementación. La herramienta utilizada fue Figma[5]. Figma es un editor de gráficos vectorial y una herramienta de generación de prototipos basada principalmente en la web, aunque también dispone de características *off-line* habilitadas para la aplicación de escritorio de macOs y Windows. Es un software propietario, aunque dispone de un plan gratuito con múltiples funcionalidades, lo suficiente para la elaboración del prototipo de EducaWod.

La creación del prototipo no solo sirvió para idear el aspecto de la interfaz de usuario, sino que también ayudó a unificar todas las funcionalidades presentadas en una misma herramienta, aclarar ciertos aspectos de algunos requisitos que no estaban claros y asegurar que la interfaz completa cumplía con las especificaciones del sistema. En la Figura 3.8 se muestra el prototipo final diseñado, donde podemos observar los diferentes casos de uso representados:

- En la **página principal** se representa el caso de uso referente a la visualización de las últimas anotaciones (CU023).
- En la página de **inicio de sesión** se representan los casos de uso de inicio de sesión (CU002) y el de registrarse (CU001).

[5] `https://www.figma.com/`

(a) Página principal

(b) Inicio de Sesión

(c) Ajustes

(d) Navegación por el mapa

(e) Creación de un nuevo árbol

(f) Detalles de un árbol

Figura 3.8: Prototipo inicial de la interfaz de usuario de EducaWood utilizando Figma.

- En la página de **ajustes** se representan la mayoría de casos de uso relacionados con la gestión de usuarios: cerrar sesión, darse de baja, consultar mis datos, modificar mis datos, cambiar contraseña y recuperar contraseña (CU003-08).

- En la página del **mapa** se representan todos los casos de uso referentes a la gestión del mapa (CU009-14).

- En la página de **crear un nuevo árbol** están representados todos los casos de uso que implican anotación (CU016-22).

- En la página de los **detalles de un árbol** se representan los casos de uso relacionados con consultar la información relativa a un árbol (CU024-31).

3.6. Modelo de dominio

El modelo de dominio permite representar los datos y las relaciones entre ellos de un dominio específico. En este caso en particular, nos servirá para identificar los principales conceptos del sistema y su relación con la ontología diseñada en el dominio forestal. Para el desarrollo de esta fase fue indispensable la colaboración de los ingenieros forestales del proyecto. El resultado obtenido es el diagrama de clases de la Figura 3.9. Cada una de las clases representa los diferentes conceptos manejados en EducaWood. Las principales son: `Usuario`, `Arbol`, `Anotación`, `Microhábitat` y `Mapa`. En el primer compartimento de las cajas se encuentra el nombre de los conceptos, mientras que en el segundo se enumeran sus propiedades. Estas deberán tenerse en cuenta a la hora de diseñar la ontología. Las relaciones entre los diferentes conceptos se representan mediante flechas junto con su cardinalidad (representada con números en los extremos de las mismas). Las flechas huecas hacen referencia a especializaciones de conceptos, por ejemplo, una anotación pueden ser de muchos tipos. Todos estos tipos son especializaciones de la clase `Anotación`.

3.7. Discusión y conclusiones

Durante la fase de análisis se debatió un problema existente en la mayoría de aplicaciones que implican anotación social, el curado de los datos. EducaWood inicialmente no cuenta con expertos del dominio forestal que puedan validar la información publicada por sus usuarios, por lo que esta información puede no ser válida. Para solventar este problema, se debatieron dos posibles aproximaciones. La primera de ellas consistiría en implementar una jerarquía de usuarios y votaciones, de manera que las anotaciones válidas fueran las que más votaciones obtengan. Además, estarían ponderadas según la posición en la que se encuentre el usuario que vota. Este método requeriría de mayor tiempo de implementación, por lo que se decidió utilizar el método de actualización automática de anotaciones válidas consistente en que los árboles publicados en EducaWood tengan **anotaciones primarias** de especie, ubicación, altura, diámetro y estado. Estas anotaciones serán las últimas realizadas en cada momento. Este mecanismo se refleja en la ontología presentada en el capítulo siguiente. Las propiedades de la misma para soportar esto se comentarán con mayor detalle entonces. De igual manera, se presentará un ejemplo de uso concreto en el Capítulo 5.

Con la finalización del proceso de análisis se ha comprendido de una manera más profunda y completa el dominio de aplicación de nuestro sistema software, proporcionando una visión general de la aplicación propuesta, identificando a los diferentes usuarios que van a interactuar con ella, estableciendo claramente la frontera del sistema con los diferentes servicios que componen el *backend* e identificando todas las funcionalidades que debe satisfacer la aplicación. Estas están agrupadas en tres tipos claramente diferenciados, como se ha visto en la definición de requisitos funcionales y los casos de uso. El diseño e implementación deberá cubrir todas ellas, teniendo adicionalmente en cuenta los requisitos no funcionales aquí descritos. La integración del sistema con los diferentes componentes del *backend* cobrará especial relevancia, puesto que son necesarios para el correcto desarrollo de la misma. Puede parecer que se trata de una fase poco productiva, puesto que durante ella no se encuentra ningún avance en el desarrollo software como tal. Sin embargo, es muy importante invertir tiempo y esfuerzos en llevar a cabo una buena planificación y análisis, ya que servirán de base para las fases posteriores.

Finalizadas estas primeras fases, podemos dar paso a las siguientes. En el siguiente capítulo se abordarán las fases de Diseño e Implementación de EducaWood.

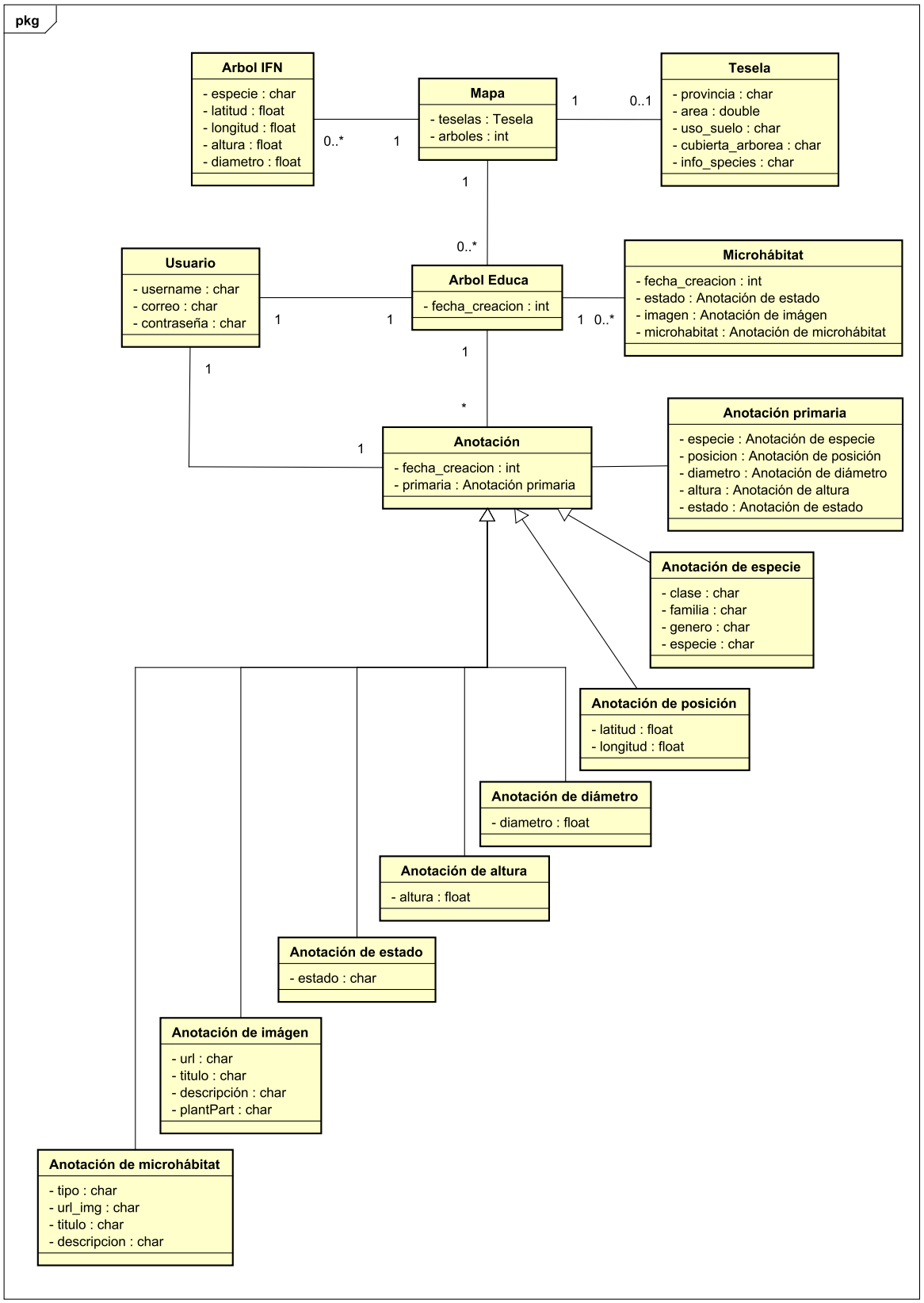


Figura 3.9: Mapa conceptual del dominio de EducaWood

Capítulo 4

Diseño e implementación

4.1. Introducción

Tras la especificación de los requisitos del sistema y finalizadas las fases de planificación y análisis, se debe especificar la arquitectura que va a seguir la aplicación en cuestión, distinguiendo las diferentes partes que lo componen y las comunicaciones necesarias entre ellas. Además, es importante presentar y conocer las diferentes tecnologías que se van a utilizar en la implementación. Cabe resaltar que, aunque en el capítulo anterior centrábamos nuestro foco en el análisis excluyendo al *backend*, en este capítulo se van a detallar ciertos aspecto del mismo necesarios para la correcta integración de todas las partes que componen EducaWood.

El contenido de este capítulo se dedica a las fases finales del SDLC de EducaWood, en concreto las fases de Diseño e Implementación. Se estructura como sigue: en la sección 4.2 se presenta la arquitectura de la aplicación, mientras que en la 4.3 comentan las diferentes tecnologías que se utilizarán para su desarrollo. En la sección 4.4 se presenta la implementación necesaria del *backend* para su correcta integración con el *frontend*, concretamente se detalla la ontología diseñada para estructurar los datos abiertos enlazados utilizados, la configuración de CRAFTS y lo relacionado con los servicios de autenticación y almacenamiento. A continuación, en la sección 4.5 se presenta lo propio del *frontend*. El capítulo termina con unas breves conclusiones sobre el mismo y la consecución de las fases aquí tratadas.

4.2. Arquitectura

En la Figura 4.1 se muestra la arquitectura de la aplicación web EducaWood. La arquitectura de tres niveles propuesta (comúnmente conocida como *three-tier architecture* [Ram03]) se compone de un *frontend* y un *backend*, además de los diferentes navegadores de los dispositivos de cada usuario y los conjuntos de datos abiertos enlazados utilizados. Los navegadores se corresponderían con la capa de presentación. Actúan como clientes del *frontend*, exponiendo las diferentes vistas de la interfaz web a los usuarios finales. El *frontend* se corresponde con la capa lógica, la cual realiza un procesamiento detallado de los datos, además de encargarse de mantener el estado global de la aplicación. Por último, el *backend* se corresponde con la capa de datos, la cual incluye los mecanismos de persistencia de datos y la capa de acceso a los mismos que encapsula estos mecanismos y los expone. La capa de acceso a los datos proporciona una API REST a la capa lógica, en este

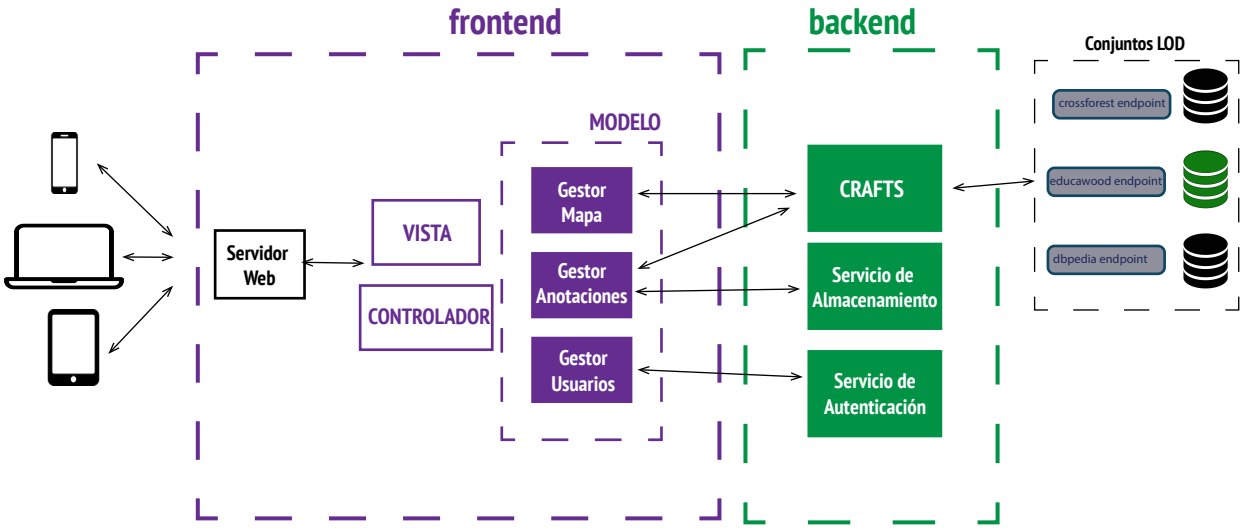


Figura 4.1: Arquitectura de EducaWood

caso, CRAFTS expondrá una API REST propia, al igual que el servicio de autenticación y de almacenamiento. Estos dos servicios no pueden ser abiertos, por lo que no serán LOD teniendo en cuenta la privacidad de los usuarios y que las imágenes no deben ser almacenadas en un almacén de triplas.

Centrándonos en el *frontend*, se ha escogido un modelo-vista-controlador (MVC, del inglés *Model-View-Controller*) [Dea09] para permitir independizar la lógica de la aplicación de la parte visual, mediante el uso de un controlador que administra los procesos sirviendo como puente entre estos. En el **modelo** estará toda la lógica de negocio de la aplicación. Se ha dividido en tres partes, de acuerdo con la división de funcionalidades detallada en los requisitos funcionales: gestor de usuarios, gestor del mapa y gestor de anotaciones (ver 3.3.1). Además, será el encargado de las comunicaciones con los diferentes componentes del *backend*. En la **vista** se definirá la parte visual del sistema, en la que se establecerán todas las interfaces gráficas de usuario (según el prototipo detallado en la Figura 3.7). Con la vista se representa todo el modelo, permitiendo la interacción entre la aplicación y los usuarios finales. Por último, el **controlador** será el encargado de definir la lógica de administración del sistema, estableciendo la conexión entre la vista y el modelo.

4.3. Tecnologías de implementación

Tras diseñar la arquitectura de la aplicación y definir los diferentes componentes necesarios, se han escogido una serie de tecnologías de implementación que serán las encargadas de cumplir con los requisitos de diseño del sistema. Estas tecnologías se muestran en la Figura 4.2

Para el servidor web se ha escogido **NGINX**, un servidor web que actúa también como *proxy* inverso [Ngi]. Se hablará de él en el siguiente capítulo (ver sección 5.2).

Para el desarrollo del *frontend* se ha escogido el *framework* **Angular** [**Ang**]. Se trata de un *framework* de código abierto que hace uso de TypeScript, un lenguaje de código

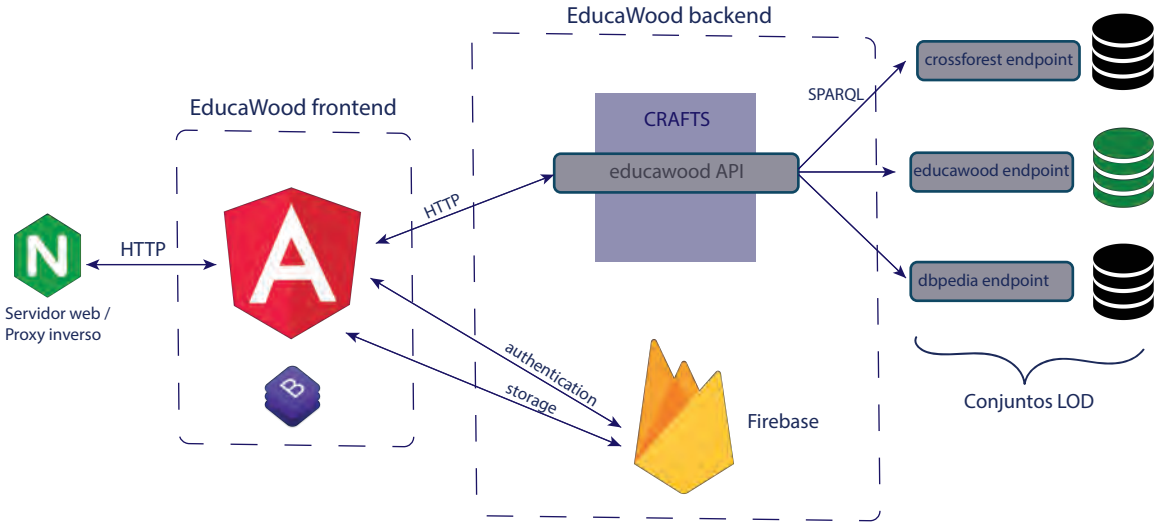


Figura 4.2: Tecnologías de implementación de EducaWood

abierto que se basa en JavaScript[1], con la adición de declaraciones de tipos a los datos. Ha sido elegido principalmente debido a su patrón MVC, ya que permite la implementación de los tres componentes de este modelo gracias a la existencia de **plantillas** (vista), **componentes** (controlador) y **servicios** (modelo). Las plantillas (*templates*) son ficheros HTML que combinan el HTML convencional, el lenguaje de marcado CSS y las directivas propias de Angular. Se encargan de definir las diferentes interfaces de usuario de la aplicación. Los componentes son clases con propiedades y métodos. Son los encargados de relacionar las vistas (plantillas) de cada uno de ellos con la lógica que proporcionan los servicios. Los componentes relacionados entre sí se agrupan por **módulos**. Por último, los servicios proporcionan funcionalidades específicas que no están relacionadas directamente con las vistas, sino con los datos que se manejan en la aplicación. Por ello, Angular encaja a la perfección con la descripción de requisitos de la arquitectura. El otro motivo de su elección es que permite desarrollar aplicaciones web de una sola página (SPA, del inglés *Single Page Applications*) [Mik13]. Una SPA es una aplicación web que interactúa con el usuario modificando dinámicamente la página actual con nuevos datos del servidor, en lugar de cargar una página nueva entera como se hace con el modelo tradicional. El objetivo es lograr transiciones más rápidas que hagan que la aplicación web se parezca más a una aplicación nativa. Desarrollar una SPA tiene ciertas ventajas, ya que, aunque la velocidad de carga puede resultar un poco lenta la primera vez que se abre, navegar a partir de entonces es mucho más dinámico que en un sitio web tradicional. Esto se debe a que se carga toda la página de golpe con la primera llamada. Sin embargo, no todo son ventajas, ya que las SPA son una evolución que se aleja del modelo de diseño de páginas sin estado para el que se diseñaron originalmente los navegadores. Esto se traduce en una mayor complejidad en el diseño de las mismas.

Para el diseño de las vistas, se ha hecho uso de **Bootstrap** [**Boo**], un *framework* CSS utilizado en aplicaciones *frontend* para desarrollar aplicaciones que se adaptan a cualquier dispositivo. Combina CSS y JavaScript para estilizar los elementos de una página HTML. Es una herramienta que proporciona interactividad en la aplicación, por lo que ofrece una

[1] `https://www.typescriptlang.org/`

serie de componentes que facilitan la comunicación con el usuario, como menús de navegación, controles de página, barras de progreso, etc. Además de todas las características que ofrece el *framework*, su principal objetivo es permitir la construcción de sitios web que se adapten para funcionar en ordenadores, tablets y *smartphones*, de una manera simple y organizada.

Tratando ahora el *backend*, tenemos **CRAFTS** [VG21]. Es la herramienta utilizada para acceder a los diferentes conjuntos LOD que consumirá EducaWood, a través de una única API REST uniforme y que no requiere de conocimiento de tecnologías semánticas como RDF y SPARQL (no muy conocidas por los desarrolladores web). Por otro lado, para la implementación tanto del servicio de autenticación como el de almacenamiento se ha utilizado **Firebase** [Firc], una plataforma de Google que proporciona servicios de *backend* en la nube (BaaS, del inglés *Backend as a service*). Entre sus muchas funcionalidades, se ha usado para las tareas de gestión de usuarios (*Firebase Authentication*) y como almacenamiento de las imágenes subidas por los usuarios de EducaWood (*Firebase Cloud Storage*). ***Firebase Authentication***[2] posee varios SDK (*software development kits*) fáciles de usar y bibliotecas de IU ya elaboradas para autenticar a los usuarios en la aplicación. Además, admite autenticación mediante correo/contraseña, número de teléfono o incluso proveedores de identidad federada populares, como Google, Facebook y Twitter. Por otro lado, ***Cloud Storage***[3] para Firebase es un servicio de almacenamiento de objetos potente, simple y rentable construido para el escalado de Google. Sus SDK agregan la seguridad de Google a las operaciones de carga y descarga de archivos de las aplicaciones de Firebase, sin importar la calidad de la red. La principal ventaja es que ambas funcionalidades se pueden integrar con facilidad en un *backend* personalizado.

Por último, están los puntos SPARQL (*endpoints* en la Figura 4.2), que permiten el acceso a los diferentes conjuntos LOD utilizados en la aplicación. Existen dos conjuntos LOD de sólo lectura, correspondientes a los datos del Cross-Forest y los de la dbpedia y un conjunto creado especialmente para la aplicación, tanto de lectura como escritura, que almacena los datos creados por los usuarios *Woodies*. La manera de interactuar con estos datos es mediante consultas SPARQL a cada uno de los *endpoints* especificados, tarea llevada a cabo por CRAFTS.

4.4. Implementación del *backend*

En esta sección se indaga en diferentes aspectos del *backend* necesarios para realizar la integración del mismo con el *frontend*. Estos son: el diseño de la ontología de EducaWood, el diseño e implementación de una API CRAFTS y la implementación de Firebase.

4.4.1. Ontología

A partir del modelo de dominio especificado en el capítulo anterior (sección 3.6), se procede a diseñar las clases y propiedades de la ontología para la anotación social de árboles y microhábitats (STA, *Social Tree Annotation Ontology*). El resto de LOD utilizados en la aplicación ya tienen sus propias ontologías. Sin embargo, la ontología STA se ha tenido que diseñar para cumplir con los requisitos de anotación social. La ontología creada

[2] `https://firebase.google.com/docs/auth`
[3] `https://firebase.google.com/docs/storage`

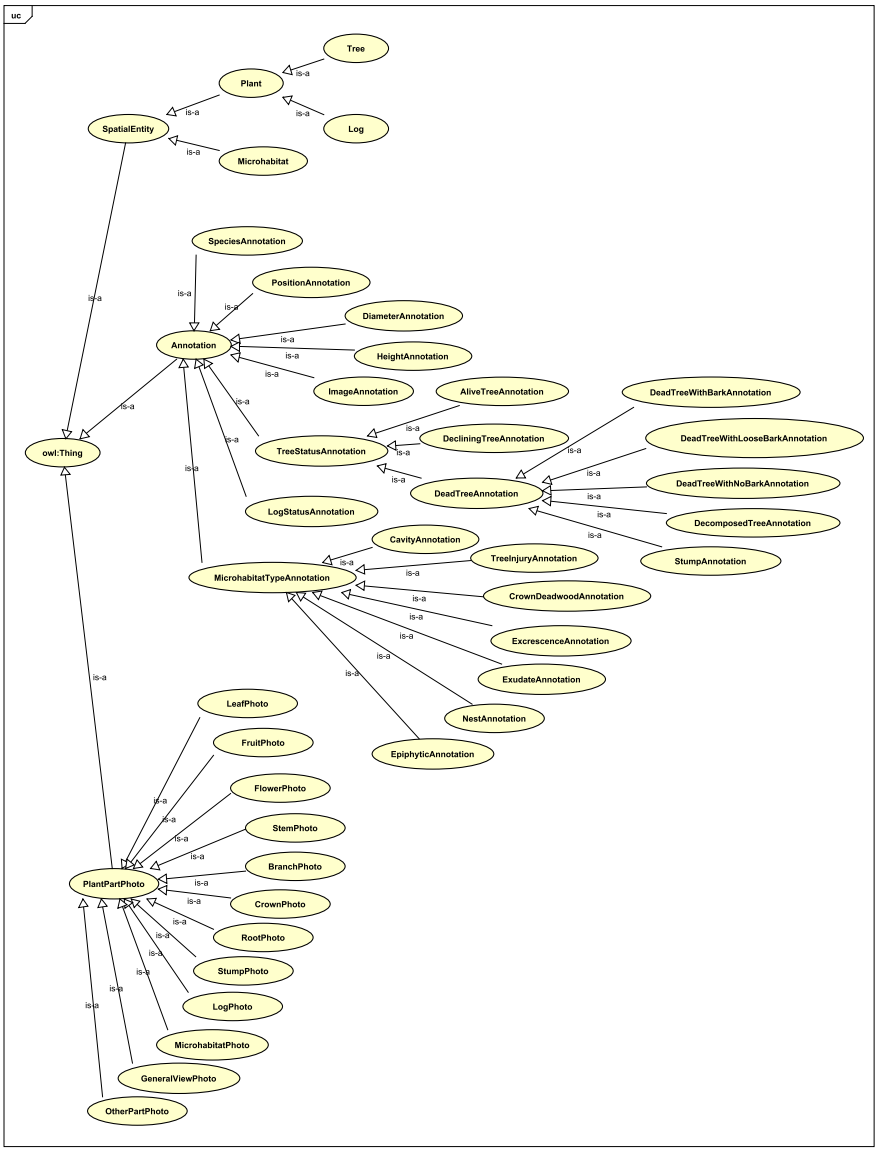


Figura 4.3: Clases de la ontología creada para EducaWood

es una evolución de la ontología diseñada en [MS20] (*Social Tree App ontology*), en la que la mayoría de los términos son propios del dominio de la aplicación. La ontología está propuesta en inglés para facilitar su futura reutilización.

La creación de una ontología consiste en definir los diferentes componentes de la misma a través de un lenguaje de definición de ontología. Se suele llevar a cabo en dos partes [Bre09]: una primera etapa informal, en la que la ontología se realiza utilizando descripciones en lenguaje natural y, seguidamente, una etapa formal, en la que se codifica en un lenguaje formal específico para ontologías, como puede ser OWL. En esta última etapa se pueden utilizar herramientas para su visualización, como WebVOWL [Loh15]. En la etapa informal participamos todo el equipo de manera activa en la definición de las diferentes clases que se necesitarían de acuerdo al modelo de datos, especialmente los expertos forestales. La etapa formal fue desarrollada por Guillermo Vega,

tutor de este TFM. La ontología rediseñada ha pasado a llamarse *Social Tree Annotation ontology* (por ello las siglas STA). La URI también se ha renombrado, siendo ahora `http://educawood.gsic.uva.es/sta/ontology/`.

En la Figura 4.3 se representan las diferentes clases de la ontología modelada, así como las relaciones subclase y superclase de cada una de ellas. Podemos diferenciar tres clases base: `SpatialEntity` (Entidad espacial), `Annotation` (Anotación de un árbol) y `PlantPartPhoto` (Parte de la planta de la foto). De `SpatialEntity` cuelgan `Plant` y `Microhabitat`. A su vez, `Tree` y `Log` son especializaciones de `Plant`. Estas entidades (`Tree`, `Log` y `Microhabitat`) serán creadas por los usuarios. A su vez, la entidad `Microhabitat` podrá ligarse a un `Tree` o a un `Log` con la propiedad `isInPlant`. Sobre cada una de estas entidades pueden realizarse anotaciones (clase `Annotation`). De `Annotation` cuelgan todos los tipos de anotaciones posibles, entre ellas `SpeciesAnnotation`, `PositionAnnotation`, `ImageAnnotation`, `LogStatusAnnotation`, `MicrohabitatTypeAnnotation` (estas dos últimas con sus correspondientes especializaciones), etc. Por simplicidad, se han omitido en el diagrama las diferentes subclases de cada especialización de `MicrohabitatTypeAnnotation`, pero se pueden consultar en la ontología completa del Anexo B. Además, se han omitido las especializaciones de tronco caído (`LogStatusAnnotation`) puesto que, aunque están incluidas en la ontología, no forman parte del modelo de datos diseñado para la aplicación, aunque podrán ser implementadas en un futuro sin mucho esfuerzo adicional. Por último, hay 12 tipos de `PlantPartPhoto`, con los que puede indicarse de qué va una foto, por ejemplo la vista general de un árbol, una hoja, la corona, un tronco caído, un microhábitat, etc.

Una entidad espacial puede tener cero, una o varias anotaciones de cada tipo. Para ligar las entidades espaciales con los diferentes tipos de anotaciones se han definido múltiples propiedades que especializan a la propiedad `hasAnnotation` (tiene anotación). Por ejemplo, `hasImageAnnotation` para una anotación de tipo imagen. Dependiendo del tipo de anotación, la especialización de `hasAnnotation` será diferente. Además, se han definido también propiedades para determinar las anotaciones de tipo primario (en todos los casos salvo en las de tipo imagen), que indican cuál debería ser la anotación preferida en el caso de haber varias. Como ya se comentó en el capítulo anterior, en el prototipo a desarrollar la anotación primaria será la última anotación realizada de cada tipo.

Una vez terminada la fase formal, identificando los diferentes elementos de la ontología (clases, propiedades y relaciones entre ellos), es necesario pasar a la codificación de la misma. Como resultado se obtiene la ontología disponible en el Anexo B, en la que se ha utilizado fundamentalmente el lenguaje de ontologías RDFS y el vocabulario de OWL.

4.4.2. CRAFTS

CRAFTS es uno de los pilares fundamentales de EducaWood, puesto que es la herramienta encargada de la transformación de los datos abiertos enlazados utilizados en la aplicación a JSON y viceversa. Esta sección especifica en primer lugar los pasos realizados para obtener una API CRAFTS y, posteriormente, una pequeña guía de cómo se ha utilizado.

Configuración

Para poner a punto su utilización han sido necesarias tres acciones: (i) darse de alta, (ii) crear un repositorio de triplas donde almacenar los datos creados por los usuarios de EducaWood y (iii) configurar una API CRAFTS para la aplicación. Para darse de alta en CRAFTS basta con rellenar el formulario en su página principal[4] o bien realizar una llamada HTTP POST a la url `https://crafts.gsic.uva.es/users` con el JSON siguiente en el cuerpo de la llamada:

```
{
  "login": "jandrade",
  "password": "********",
  "firstName": "Jimena",
  "lastName": "Andrade",
  "email": "jimena.andrade@alumnos.uva.es"
}
```

Tras mandar la petición, llegará al correo electrónico un mensaje para activar la cuenta: "*Welcome to this CRAFTS service. In order to activate your account, please click on this link*". Al acceder al enlace, si todo va bien, CRAFTS devuelve un código de respuesta 200, indicando que el usuario ha sido activado.

Para la creación del almacén de triplas RDF de EducaWood se ha utilizado OpenLink Virtuoso [Vir], una plataforma híbrida de almacén de triplas y middleware de alto rendimiento y escalable para el desarrollo de LD que soporta SPARQL 1.1 integrado en SQL para consultar datos RDF almacenados en la base de datos de Virtuoso. Se ha configurado un *endpoint* cuya URI de acceso es `https://crossforest.gsic.uva.es/pruebas/sparql`. En este almacén de triplas se ha cargado la ontología diseñada en 4.4.1.

Por último, se ha diseñado el fichero de configuración para crear la API CRAFTS. Este fichero consta de cuatro partes fundamentales: el campo `apiId` para especificar el nombre de la API CRAFTS, el campo `endpoints` en el que se especifican los diferentes puntos SPARQL de donde obtener los datos abiertos enlazados, el campo `model` para listar los diferentes datos que se quieren extraer de los *endpoints* especificados y, por último, el campo `queryTemplates`, en el que se especifican consultas SPARQL parametrizadas. A modo de ejemplo, se va a ilustrar parte del fichero de configuración. No obstante, se puede consultar el JSON completo en el Anexo C.

- Nombre de la API CRAFTS: `educawood`

```
{
    "apiId": "educawood",
}
```

- Configuración de los *endpoints*: se muestra la configuración del *endpoint* de CrossForest. En ella se especifica la URI del punto SPARQL, el nombre del grafo y el método para realizar peticiones SPARQL, en este caso solo de lectura. Para el caso del *endpoint* de educawood se permiten escrituras proporcionando credenciales para SPARQL Update.

[4] `https://crafts.gsic.uva.es/`

```
{
    "endpoints": [
        {
            "id": "crossforest",
            "sparqlURI": "https://crossforest.gsic.uva.es/sparql/",
            "graphURI": "http://crossforest.eu",
            "httpMethod": "GET"
        },
    ]
}
```

- Configuración de recursos del modelo. Se ilustra la especificación del recurso cuyo identificador es `EducaTree`, el cual representa un árbol creado por un usuario *Woodie*. Dentro del campo `oprops` se describen las propiedades de tipo objeto (definidas en OWL), como por ejemplo el creador del árbol (`creator`), su posición primaria (`position`) y sus anotaciones de posiciones (`positionAnnotations`). En todas ellas se debe especificar la IRI y el *endpoint* del cuál se obtendrá la información del recurso. En el campo `dprops` se especifican las propiedades de tipo dato (`datatype` property en OWL), en este caso particular la fecha de creación del árbol (`created`). De la misma manera se especifican el resto de recursos del modelo detallado en la sección 3.6.

```
{
    "model": [
    {
        "id": "EducaTree",
        "oprops": [
            {
            "label": "creator",
            "iri": "http://purl.org/dc/terms/creator",
            "endpoint": "educawood"
            },
            {
            "label": "position",
            "targetId": "PositionAnnotation",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/ ...
                hasPrimaryPosition",
            "embed": true,
            "endpoint": "educawood"
            },
            {
            "label": "positionAnnotations",
            "targetId": "PositionAnnotation",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/ ...
                hasPositionAnnotation",
            "embed": false,
            "endpoint": "educawood"
            },
        ],
        "dprops": [
            {
            "label": "created",
            "iri": "http://purl.org/dc/terms/created",
            "endpoint": "educawood"
            }
        ],
```

```
33          ]
34  }
```

- Consultas SPARQL: se encuentran dentro del campo `queryTemplates`. Para la configuración de esta sección es necesario tener conocimientos de RDF y SPARQL, puesto que deben especificarse las consultas en lenguaje SPARQL dentro del campo `template`. De esta tarea se ha encargado Guillermo Vega, tutor de este TFM, puesto que es el experto en consultas de este tipo. En la configuración se han especificado varias consultas que se pueden realizar:
  - `indivs`: para obtener miembros de un recurso determinado, especificando opcionalmente un offset.
  - `allSpecies`: permite obtener todas las especies con su nombre científico, sus nombres vulgares en diferentes idiomas y su padre.
  - `infoClasses`: permite obtener información relacionada con las clases de los árboles.
  - `lastEducatrees`: para obtener los últimos árboles creados por los usuarios de EducaWood. La consulta se realiza al *endpoint* de educawood.
  - `patchesinbox`: para obtener las teselas existentes en una determinada zona delimitada por un rectángulo. Dicho rectángulo se especifica introduciendo las coordenadas de latitud y longitud.
  - `treesinbox`: para obtener los árboles del IFN existentes en un rectángulo delimitado por las coordenadas de latitud y longitud especificadas. La consulta se realiza al *endpoint* del cross-forest.
  - `educatreesinbox`: permite obtener los árboles creados por los usuarios de EducaWood existentes en un rectángulo delimitado por las coordenadas de latitud y longitud especificadas.

Finalmente, para publicar la API será necesario realizar una llamada HTTP PUT a la URL `https://crafts.gsic.uva.es/apis/{apiId}` con el JSON en el cuerpo de la petición. Con esta configuración y la creación de la API ya será posible realizar la implementación para satisfacer todos los casos de uso especificados en la sección 3.4.

Utilización

CRAFTS está documentado[5] con la especificación OpenAPI, permitiendo hacer llamadas a la API de manera sencilla utilizando simplemente un navegador. Todas las operaciones disponibles se pueden repasar en la Tabla 2.3. CRAFTS se utiliza en EducaWood para obtener LOD de teselas (solo lectura), árboles del IFN (solo lectura) y árboles creados por los usuarios (lectura y escritura). Para obtener la información de una entidad conocida es tan sencillo como realizar una consulta a la url `https://crafts.gsic.uva.es/apis/educawood/resource?id=EducaTree&iri={iri}`. Sin embargo, muchas veces no se conocen a priori los identificadores (iris) de las entidades, por lo que hacen falta descubrirlas. Para eso se utilizan las consultas parametrizadas. Para ilustrar la manera de trabajar, se va a explicar un ejemplo con los datos detallados en la configuración anterior. Es inportante resaltar que para realizar las consultas es necesario estar registrado en la herramienta.

[5] `https://crafts.gsic.uva.es/docs/`

Se quiere obtener todos los árboles creados en la aplicación que se encuentren en una región de España delimitada por dos puntos definidos por su latitud y longitud. Para ello, utilizaremos la consulta parametrizada cuyo identificador es `educatreesinbox`. La consulta utilizada se corresponde con la C9 de la Tabla 2.3, en la que `apiId` se corresponde con `educawood` y la *query* se especifica tras el signo de interrogación (`?`). El resultado es: `https://crafts.gsic.uva.es/apis/educawood/query?id=educatreesinbox&lngwest=-3.6&lngeast=-3.5&latnorth=40.81&latsouth=40.8` La respuesta a esta petición se muestra en la Figura 4.4, donde también se incluye un campo `query` en el que se especifica la consulta realizada en lenguaje SPARQL. Se observa que existen dos árboles creados en la aplicación en esa región. Si se quiere consultar el contenido del primer recurso, cuya IRI es: `http://educawood.gsic.uva.es/tree/2a0d17f0-408a-43e8-829d-fa66b807edef`, se realizará una consulta de tipo C4, donde `apiId` es `educawood`, `id` se corresponde con `EducaTree` e `iri` será la especificada anteriormente, de manera que la URL de consulta será: `https://crafts.gsic.uva.es/apis/educawood/resource?id=EducaTree&iri=http://educawood.gsic.uva.es/tree/2a0d17f0-408a-43e8-829d-fa66b807edef`. Esta petición también se podría realizar desde la interfaz de la documentación, rellenando los campos del formulario como se muestra en la Figura 4.5.


```
▼ results:
    distinct:          false
    ordered:           true
  ▼ bindings:
    ▼ 0:
      ▼ tree:
          type:        "uri"
        ▼ value:       "http://educawood.gsic.uva.es/tree/2a0d17f0-408a-43e8-829d-fa66b807edef"
      ▼ lat:
          type:        "typed-literal"
          datatype:    "http://www.w3.org/2001/XMLSchema#decimal"
          value:       "40.800683"
      ▼ lng:
          type:        "typed-literal"
          datatype:    "http://www.w3.org/2001/XMLSchema#decimal"
          value:       "-3.597408"
    ▼ 1:
      ▼ tree:
          type:        "uri"
        ▼ value:       "http://educawood.gsic.uva.es/tree/c699dacc-8e6d-4c32-ac53-f66fa9a3902c"
      ▼ lat:
          type:        "typed-literal"
          datatype:    "http://www.w3.org/2001/XMLSchema#decimal"
          value:       "40.800699"
      ▼ lng:
          type:        "typed-literal"
          datatype:    "http://www.w3.org/2001/XMLSchema#decimal"
          value:       "-3.597836"
```


Figura 4.4: Árboles creados por usuarios de EducaWood en una región determinada

Para terminar, se va a ilustrar un ejemplo de creación de árboles y anotaciones en CRAFTS. Para crear una entidad de tipo `EducaTree` se utiliza el método PUT a la URL `https://crafts.gsic.uva.es/apis/educawood/resource?id={id}&iri={iri}` (correspondiente a la consulta C5 de la Tabla 2.3), más la representación en formato JSON del árbol en el cuerpo de la petición. El parámetro `id` se corresponde con `EducaTree`. Al tratarse del método PUT, el parámetro `iri` debe ser elegido por el usuario a la hora de realizar

la consulta. El diseño de las URIs se tratará en la sección siguiente (4.5.2), por ahora se hablará de manera genérica de las mismas. La petición incluye autenticación tipo *Bearer*. *Bearer authentication* [Jon12] (también llamada autenticación de *token*) es un esquema de autenticación HTTP que implica *tokens* de seguridad llamados *bearer tokens*. Estos *tokens* sirven para permitir el acceso a un determinado recurso al portador del mismo. El *token* portador es una cadena, normalmente generada por el servidor en respuesta a una solicitud de acceso. El cliente debe enviar este *token* en la cabecera de autorización cuando realice peticiones a recursos protegidos. Para ello, en cada petición PUT debe añadirse la cabecera: `Authorization: Bearer <token>`. La representación del recurso se muestra a continuación:

```
1  {
2      "iri": "http://educawood.gsic.uva.es/posann/id0",
3      "creator": "http://educawood.gsic.uva.es/user/jandrade",
4      "created": "2021-05-13T11:42:10.398Z",
5      "position": {
6          "iri": "http://educawood.gsic.uva.es/posann/id0",
7          "creator": "http://educawood.gsic.uva.es/user/jandrade",
8          "created": "2021-05-13T11:42:10.398Z",
9          "latWGS84": 40,
10         "lngWGS84": -3,
11         "types": ...
               "http://educawood.gsic.uva.es/sta/ontology/PositionAnnotation"
12     },
13     "positionAnnotations": "http://educawood.gsic.uva.es/posann/id0",
14     "types": "http://educawood.gsic.uva.es/sta/ontology/Tree"
15 }
```

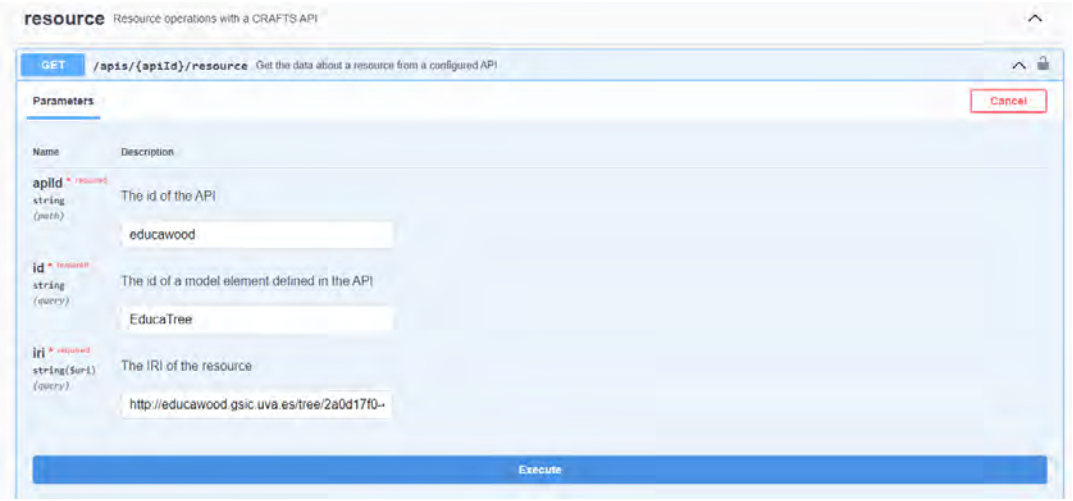


Figura 4.5: Obtención de la información de un recurso con la documentación de CRAFTS

De esta manera, se creará un recurso de tipo `EducaTree` cuya `iri` será la especificada en la línea 2. El creador también se especifica con una IRI de cuyo formato se hablará posteriormente. Adicionalmente, se especifica la fecha de creación (campo `created`), y una anotación de posición que incluye una IRI, el creador, la fecha de creación y los parámetros de latitud y longitud del árbol. Además, debe especificarse el tipo de datos con el que se corresponden tanto la anotación como el árbol (campos de tipo `types`). Si a este árbol creado se le quiere añadir una anotación nueva de posición, se utilizará el

método PATCH a la misma URL anterior (consulta C6). Esta anotación pasará a ser la anotación primaria de posición. La representación del recurso será:

```
[
 {
   "op": "replace",
   "path": "/position",
   "value": {
        "iri": "http://educawood.gsic.uva.es/posann/id1",
        "creator": "http://educawood.gsic.uva.es/user/jandrade",
        "created": "2021-05-13T11:42:10.398Z",
        "latWGS84": 40.5,
        "lngWGS84": -3.1,
        "types": ...
            "http://educawood.gsic.uva.es/sta/ontology/PositionAnnotation"
    }
 },
 {
   "op": "add",
   "path": "/positionAnnotations/-",
   "value": "http://educawood.gsic.uva.es/posann/id1"
 }
]
```

Con este JSON, en primer lugar se reemplaza la anotación primaria existente con una nueva anotación cuya IRI es `iri_pos2`. A continuación, se incorpora esta nueva anotación (operación `add`) al listado de anotaciones de posición existente. La manera de actuar para crear anotaciones de otros tipos es similar a la explicada en este ejemplo, modificando los parámetros específicos de cada tipo de anotación y el campo `types` según corresponda.

4.4.3. Firebase

La integración de Firebase con el *frontend* de la aplicación resulta relativamente sencilla gracias a los SDK desarrollados por Google. Cabe recordar que el *frontend* está desarrollado con Angular, el cual es también un producto de Google. Los pasos realizados para implementar Firebase en EducaWood fueron:

1. Crear un nuevo proyecto con mi cuenta personal de Google en `https://firebase.google.com/`.
2. Selección de un nuevo proyecto, cuyo nombre es **educawood**. El proyecto es de tipo *App Web*.
3. Agregar los SDK de Firebase (dependencias necesarias para utilizarlo) y, luego, inicializar el proyecto.
4. Generar un fichero de variables de configuración en el *frontend* (proyecto de Angular, fichero `environment.ts` disponible en el Anexo D).
5. Inyectar la librería Firebase en el *frontend* de Angular (en el fichero `app.module.ts`).
6. Importar el módulo `AngularFireAuth` en cada componente Angular que lo utilice.

La gestión de usuarios se realiza a través de la interfaz web del proyecto creado, como se muestra en la Figura 4.6. Desde la interfaz de autenticación se lleva un registro de

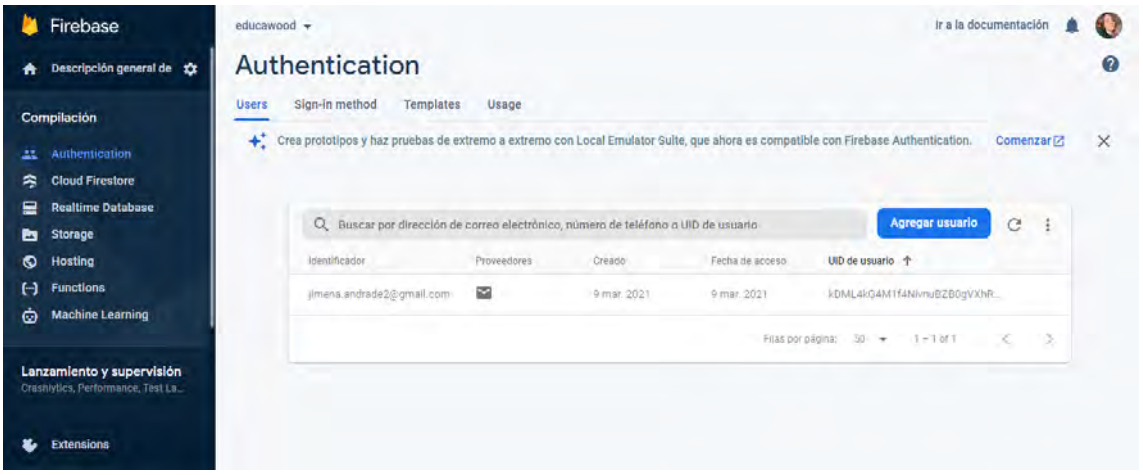


Figura 4.6: Interfaz de gestión de usuarios en Firebase

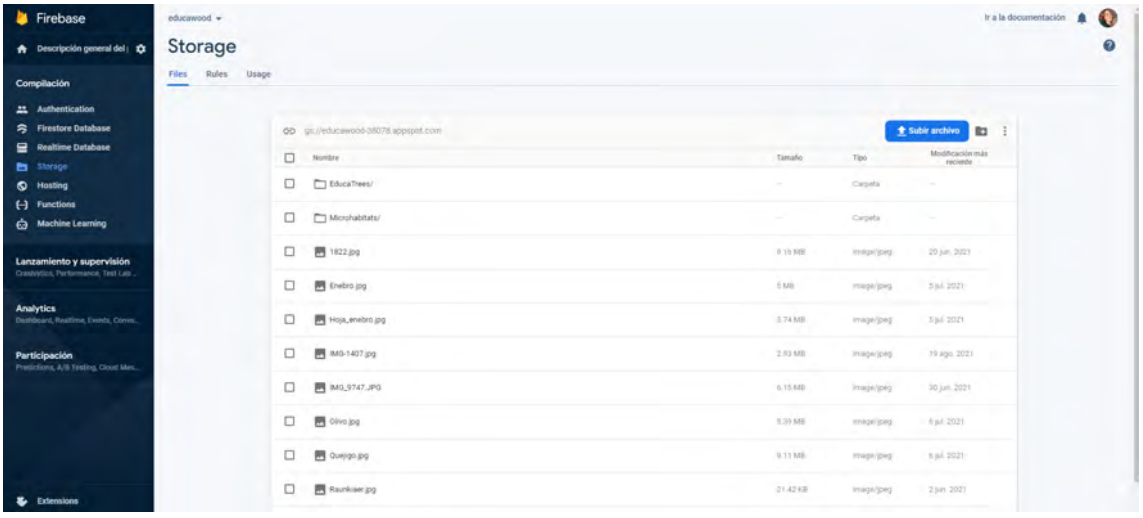


Figura 4.7: Interfaz de gestión del almacenamiento en Firebase

todos los usuarios activos en EducaWood. Además, es posible activar una funcionalidad que hace que Firebase envíe un vínculo de autenticación al correo electrónico del usuario o recuperar la contraseña mediante el mismo. Estas acciones están implementadas en el *frontend* de la aplicación, para las cuales es necesario personalizar las plantillas (dentro del apartado *templates*) que se mandarán al correo de los usuarios. Para la implementación de esta parte se han seguido los pasos de [Fira], en la que es necesario importar la librería `AngularFireAuth` en el proyecto de Angular. Por otro lado, la gestión del almacenamiento de las imágenes (*Storage*) se realiza a través de la interfaz de la Figura 4.7, en la que se pueden observar las carpetas destinadas al almacenamiento de las imágenes relacionadas con los árboles de tipo `educatrees` y las imágenes de microhábitats. Seleccionando cualquiera de las imágenes es posible consultar información básica de las mismas, como su ubicación o sus metadatos, además de poder eliminarlas fácilmente. Las imágenes se suben a través de la interfaz del *frontend* de la aplicación, cuya implementación se ha realizado siguiendo [Firb]. La librería necesaria para realizar la implementación es la `AngularFireStorage`.

Como ejemplo de integración de Firebase en Angular, se muestra un ejemplo de un

método implementado para crear un nuevo usuario en la aplicación. Como se puede comprobar, con las librerías importadas en el proyecto es muy sencillo utilizar los métodos de Firebase para realizar las diferentes acciones necesarias. En este caso concreto, el método `createUserWithEmailAndPassword(email, pwd)` nos permite crear un nuevo usuario en el sistema introduciendo un email y una contraseña.

```
public crearNuevoUsuario() {
      this.afAuth.createUserWithEmailAndPassword(email, password)
        .then((user) => {
          if (this.nombreUsuario != null) {
            this.UserServ.actualizarNombreUsuario(this.nombreUsuario);
          }
          alert("Usuario creado correctamente.");
        })
        .catch((error) => {
          if (error.code == "auth/email-already-in-use") {
            alert("El correo electronico ya est  en uso.");
          }
          console.log(error);
        });
  }
```

4.5. Diseño e implementación del *frontend*

Una vez ilustrado cómo usar los servicios del *backend* necesarios para proporcionar la funcionalidad de la aplicación, pasamos al diseño e implementación del *frontend*. El desarrollo del mismo se ha realizado con Angular, el cual constituye el componente principal del sistema. El *frontend* es el encargado tanto de la presentación de la interfaz como de las tareas relacionadas con modelar de manera eficiente las interacciones entre los usuarios, los datos y los servicios de *backend*.

4.5.1. Aspectos de diseño

En primer lugar, se presenta en la Tabla 4.1 el diseño de las URIs de todos los recursos expuestos por el *frontend* de la aplicación. Estos recursos responden a las diferentes vistas de la interfaz de usuario diseñada en la sección 3.5

Recurso	**URL**
Inicio	`https://educawood.gsic.uva.es/`
Inicio de Sesión	`https://educawood.gsic.uva.es/login`
Ajustes	`https://educawood.gsic.uva.es/ajustes`
Mapa	`https://educawood.gsic.uva.es/mapa`
Nuevo Árbol	`https://educawood.gsic.uva.es/nuevoArbol`
Detalles de un árbol	`https://educawood.gsic.uva.es/educatree/{idArbol}`
Actividades	`https://educawood.gsic.uva.es/actividades`

Tabla 4.1: Recursos ofrecidos por el *frontend* de EducaWood

Estructura del código

El código completo del proyecto se puede consultar en [AH21a]. En [AH20] ya se empleó Angular para desarrollar la aplicación Timber. Sin embargo, dicha aplicación resultó ser poco escalable debido a la falta de organización del código desarrollado. Por ello, se decidió empezar de cero con EducaWood, proponiendo un cambio arquitectónico que permitiera mayor escalabilidad y mejor organización del código de la aplicación. En primer lugar, se siguieron un conjunto de buenas prácticas cuyos conocimientos se adquirieron con [Ope20]. Estas son:

1. Unificar los nombres de los ficheros según la funcionalidad que desempeñan. En la Tabla 4.2 se especifica el formato a seguir.

Tipo de fichero	**Nombre**
Componente	`{nombre}.component.ts`
Módulos	`{nombre}.module.ts`
Servicios	`{nombre}.service.ts`
Modelos de datos	`{nombre}.model.ts`

Tabla 4.2: Formato de nombres para los ficheros del proyecto

2. Seguir el principio de responsabilidad única: un objeto o una clase solamente debe realizar una tarea. Se debe intentar simplificar al máximo la responsabilidad a los componentes.

3. Pensar en componentes. Se trata de dividir la vista de la aplicación web en pequeñas partes de funcionalidad, de manera que se tenga una organización lógica en las carpetas del proyecto. La idea es ver la aplicación como la suma de múltiples funcionalidades y no como una vista completa.

4. Organizar el código propio en carpetas. Dentro del proyecto, el código implementado propiamente dicho se encuentra dentro del directorio `educawood-app/src/app`. Para una correcta organización, se propone una arquitectura formada por cuatro directorios:

   - Directorio `components`: en ella se diferencian todas las funcionalidades básicas de la aplicación que no tienen nada que ver unas con otras. Estas son: `actividades` (para la implementación de todos los casos de uso relacionados con la gestión de anotaciones y gestión de actividades), `ajustes`, `explorer` (para implementar todos los casos de uso relacionados con la gestión del mapa), `header`, `inicio` y `login` (para la implementación de todos los casos de uso relacionados con la gestión de usuarios).
   - Directorio `core`: nos permite abstraer las funcionalidades de la aplicación. Está subdividida en varios directorios, algunos de los cuáles están inutilizados. Sin embargo, la existencia de los mismos nos permitirá continuar con la implementación de futuras versiones.

- Directorio `authorization`: almacena clases con funcionalidades relacionadas con autorizaciones, por ejemplo, para la autenticación básica HTTP necesaria en CRAFTS.
- Directorio `cache`: para la implementación de la caché de la aplicación. Esta funcionalidad no está desarrollada en esta versión de EducaWood.
- Directorio `localization`: contiene todo lo referente a los archivos de idiomas (de traducciones). Este directorio tampoco está utilizado.

- Directorio `services`: se ubican los diferentes ficheros con la lógica para manejar datos de la aplicación (servicios) y los ficheros con estructuras de datos predefinidas (modelos o interfaces).
- Directorio `ui-controls`: para controles y funciones que se pueden reutilizar.
  - Directorio `dialogs`: errores, mensajes de la web para el usuario, fallo de conexión con el servidor, etc.
  - Directorio `material`: módulos con datos importantes, como por ejemplo la información de Firebase.
  - Directorio `util`: métodos básicos no relacionados directamente con los datos propios de la aplicación, como por ejemplo generar los UUIDs o generar fechas.

Por otro lado, se empleó la extensión `TSLint`[6] para ir comprobando la mantenibilidad del código implementado. TSLint es una herramienta de análisis estático que comprueba el código TypeScript en busca de errores de legibilidad, mantenimiento y funcionalidad. Es ampliamente compatible con los editores y sistemas de compilación modernos y se puede personalizar con sus propias reglas. El fichero `tslint.json` se encuentra en el directorio raíz del proyecto. En este fichero están definidas todas las normas de estilo que debe cumplir el proyecto. Con el comando `ng lint` se comprueban si se están cumpliendo dichas normas y, además, es posible solucionar los errores mediante el comando `ng lint -fix`.

Componentes

Siguiendo las indicaciones de las buenas prácticas de la sección anterior, en concreto la número tres, los componentes de la aplicación siguen una estructura uniforme formada por un directorio `components`, en la que se guardan los diferentes componentes y, un módulo encargado de unificar todos ellos. Dentro del directorio de componentes, existe uno de ellos encargado de la vista de la interfaz. Dicho componente se nombra de manera específica como `{nombreComponente}-layout.component.ts`. En él se especifican los formatos de estilo y se posicionan el resto de componentes, cada uno de los cuales se encarga de una funcionalidad concreta y muy específica. Los ficheros de *layout* de cada uno de los módulos dan como resultado las diferentes vistas ofrecidas por la interfaz de EducaWood. En la Figura 4.8 se muestra como ejemplo el módulo de *login* (`login.module.ts`), en cual está formado por cuatro componentes, uno de ellos el `login-layout.component.ts` cuya funcionalidad ya se ha mencionado anteriormente, otro encargado del formulario de registro (`registro.component.ts`), otro encargado del formulario de inicio de sesión (`login-form.component.ts`) y, finalmente, uno encargado de dar la bienvenida a los usuarios cuando se registran (`bienvenida.components.ts`).

[6] urlhttps://palantir.github.io/tslint/

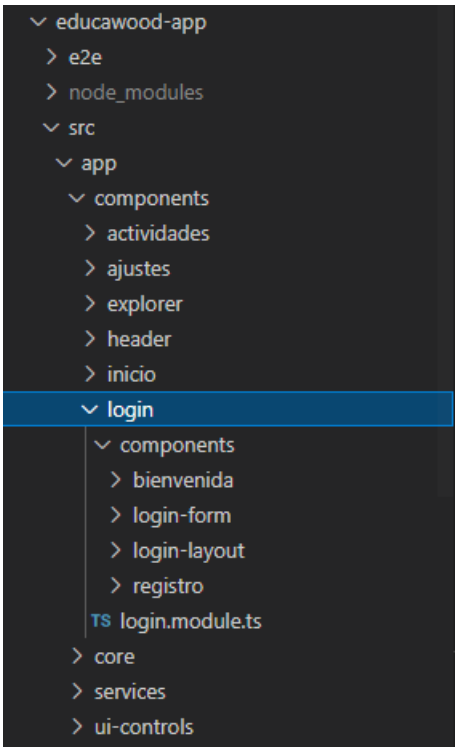


Figura 4.8: Organización de los componentes en directorios

Servicios y modelos de datos

Los servicios son los encargados de proporcionar la lógica necesaria para manejar los diferentes datos de la aplicación. Para ello, se apoyan en modelos de datos (o interfaces) que han sido diseñadas siguiendo el modelo de dominio del sistema. En concreto, se han creado modelos de datos para la mayoría de las clases de la Figura 4.3. A continuación se muestra el fichero `position.model.ts`, el cual especifica el formato de los datos que componen una anotación de posición. De la misma manera se ha procedido con el resto de clases de la ontología.

```
1  // Modelo con los parametros de una anotacion de posicion
2  export interface PositionAnnotation {
3      iri: string;
4      created: string;
5      creator: string;
6      latWGS84: number;
7      lngWGS84: number;
8      isPrimary?: boolean;
9      types?: string;
10 }
```

Las diferentes funcionalidades de la aplicación se han agrupado para crear un total de nueve servicios, que se detallan a continuación:

- Servicio CRAFTS (`crafts.service.ts`): encargado de la comunicación con CRAFTS mediante peticiones HTTP. Todos los componentes que manejan datos abiertos enlazados hacen uso de este servicio.

- Servicio de árboles (`arboles.service.ts`): encargado de las tareas relacionadas con los datos de los árboles, tanto del IFN como los de tipo `EducaTree`. Se apoya en los modelos de datos `Arbol` y `ArbolEduca`, además de hacer uso del servicio de anotaciones y el de usuarios.

- Servicio de anotaciones (`anotaciones.service.ts`): encargado de las tareas relacionadas con los datos de las anotaciones. Hace uso del servicio de CRAFTS, además de utilizar todos los modelos de datos referentes a las anotaciones.

- Servicio de especies (`especies.service.ts`): es una especialización del servicio de anotaciones, para tratar específicamente todo lo relacionado con las especies.

- Servicio de firebase (`firebase-storage.service.ts`): almacena los métodos que gestionan la comunicación y el almacenamiento en la nube de Firebase.

- Servicio del mapa (`mapa.service.ts`): encargado de las funcionalidades relacionadas con el mapa. Hace uso del modelo `tesela` y del servicio de teselas, para la representación de las mismas en el mapa.

- Servicio de microhábitats (`microhabitat.service.ts`): es una especialización del servicio de anotaciones, para tratar específicamente todo lo relacionado con las anotaciones de microhábitats.

- Servicio de teselas (`teselas.service.ts`): contiene métodos especiales para gestionar la información de las teselas. Hace uso del modelo `tesela`.

- Servicio de usuarios (`user.service.ts`): encargado de los métodos para gestionar a los usuarios. Hace uso del modelo `User`.

4.5.2. Aspectos de implementación

El primer paso en la implementación de una aplicación web con Angular es instalar NodeJS[7] y su herramienta Angular CLI[8], un intérprete de línea de comandos que facilita la creación y el desarrollo de proyectos en Angular. A continuación, se crea un proyecto Angular con el comando `ng new educawood-app`, con el que se nos creará automáticamente la estructura de ficheros necesaria para que funcione adecuadamente la aplicación.

Una vez estructurados los ficheros y renombrados según las buenas prácticas comentadas anteriormente, se puede proceder a la implementación de los diferentes casos de uso propuestos. Para los casos de uso relacionados con la gestión de usuarios, han sido necesarias las librerías de Firebase comentadas en la sección 4.4.3. Todos los métodos necesarios están documentados en la página web oficial. Por otro lado, para la implementación de los casos de uso relacionados con la gestión del mapa ha sido de gran utilidad la librería Leaflet[9], una librería JavaScript de código abierto utilizada para la publicación e interacción de mapas en la Web fácilmente adaptables para dispositivos móviles. Leaflet proporciona una API fácil de usar con la que ya se había trabajado previamente. Por último, para la gestión de anotaciones ha sido especialmente relevante la definición y el uso de los diferentes modelos de datos descritos en la sección anterior, puesto que estos casos de uso

[7] `https://nodejs.org/es/`
[8] `https://angular.io/cli`
[9] `https://leafletjs.com/`

manejan una gran cantidad de información. Para la implementación de todas las vistas de EducaWood ha sido de gran ayuda el uso de Bootstrap, especialmente para la implementación de formularios y menús de navegación.

La implementación completa de la aplicación se pueden consultar en el repositorio de GitHub [AH21a].

Integración con CRAFTS

En la sección 4.4.2 se comentó la necesidad de diseñar las URIs para las diferentes entidades que se pueden crear en CRAFTS. Como se recomienda en [BL06], se han utilizado URIs para identificar todos los objetos y conceptos de la aplicación. Un identificador uniforme de recursos (URI) es una secuencia compacta de caracteres que identifica un recurso dándole un nombre único. Partiendo del espacio de nombres de la ontología (*namespace*) de la ontología a `http://educawood.gsic.uva.es/sta/ontology/`, se ha seguido la misma estructura para identificar las diferentes clases de la ontología:

- Formato de URI para identificar los árboles creados por los usuarios de la aplicación (árboles de tipo `educatree`): `http://educawood.gsic.uva.es/tree/{idTree}`, en el que `idTree` es el identificador del árbol.
- Formato de URI para identificar las anotaciones de posición: `http://educawood.gsic.uva.es/posann/{idPosAnn}`, en el que `idPosAnn` es el identificador de la anotación.
- Formato de URI para identificar las anotaciones de especie: `http://educawood.gsic.uva.es/spann/{idEsAnn}`, en el que `idEsAnn` es el identificador de la anotación.
- Formato de URI para identificar las anotaciones de imagen: `http://educawood.gsic.uva.es/imgann/{idImg}`, en el que `idImg` es el identificador de la anotación.
- Formato de URI para identificar las anotaciones de diámetro: `http://educawood.gsic.uva.es/diamann/{idDiamAnn}`, en el que `idDiamAnn` es el identificador de la anotación.
- Formato de URI para identificar las anotaciones de altura: `http://educawood.gsic.uva.es/heightann/{idHeiAnn}`, en el que `idHeiAnn` es el identificador de la anotación.
- Formato de URI para identificar las anotaciones del estado de un árbol: `http://educawood.gsic.uva.es/treestann/{idStAnn}`, en el que `idStAnn` es el identificador de la anotación.
- Formato de URI para identificar las anotaciones de los microhábitats de un árbol: `http://educawood.gsic.uva.es/microhabtann/{idMicAnn}`, en el que `idMicAnn` es el identificador de la anotación.

Todos los identificadores de los diferentes recursos están formados por una cadena de caracteres alfanuméricos aleatoria generada con UUIDs V4 (*universally unique identifier*)[10]. Finalmente, se ha seguido el mismo formato para identificar a los creadores de

[10] `https://www.npmjs.com/package/uuid`

las anotaciones en la aplicación. Estos creadores son los usuarios registrados a través de Firebase:

- URI para identificar a un usuario registrado en EducaWood: `http://educawood.gsic.uva.es/user/{nombreUsuario}`, siendo `nombreUsuario` el nombre de usuario con el que se identifica al registrarse en la aplicación.

Con el diseño de estas URIs, todos los datos creados en EducaWood siguen un formato único que les permite ser accesibles dentro del conjunto LOD.

4.6. Discusión y conclusiones

En el presente capítulo se han tratado los aspectos de diseño e implementación de EducaWood. Estas dos fases se han ido desarrollando prácticamente de manera conjunta, según las necesidades de cada momento específico del proceso. Antes de empezar con la programación propiamente dicha, fue necesario diseñar la arquitectura completa de la aplicación, analizando y justificando claramente cuáles serían las tecnologías a utilizar. Las del *frontend* no dieron lugar a discusión, puesto que ya se analizaron y aprobaron con éxito tras la finalización del TFG. Sin embargo, la discusión de las tecnologías a utilizar en el *backend* sí fue decisiva para llegar al resultado deseado. Esto es debido a que la correcta integración entre *frontend* y *backend* jugaba un valor muy importante para alcanzar los objetivos del TFM. La relevancia reside en que, al tratarse de LOD, no vale cualquier herramienta de *backend*, sino que es necesario disponer de una con la que se pudiera ser capaz de tratar LOD con un grado de conocimiento de RDF y SPARQL limitado, como era mi caso. La elección de CRAFTS para ello ha jugado un papel muy importante en este desarrollo puesto que, con la ayuda de Guillermo Vega para su configuración, trabajar con LOD forestales ha resultado muy similar a haber trabajado con cualquier API REST que expusiera datos no enlazados. Por ello, considero que CRAFTS ha sido uno de los pilares fundamentales para lograr con éxito el principal objetivo del TFM.

El proceso de implementación de EducaWood ha sido el más duradero y relevante de todo el TFM. Como resultado, se dispone de una aplicación web que cumple con los requisitos expuestos en el Capítulo 3, funcional y dinámica, a disposición de cualquier persona que la quiera utilizar. En el siguiente capítulo se ejemplificarán algunos casos de uso implementados, mostrando el resultado final de la misma.

Capítulo 5

Prototipo final

5.1. Introducción

Una vez finalizado el proceso de implementación de la aplicación, es necesario desplegar la herramienta en un servidor que nos permita poder trabajar con el prototipo obtenido. En este capítulo se hablará del servidor escogido y de cómo se ha realizado su configuración. Por otro lado, se presenta una prueba de concepto para realizar varios escenarios propuestos por ingenieros forestales, que permita comprobar el correcto funcionamiento de la aplicación.

Con EducaWood se pueden desarrollar múltiples actividades y a distintos niveles de educación: primaria, secundaria y universitaria. Aunque la primera versión no soporta la realización de las actividades en la propia aplicación, ya se puede utilizar EducaWood para realizar varias actividades que involucren la educación medioambiental. Todas ellas dependen de la imaginación del docente y del nivel educativo, como por ejemplo buscar previamente una zona en el mapa donde abunde una especie y proponer a los alumnos identificar un árbol de dicha especie, o un árbol de una especie diferente. Otra opción podría ser que cada uno registre un nuevo árbol y midan su altura y diámetro, ya que con esos datos se podrán plantear diferentes actividades en el aula. Además, se puede hacer un análisis de la biodiversidad, ya que existe la opción de identificar microhábitats.

Este capítulo se organiza como sigue: en la sección 5.2 se describen los pasos necesarios para la puesta en marcha de la aplicación, mientras que en la sección 5.3 se ilustran las capacidades de EducaWood en un escenario de uso concreto. Dicho escenario fue expuesto en la presentación del Desafío Aporta. Por último, en la sección 5.4 se presentan unas conclusiones finales relacionadas con lo visto en este capítulo.

5.2. Prototipo

Durante el desarrollo de la aplicación, normalmente se utiliza el comando `ng serve` para construir, observar y servir la aplicación desde la memoria local del ordenador, utilizando `webpack-dev-server`. Sin embargo, una vez finalizada la implementación, es necesario desplegarla en producción. Para ello, el primer paso que se debe seguir es construir la aplicación con los ficheros necesarios para su funcionamiento con el comando `ng build - -prod - -base-href=/`. Tanto `ng build` como `ng serve` limpian la carpeta de salida antes de construir el proyecto, pero sólo el comando `ng build` escribe los artefactos de

construcción generados en la carpeta de salida (la carpeta de salida por defecto se llama *dist*, en ella se encuentran los ficheros de producción). A continuación, habrá que desplegar los artefactos de construcción en un servidor.

El servidor escogido es Nginx [Ngi] que, como ya se introdujo en el capítulo anterior, es un servidor web y *proxy* inverso, ligero y de alto rendimiento. La principal razón de su elección es que hace de *proxy* inverso de una máquina existente en el laboratorio del grupo de investigación GSIC-EMIC, en la que hay un servidor Nginx ejecutándose también. De esta manera, no necesitaremos dedicar una IP pública exclusiva para EducaWood. El servidor de EducaWood se encuentra desplegado en una máquina virtual creada en la nube de cómputo Openstack del grupo GSIC-EMIC y puede escalarse con los recursos que sean necesarios. Como punto SPARQL se ha utilizado el despliegue existente de Virtuoso con un grafo dedicado a EducaWood que contiene la ontología STA. Los pasos de la configuración del servidor Nginx se detallan a continuación:

1. Instalación del servidor en la máquina virtual con el comando: `sudo apt-get install nginx`.
2. Copiar los ficheros de producción en el directorio `/usr/share/nginx/html` (directorio donde Nginx lee los ficheros estáticos). Estos se encuentran en la ruta `dist/educawood-app` tras ejecutar el comando `ng build`.
3. Para finalizar con la configuración del servidor, se deben añadir las siguientes líneas de código en el fichero `/etc/nginx/sites-available/default`:

```
server {
        listen 4200;
        root /usr/share/nginx/html/dist/educawood-app;
        index index.html;
        server_name educawood.com;
        # Permite las rutas de Angular
        location / {
            try_files $uri $uri/ /index.html;
        }
}
```

4. Iniciar el servicio: `sudo service nginx start`.

Con esta configuración, ya tendremos el servicio ejecutándose en el puerto 4200 de la máquina virtual de EducaWood y podemos proceder a realizar una prueba de concepto.

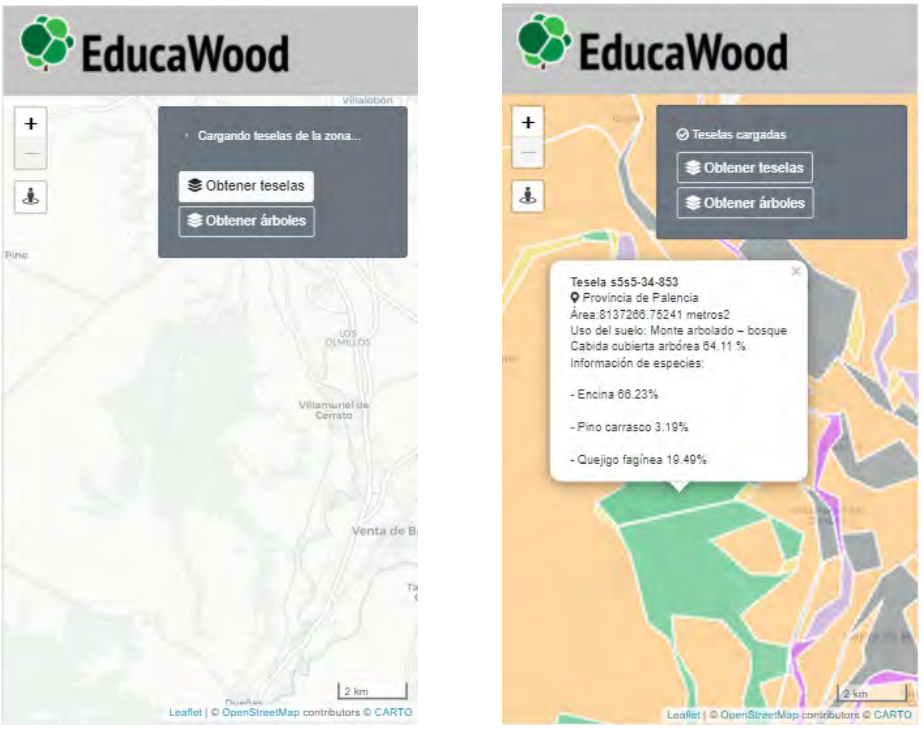
5.3. Prueba de concepto

A continuación, se expone un escenario de educación medioambiental propuesto por los expertos en ingeniería forestal del proyecto EducaWood. El escenario comienza con un profesor que da clase a alumnos de bachillerato en un instituto de Palencia. El profesor está tratando con sus alumnos las diferentes especies, familias y géneros que existen, por lo que se plantea realizar una actividad en la que los alumnos puedan identificar diferentes especies de un mismo género. Para hacerlo más interesante, el profesor decide que sería buena idea llevar a sus alumnos al Monte el Viejo, un monte muy cercano a la ciudad con un valor forestal muy importante. Una vez allí, los alumnos podrían observar las diferentes

Tabla 5.1: Escenarios de la prueba de concepto de EducaWood

Identificador	Descripción
E1	Un profesor navega por el mapa para descubrir una zona adecuada donde hacer la visita forestal.
E2	Un alumno anota un quejigo y una encina.
E3	Un segundo alumno corrige la especie anotada por el primero.
E4	Realización de una visita virtual de la zona anotada en clase.

especies, identificando cada una y compartiéndolo con sus compañeros. De esta manera, los alumnos podrán ayudarse unos a otros con las diferentes especies y corregirles si fuera necesario. En esta sección se presenta una prueba de concepto que trata de realizar el escenario propuesto con EducaWood. En la Tabla 5.1 se recoge el flujo de la prueba de concepto a realizar.



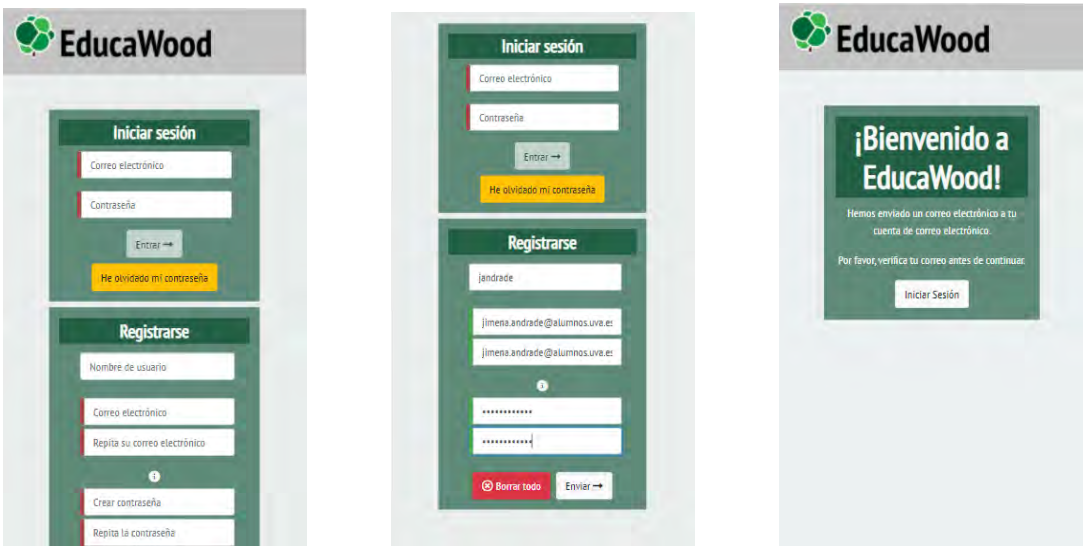
(a) Cargando teselas (b) Selección de una tesela

Figura 5.1: Navegando por el mapa para escoger la zona de interés

E1. El profesor que desea realizar la actividad de identificación de especies con sus alumnos debe conocer primero la zona donde realizarla, además de las especies que abunden en ella. Para ello, abre su navegador desde su dispositivo móvil y accede al mapa interactivo de EducaWood a través de la URL `http://educawood.gsic.uva.es/mapa`.

A continuación, se mueve por el mapa hasta encontrar el Monte el Viejo. Una vez está situado allí, selecciona el botón de "*Obtener teselas*" para cargar las teselas de dicha zona. Mientras las teselas se están cargando, en la leyenda aparece un mensaje que indica "*Cargando teselas de la zona*". Cuando este proceso ha terminado, se indica con un tick en la leyenda y se observan las diferentes teselas coloreadas según el tipo de suelo que representan cada una. En la Figura 5.1 se muestra esta transición de cargar las teselas en el mapa. El profesor selecciona la tesela situada en el Monte el Viejo. Como se muestra en la subfigura 5.1 (b), según los datos del IFN, en dicha zona predominan principalmente dos especies, la encina en un 66 % y el quejigo en un 20 %. Ambas especies pertenecen al mismo género, son *Quercus*, y dado que es fácil que las confundan, el profesor decide llevar a sus alumnos allí a realizar la actividad. Los alumnos deberán anotar un árbol de la especie encina y un quejigo.

E2. Antes de realizar la actividad propuesta por el profesor, los alumnos deben crearse una cuenta en la aplicación a través de la URL `http://educawood.gsic.uva.es/login`. En la Figura 5.2 se ilustran los pasos a seguir: (a) el alumno accede a la interfaz de *login*, la cual permite a los usuarios registrados iniciar sesión y a los nuevos, registrarse en la aplicación. (b) El alumno introduce los datos necesarios para el registro, que son un nombre de usuario, un correo electrónico y una contraseña. (c) Una vez completados los datos, pulsando a "*Enviar*',' EducaWood manda un correo electrónico al alumno para que confirme su identidad pinchando en el enlace adjunto al correo. Tras realizar este paso, el alumno ya puede realizar anotaciones en EducaWood.



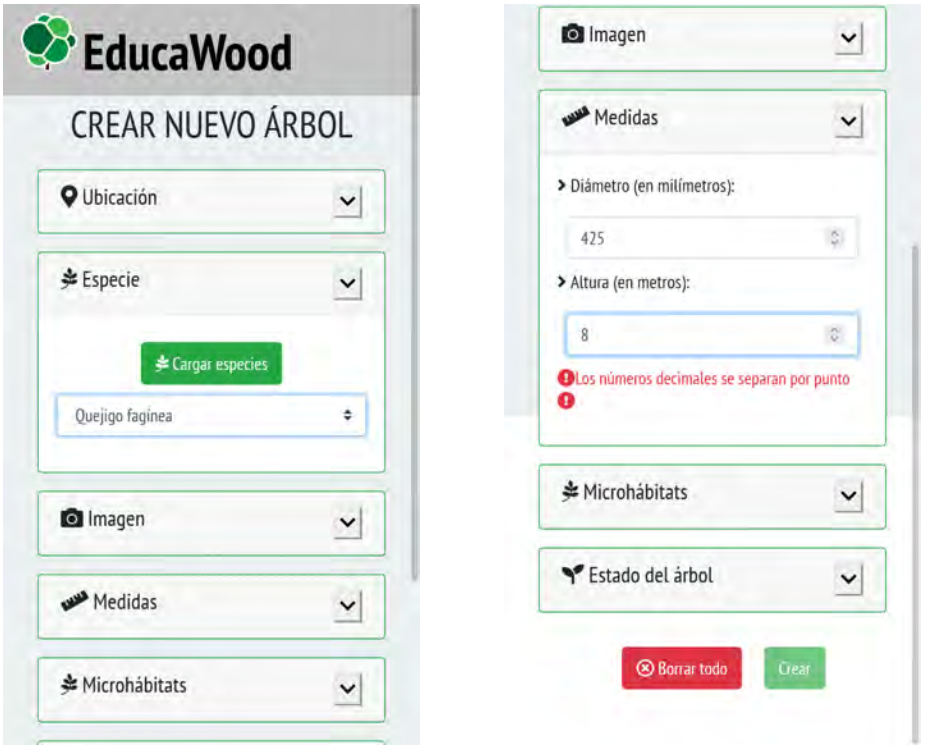
(a) Formulario de registro

(b) Completando el formulario

(c) Confirmación de registro

Figura 5.2: Registrando un usuario en la aplicación

Una vez en el monte, el alumno encuentra un árbol y, por la forma de las hojas, considera que es un quejigo. Para realizar las anotaciones en EducaWood, navega por el menú principal desde su dispositivo móvil y selecciona el icono del árbol, tras el que se le abrirá el formulario de creación de un árbol nuevo, como se muestra en la Figura 5.3. El usuario utiliza el botón de "*Usar mi ubicación actual*" para cargar las coorde-



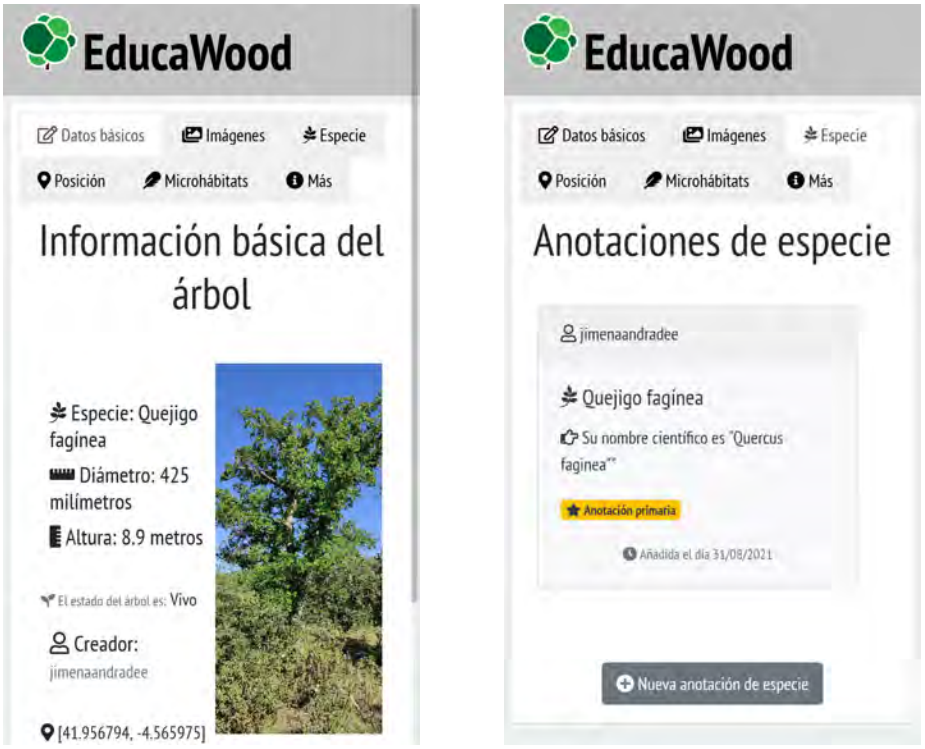
(a) Anotación de la especie (b) Anotación de las medidas

Figura 5.3: Anotando un nuevo árbol

nadas donde está ubicado el árbol mediante el GPS del propio dispositivo móvil. Encuentra la especie deseada pulsando el botón de "*Cargar especies*", como se muestra en la subfigura 5.3 (b). De igual manera, completa el resto de campos con los datos especificados en la Tabla 5.2. Tras completar el formulario, el alumno selecciona el botón de "*Crear*" para registrar el nuevo árbol en la aplicación. EducaWood realiza las tareas necesarias para adecuar los datos y mandárselos a CRAFTS. Entre ellas, establece un nuevo identificador para el árbol, que en este caso será `http://educawood.gsic.uva.es/tree/a2840420-a7cf-4e50-9adc-f7bd07a9b282` y traslada al alumno a la interfaz en la que se detallan las anotaciones del árbol realizadas (esta interfaz tiene la URL `http://educawood.gsic.uva.es/educatree/a2840420-a7cf-4e50-9adc-f7bd07a9b282`).

En la Figura 5.4 se observa la interfaz con los detalles del árbol creado, en la que se pueden comprobar los diferentes parámetros de la Tabla 5.2. En la pestaña "*Datos básicos*", el alumno comprueba cuáles son las anotaciones primarias del árbol. En este caso, como se acaba de crear y ningún otro usuario ha realizado ninguna anotación, las primarias se corresponden con las introducidas por él mismo. Para observar el resto de anotaciones del árbol, se puede navegar por las diferentes pestañas de la interfaz. Por ejemplo, observamos en la Figura 5.4 (b) las anotaciones de especie. Se observa que tan solo existe una anotación de especie, por lo que se indica por pantalla que dicha anotación es la anotación primaria.

E3. Supongamos ahora que un segundo alumno observa la anotación de especie de dicho árbol, dándose cuenta de que no se trata de un quejigo, sino de una encina. Este



(a) Anotaciones primarias (b) Anotaciones de especie

Figura 5.4: Observando las anotaciones del nuevo árbol creado

alumno, estando en la interfaz de la Figura 5.4 (b), selecciona el botón de "*Nueva anotación de especie*", con el que se le abre una ventana emergente con un formulario para introducir una nueva especie. El alumno busca en el desplegable la encina, como se muestra en la Figura 5.5, y pulsa el botón de "*Crear*". A continuación, aparece otra ventana que indica lo siguiente: "*La anotación se ha añadido con éxito. Se va a recargar la página para cargar los nuevos datos. Pueden tardar un tiempo en aparecer*", recargando la información del árbol con la nueva anotación. Si volvemos a observar la información básica de dicho árbol, y las anotaciones de especie del mismo (Figura 5.6), observaremos que la anotación primaria de especie ha cambiado, siendo ahora una encina. De esta manera, el segundo alumno ha podido corregir a su compañero. También podría, por ejemplo, completar la información del árbol añadiendo una anotación de microhábitat.

E4. Una vez finalizada la actividad en el campo, y estando ya en el aula, tanto alumnos como profesores podrán ir observando los diferentes árboles anotados en la aplicación a través del mapa forestal. Para ello, el profesor vuelve a entrar en EducaWood, esta vez desde su ordenador portátil, para poder proyectarlo en la pantalla del proyector del aula. Seleccionando el botón "*Obtener árboles*" de la leyenda, se cargan todos los árboles registrados en el sistema. En la Figura 5.7 se muestran de color verde los árboles procedentes del IFN, mientras que de color morado, los árboles creados por los alumnos. Además, el icono de cada árbol es distinto en función de la especie del mismo. En la captura se observan varios árboles creados, fruto de la actividad propuesta por el profesor. Seleccionando cada uno de los árboles, se pueden observar sus datos básicos, además del creador, como se ilustra en la Figura 5.8. Con el identificador del árbol será posible acceder a su informa-

Tabla 5.2: Parámetros del árbol anotado por el usuario

Parámetro	**Valor**
Latitud	41,956794
Longitud	-4,565975
Especie	Quejigo fajínea
Imagen	*ruta donde se encuentra*
Título	Quejigo de Jimena
Descripción	Identificación de un quejico en el Monte el Viejo
Parte	Foto general
Altura	8.9
Diámetro	425
Estado	vivo

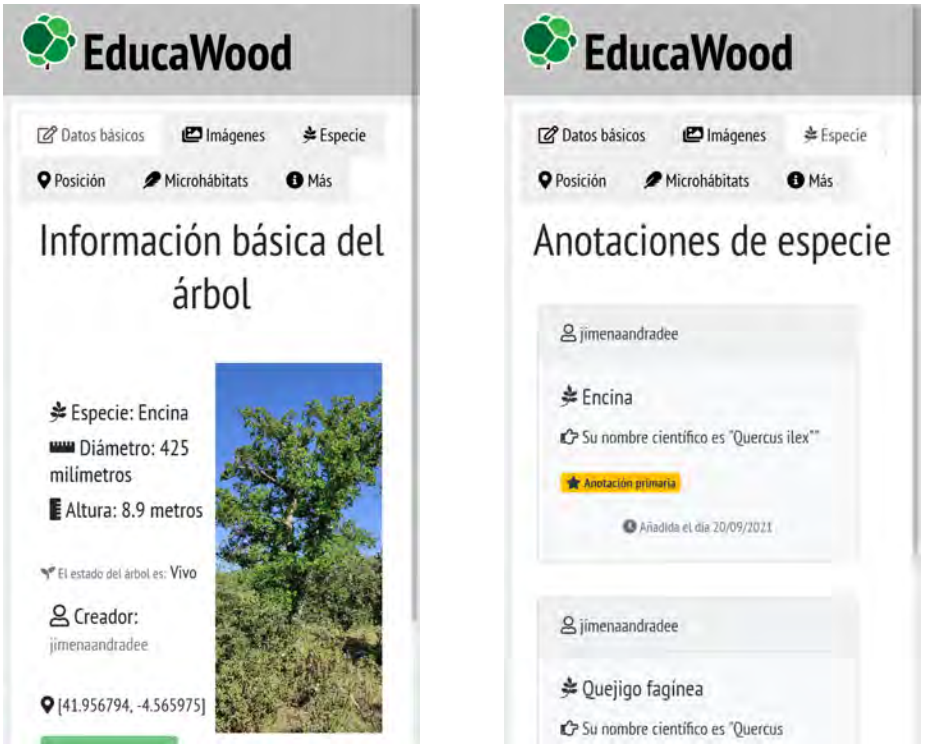
ción completa. Para ello, se debe copiar el identificador del árbol y pegarlo en la barra de búsqueda disponible en el encabezado de la aplicación. De esta manera, el profesor puede ir comprobando con los alumnos si las especies identificadas son correctas. En caso de que alguna especie no lo fuera, bastaría con añadir una nueva anotación de especie con el botón *Nueva anotación de especie* en la parte inferior de la Figura **??**. Una vez corregida la información relativa a las especies, se podrán utilizar los datos recogidos en la aplicación para realizar otro tipo de actividades, como por ejemplo, estimar el carbono fijado de cada uno de los árboles escogidos.

5.4. Conclusiones

En el presente capítulo se ha realizado el despliegue de EducaWood en un servidor Nginx para poder comprobar su correcto funcionamiento. Nginx juega el papel de proxy inverso para redireccionar el servicio a una IP pública de una de las máquinas del grupo de investigación. Su configuración resulta relativamente sencilla. Una vez se ha conseguido tener funcionando el servicio, se ha expuesto un escenario de uso de EducaWood consistente en la identificación de especies arbóreas que podría ponerse en práctica con alumnos de secundaria y bachillerato. Se ha mostrado cómo la aplicación da soporte a la visualización del uso del suelo de zona específica del territorio español, facilitando a los profesores la elección del lugar óptimo en el que realizar las actividades medioambientales con los alumnos. Además, EducaWood dispone de un formulario donde los alumnos pueden realizar las anotaciones de manera sencilla y, posteriormente, poder compartirlas y comentarlas con los demás compañeros y profesores. Con el uso de la aplicación, se espera que los alumnos despierten mayor curiosidad por el medio ambiente, prestando más atención a su entorno en las salidas al campo, puesto que si les gusta la aplicación, también podrán hacer uso de ella fuera del horario escolar. El resto de funcionalidades disponibles en esta primera versión de EducaWood se pueden consultar en el manual de usuario disponible en el Anexo E.

Figura 5.5: Añadiendo una nueva anotación de especie

El prototipo ha sido puesto a prueba por diferentes usuarios hasta la fecha, cuyo experiencia con la aplicación ha sido satisfactoria. Como resultado, ya se tienen registrados una gran variedad de árboles por todo el territorio español, publicados como datos abiertos enlazados. Finalmente, se espera que la aplicación pueda probarse en un ámbito educativo, puesto que las pruebas hasta el momento han sido realizadas fuera de dicho ámbito.



(a) Anotación primaria de especie actualizada

(b) Anotaciones de especie

Figura 5.6: Observando las anotaciones corregidas del árbol

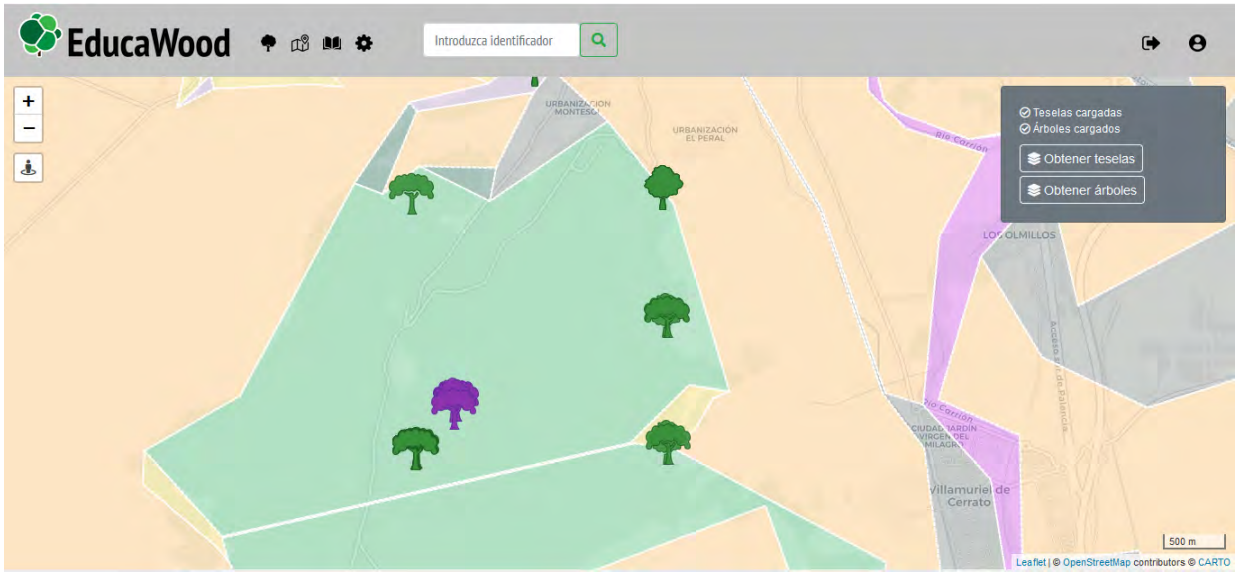


Figura 5.7: Mapa forestal situado en el Monte el Viejo con árboles del IFN (verdes) y creados por alumnos (morados)

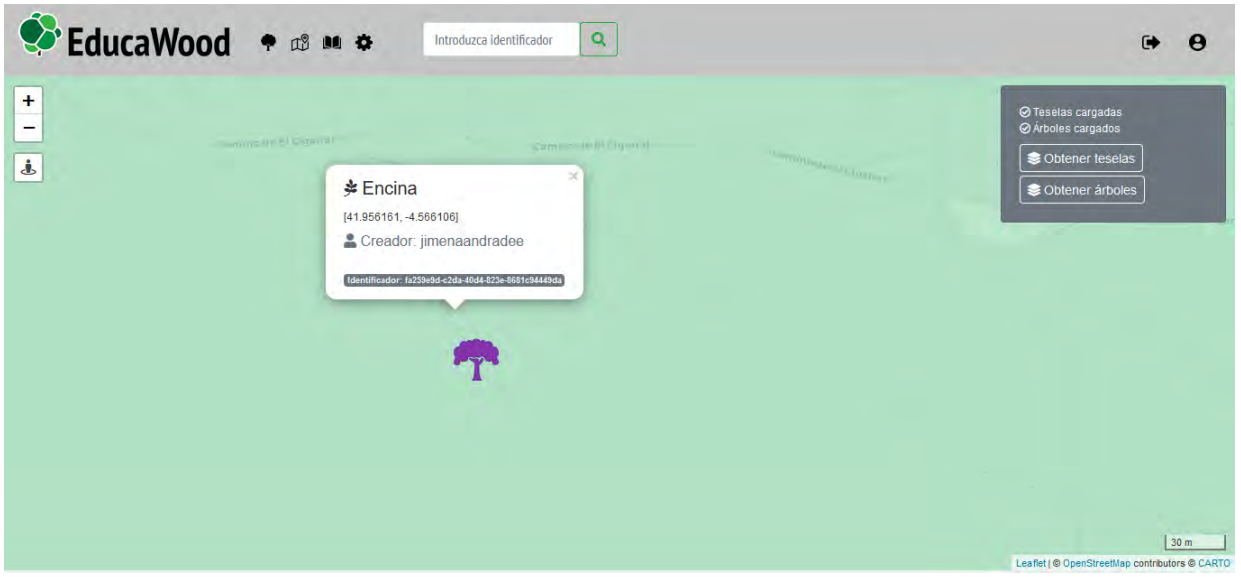


Figura 5.8: Observando en el mapa los detalles de un árbol

Capítulo 6

Conclusiones y líneas de trabajo futuro

6.1. Introducción

En este último capítulo del TFM se exponen las conclusiones finales sobre todo el trabajo realizado a lo largo de este proyecto. Además, se redactan varias funcionalidades que podrían implementarse como trabajo futuro para continuar con el desarrollo de EducaWood.

6.2. Conclusiones del trabajo realizado

En este Trabajo Fin de Máster se ha analizado, diseñado, desarrollado y puesto en marcha EducaWood, una aplicación web de carácter educativo y medioambiental que permite visualizar y generar datos abiertos enlazados. EducaWood permite dar apoyo a situaciones de aprendizaje ubicuo medioambiental que favorezcan las salidas al campo y la toma de conciencia medioambiental por parte de los alumnos.

Para lograr las tres metas planteadas en el Capítulo 1, se han tenido que hacer frente a varios problemas. El primero de ellos, y más relevante, está relacionado con la forma en la que se publican los datos abiertos en la Web. La publicación de estos datos sin estructura única y utilizando diversos formatos conlleva a dificultades de integración de los mismos, inconsistencias y mala documentación. A este problema hay que sumarle la falta de herramientas disponibles para poder estructurar los datos almacenados. Todo esto se traduce en una falta de reutilización de los mismos. Como solución a estos problemas, se recurrió a la Web Semántica. La aplicación de tecnologías semánticas en el ámbito forestal permite la publicación de datos abiertos enlazados accesibles al público general, su vinculación a ontologías relacionadas con territorios geográficos o de descripción de especies y su posterior reutilización en aplicaciones muy diversas, gracias a su estructura única bien definida.

Para conseguir la segunda meta, se ha hecho frente a otro problema, el cual surge como consecuencia de la primera solución. Hoy en día existe una comunidad de desarrolladores web ampliamente extendida que no conoce nada de los fundamentos de la Web Semántica ni de sus tecnologías. Por lo que el segundo reto del TFM ha consistido en buscar soluciones que solventasen dicho problema. Existen varias aproximaciones, entre la que destaca CRAFTS, una herramienta que permite crear APIs REST que utilicen datos de varios

conjuntos LOD. Utilizando esta herramienta se han conseguido dos objetivos clave, (i) utilizar LOD forestales sin disponer de un alto grado de conocimiento de las tecnologías de la Web semántica e (ii) integrar LOD de sitios muy diversos, como del Inventario Forestal Nacional, del Mapa Forestal español o de la DBpedia. Desde mi rol como desarrolla web, con un nivel reducido de conocimiento de tecnologías semánticas, el uso de CRAFTS ha resultado ser muy satisfactorio debido a su facilidad de uso.

Finalmente, la tercera meta planteada ha sido a la que más tiempo se le ha dedicado: el diseño e implementación de un *frontend* funcional. Dado que la propuesta presentada en el Desafío Aporta no iba a ser posible de abarcar en el TFM, la frontera se situó en implementar la herramienta para que pudiera ser utilizada por los alumnos, dejando a un lado la parte de autoría de actividades para los profesores. Teniendo en cuenta la parte implementada, por un lado, se ha trabajado con la librería *Leaflet* para conseguir la visualización de datos geográficos como son las teselas y los árboles, tanto del IFN como los publicados por los usuarios de EducaWood. La comunicación con CRAFTS ha sido imprescindible para poder visualizar y generar LOD, puesto que supone el puente entre el *frontend* y los diferentes puntos SPARQL utilizados en la aplicación. Por otro lado, ha sido necesario la utilización de servicios externos, tanto para almacenar datos que involucrasen información confidencial de los usuarios como para almacenar las imágenes publicadas.

Poniendo en práctica estas ideas y soluciones se tiene como resultado el presente Trabajo Fin de Máster, el cual ha sido motivado e impulsado por el Desafío Aporta, un concurso a nivel nacional que contó con un total de 22 candidaturas y del que se logró obtener el tercer premio (se puede consultar aquí la entrevista realizada en la entrega de premios[1]). Además del trabajo aquí realizado, EducaWood ha sido publicado en la ECTEL 2021[2] (*European Conference on Technology Enhanced Learning*) [AH21b]. Por todo ello, considero que los objetivos académicos que se plantearon al iniciar este TFM han sido cumplidos satisfactoriamente. Me gustaría destacar además que se han logrado objetivos personales más allá de los especificados, puesto que considero que con el Desafío Aporta he desarrollado una mayor capacidad de responsabilidad a la hora de liderar un equipo, habilidades de creación de contenidos multimedia para dar a conocer la idea y habilidades de defensa de mi prototipo ante un jurado externo a la universidad. Por todo ello, considero que el trabajo realizado ha sido muy enriquecedor para mi futuro como ingeniera de telecomunicaciones.

6.3. Líneas de trabajo futuro

La propuesta inicial presentada para el Desafío Aporta era muy ambiciosa como para abarcar todo el TFM. Pese a que el objetivo básico que debía cumplir si que se ha logrado (reutilización de datos abiertos para la educación medioambiental), la aplicación inicialmente pensada no se ha desarrollado completamente, dejándose fuera la implementación de EducaWood como herramienta de autoría de actividades para profesores. En líneas generales, se espera que EducaWood sea una herramienta de autoría en la que se puedan soportar diferentes tipos de actividades como las propuestas en la Tabla 2.1, más allá de la realización de anotaciones. Por lo que será necesario implementar los casos de uso relacionados con la gestión de actividades identificados en la sección 3.4.4. Además, será

[1] `https://www.youtube.com/watch?v=B1jXpn8t_bg`
[2] `https://ea-tel.eu/ectel2021`

necesario establecer diferentes roles entre usuarios, que distinga a alumnos de profesores. El rol del profesor tendrá capacidad de plantear las actividades soportadas, además de corregir anotaciones de sus alumnos. Por otro lado, será necesario mejorar el sistema de curado de datos puesto en marcha actualmente, para que permita una colaboración entre pares y teniendo en cuenta también la valoración de los profesores.

Centrándonos más en la implementación propiamente dicha de la aplicación, se pueden realizar ciertas mejoras que favorecerán el uso de la misma. En primer lugar, sería deseable introducir en EducaWood una funcionalidad que permitiera su uso en situaciones en las que no hubiera cobertura o el usuario no disponga de conexión a Internet, ya que es probable que en el campo o en el bosque se den este tipo de situaciones. Para ello, habría que introducir un *service worker*[3] en la aplicación. Los *Service Workers* son *scripts* en JavaScript que se ejecutan en segundo plano en la aplicación. Actúan esencialmente como servidores *proxy* asentados entre las aplicaciones web, el navegador y la red (cuando está accesible). Están destinados a permitir la creación de experiencias *offline* efectivas, interceptando peticiones de red y realizando las acciones apropiadas en el caso en el que la conexión de red esté disponible. También permiten el acceso a notificaciones tipo *push* y APIs de sincronización en segundo plano. Seguramente una precarga de datos previa podría ser necesaria, de manera que para visualizaciones sólo podría mostrarse lo que estuviese precargado, mientras que para anotaciones, lo apropiado sería que se encolaran las peticiones de anotación y que se ejecutaran al volver a tener red. Con la introducción de esta funcionalidad en la aplicación se permitirá el uso de la misma en cualquier lugar, independientemente de si existe buena conexión a Internet o no. Además, Angular dispone de una librería para su implementación.

Por otro lado, se podrían realizar ciertas mejoras en el mapa implementado con LeafLet [Lea]. Por un lado, se podría implementar un algoritmo que permitiera realizar las peticiones de los datos a cargar en el mapa tan solo de las zonas que no se hayan cargado anteriormente. Así, la navegación por el mapa resultaría ligeramente más fluida. Por otro lado, sería interesante diseñar la URL del mapa de manera que contuviera las coordenadas de la zona en la que se encuentre en cada momento. De esta manera, sería posible volver a una zona en la que se haya navegado anteriormente sin necesidad de tener que volver a buscarla cuando se recargue la página.

Algunas otras mejoras que se podrían realizar sobre esta versión de EducaWood serían: (i) permitir las acciones de modificación y borrado de anotaciones creadas por un usuario, (ii) añadir una funcionalidad que permita al usuario realizar la foto del árbol en el momento de la creación del formulario y no solo subirla desde la galería de imágenes e (iii) implementar un pequeño buscador en el mapa para que se puedieran buscar árboles según las diferentes propiedades que se pueden anotar.

[3] `https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API/Using_Service_Workers`

Referencias

[AH20] Jimena Andrade-Hoz. Diseño y desarrollo de una aplicación web para el etiquetado socio-semántico en el ámbito de la gestión forestal. *ETSI Telecomunicación. Universidad de Valladolid*, 2020. Trabajo Fin de Grado, Grado en Ingeniería de Tecnologías de Telecomunicación.

[AH21a] Jimena Andrade-Hoz. Educawood, 2021. GSIC-EMIC. Universidad de Valladolid. Repositorio GitHub. Último acceso septiembre 2021. https://github.com/gsic-emic/EducaWood.

[AH21b] Jimena Andrade-Hoz, Guillermo Vega-Gorgojo, Irene Ruano, Miguel L Bote-Lorenzo, Juan I Asensio-Pérez, Felipe Bravo, y Cristóbal Ordóñez. Educawood: A socio-semantic annotation system for environmental education. En *European Conference on Technology Enhanced Learning*, páginas 368–372. Springer, 2021.

[All11] Dean Allemang y Jim Hendler. *Semantic web for the working ontologist: effective modeling in RDFS and OWL*, chapter 1, páginas 2 – 11. Elsevier, 2ª edición, 2011.

[Ang] Angular. The modern webdeveloper's platform. [Website]. Último acceso mayo 2021. https://angular.io/.

[Apo20] Iniciativa Aporta. Cómo contribuir a mejorar la educación digital a través del desafío aporta, 2020. [Website] Disponible en: https://datos.gob.es/es/blog/como-contribuir-mejorar-la-educacion-digital-traves-del-desafio-aporta.

[BA13] Carlos Buil-Aranda, Aidan Hogan, Jürgen Umbrich, y Pierre-Yves Vandenbussche. Sparql web-querying infrastructure: Ready for action? En *International Semantic Web Conference*, páginas 277–293. Springer, 2013.

[Biz11] Christian Bizer, Tom Heath, y Tim Berners-Lee. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, chapter Linked Data: The Story So Far, páginas 205–228. IGI Global snippet, 2011.

[BL01] Tim Berners-Lee, James Hendler, y Ora Lassila. The semantic web: Scientific american. *Scientific American*, 2001.

[BL06] Tim Berners-Lee. Linked data, 2006. [Website]. Último acceso: Junio 2009. Disponible en: https://www.w3.org/DesignIssues/LinkedData.html.

[Boo] Bootstrap. Build fast, responsive sites with Bootstrap. [Website]. Último acceso mayo 2021. https://getbootstrap.com/.

[Bre09] John G Breslin, Alexandre Passant, y Stefan Decker. *The social semantic web*. Springer Science & Business Media, 2009.

[Bry13a] P. Bryan y M. Nottingham. Javascript object notation (json) patch [rfc6902]. Informe técnico, nternet Engineering Task Force (IETF), 2013. [Website] Disponible en: https://datatracker.ietf.org/doc/html/rfc6902.

[Bry13b] Paul Bryan, Kris Zyp, y Mark Nottingham. Javascript object notation (json) pointer [rfc6901]. Informe técnico, Internet Engineering Task Force (IETF), 2013. [Website] Disponible en: https://datatracker.ietf.org/doc/html/rfc6901.

[CF20] Cross-Forest, 2020. [Website]. Último acceso julio 2021. https://crossforest.eu/.

[Che19] Shu-Chen Cheng, Gwo-Jen Hwang, y Chih-Hung Chen. From reflective observation to active learning: A mobile experiential learning approach for environmental science education. *British Journal of Educational Technology*, 50:1–20, 2019.

[Cyg14] Richard Cyganiak, David Wood, y Markus Lanthaler. Rdf 1.1 concepts and abstract syntax. Informe técnico, W3C Recommendation, 2014. [Website] Disponible en: https://www.w3.org/TR/rdf11-concepts/.

[CyL21] Proyecto cambio climático en cyl, 2021. Observation.org. [Website]. Último acceso 2021. https://observation.org/projects/47/.

[Daq20] Marilena Daquino, Ivan Heibi, Silvio Peroni, y David Shotton. Creating restful apis over sparql endpoints with ramose. 2020. Accepted for publication in the Semantic Web Journal. Disponible en: http://www.semantic-web-journal.net/content/creating-restful-apis-over-sparql-endpoints-using-ramose-0.

[Dea09] John Deacon. Model-view-controller (mvc) architecture. *Online][Citado em: 10 de março de 2006.] http://www. jdl. co. uk/briefings/MVC. pdf*, 2009.

[Der14] Olga Derevenskaia. Active learning methods in environmental education of students. *Procedia-Social and Behavioral Sciences*, 131:101–104, 2014.

[Dus10] L. Dusseault y J. Snell. Patch method for http [rfc5789]. Informe técnico, Internet Engineering Task Force (IETF), 2010. [Website] Disponible en: https://datatracker.ietf.org/doc/html/rfc5789.

[Fie99] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, y Tim Berners-Lee. Hypertext transfer protocol – http/1.1. Informe técnico, Network Working Group, 1999. [Website] Disponible en: https://datatracker.ietf.org/doc/html/rfc2616.

[Fira] Authenticate with firebase using password-based accounts using javascript. [Website]. Último acceso febrero 2021. https://firebase.google.com/docs/auth/web/password-auth?hl=es.

[Firb] Cloud storage en la web. [Website]. Último acceso marzo 2021. https://firebase.google.com/docs/storage/web/start.

[Firc] Firebase. [Website]. Último acceso julio 2021. https://firebase.google.com/.

[Gar20] Daniel Garijo y Maximiliano Osorio. Oba: An ontology-based framework for creating rest apis for knowledge graphs. En *International Semantic Web Conference*, páginas 48–64. Springer, 2020.

[Har13] Steve Harris y Andy Seaborne. Sparql 1.1 query language. Informe técnico, W3C Recommendation, 2013. [Website] Último acceso mayo 2021. https://www.w3.org/TR/2013/REC-sparql11-query-20130321/.

[Hwa11] Gwo-Jen Hwang y Chin-Chung Tsai. Research trends in mobile and ubiquitous learning: A review of publications in selected journals from 2001 to 2010. *British Journal of Educational Technology*, 42(4):E65–E70, 2011.

[Hyl13] Bernadette Hyland, Atemezing Ghislain, Pendleton Michael, y Srivastava Biplav. Linked data glossary, 2013. [Website] Último acceso mayo 2021. https://www.w3.org/TR/ld-glossary/.

[Hyl14] Bernadette Hyland, Ghislain Atemezing, y Boris Villazon-Terrazas. Best practices for publishing linked data, 2014. [Website] Último acceso junio 2021. https://www.w3.org/TR/ld-bp/.

[IFN21] Inventario forestal nacional, 2021. Gobierno de España, Ministerio para la Transición Ecológica y el Reto Demográfico. [Website]. Último acceso 2021. https://www.miteco.gob.es/es/biodiversidad/temas/inventarios-nacionales/inventario-espanol-patrimonio-natural-biodiv/sistema-indicadores/04c-I-forestal-nacional.aspx.

[ITM21] Integrate tree microhabitat app, 2021. Integrate Network. [Website]. Último acceso 2021. https://informar.eu/tree-microhabitats.

[Jac00] Ivar Jacobson, Grady Booch, James Rumbaugh, y Salvadortr Sánchez. *El proceso unificado de desarrollo de software*. 2000.

[Jon12] Michael Jones y Dick Hardt. The oauth 2.0 authorization framework: Bearer token usage. Informe técnico, RFC 6750, October, 2012.

[Kit14] Rob Kitchin. *The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences*. SAGE Publications, 2014.

[Lea] Leaflet. [Website]. Último acceso julio 2021. https://leafletjs.com/.

[Lis19] Pasquale Lisena, Albert Meroño-Peñuela, Tobias Kuhn, y Raphaël Troncy. Easy web api development with sparql transformer. En *International Semantic Web Conference*, páginas 454–470. Springer, 2019.

[Loh15] Steffen Lohmann, Vincent Link, Eduard Marbach, y Stefan Negru. WebVOWL: Web-based visualization of ontologies. En *Proceedings of EKAW 2014 Satellite Events*, volumen 8982 of *LNAI*, páginas 154–158. Springer, 2015.

[Ló17] Bernadette Farias Lóscio, Caroline Burle, y Newton Calegari. Data on the web best practices, 2017. W3C Recommendation. Disponible en: https://www.w3.org/TR/2017/REC-dwbp-20170131/.

[MFE21] Mapa forestal de españa (mfe50), 2021. Gobierno de España, Ministerio para la Transición Ecológica y el Reto Demográfico. [Website]. Último acceso 2021. https://www.miteco.gob.es/es/biodiversidad/servicios/banco-datos-naturaleza/informacion-disponible/mfe50.aspx.

[Mik13] Michael Mikowski y Josh Powell. *Single page web applications: JavaScript end-to-end*, chapter 1, Introducing SPAs. Simon and Schuster, 2013.

[Mil20] Darrel Miller, Jeremy Whitlock, Marsh Gardiner, Mike Ralphson, Ron Ratovsky, y Uri Sarid. Openapi specification v3.0.3. Informe técnico, OPENAPI Initiative, 2020. [Website] Disponible en: https://spec.openapis.org/oas/v3.0.3.

[MP16] Albert Meroño-Peñuela y Rinke Hoekstra. grlc makes github taste like linked data apis. En *European Semantic Web Conference*, páginas 342–353. Springer, 2016.

[MS20] Cristina Mayo-Sarmiento. Diseño e implementación del servidor de una aplicación socio-semántica de anotación de árboles, trabajo fin de máster, máster en ingeniería de telecomunicación. *ETSI Telecomunicación. Universidad de Valladolid*, 2020.

[NGA] National Geospatial-Intelligence Agency (NGA). WORLD GEODETIC SYSTEM 1984 (WGS 84) [Website] https://earth-info.nga.mil/index.php?dir=wgs84action=wgs84.

[Ngi] Nginx. Admin Guide. [Website]. Último acceso julio 2021. https://docs.nginx.com/nginx/admin-guide/.

[Obs21] 2021. Observation.org. [Website]. Último acceso 2021. https://observation.org/.

[O'H12] Kieron O'Hara y Wendy Hall. Semantic web. 2012.

[Ope20] OpenWebinars. Curso de buenas prácticas en angular, 2020. [Website]. Último acceso 2021. https://openwebinars.net/cursos/curso-buenas-practicas-angular/.

[Ram03] Raghu Ramakrishnan, Johannes Gehrke, y Johannes Gehrke. *Database management systems*, volumen 3. McGraw-Hill New York, 2003.

[RR19] Begoña Rivas-Rebaque, Felipe Gértrudix-Barrio, y Julio César de Cisneros de Britto. La percepción del docente universitario ante el uso y valor de los datos abiertos. *Educación XX1*, 22(2):141–163, 2019.

[Spe15] Steve Speicher, John Arwe, y Ashok Malhotra. Linked data platform 1.0, 2015. [Website] Disponible en: https://www.w3.org/TR/2015/REC-ldp-20150226/.

[Spo20] Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, Pierre-Antoine Champin, y Niklas Lindström. Json-ld 1.1, 2020. [Website] Disponible en: https://www.w3.org/TR/2020/REC-json-ld11-20200716/.

[Ste15] Rod Stephens. *Beginning software engineering*, chapter 4, Requirement Gathering, páginas 53 – 87. John Wiley & Sons, 2015.

[Val17a] Joseph S Valacich, Joey F George, y Joseph S Valacich. *Modern systems analysis and design*, volumen 9, chapter 6, Determining System requirements, páginas 147 – 181. Pearson Boston, 2017.

[Val17b] Joseph S Valacich, Joey F George, y Joseph S Valacich. *Modern systems analysis and design*, volumen 9, chapter 1, the Systems Development environment, páginas 3 – 25. Pearson Boston, 2017.

[Val17c] Joseph S Valacich, Joey F George, y Joseph S Valacich. *Modern systems analysis and design*, volumen 9, chapter 4, Identifying and Selecting Systems Development Projects, páginas 87 – 110. Pearson Boston, 2017.

[Ver16] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, y Pieter Colpaert. Triple pattern fragments: a low-cost knowledge graph interface for the web. *Journal of Web Semantics*, 37:184–206, 2016.

[VG21] Guillermo Vega-Gorgojo. Crafts: Configurable rest apis for triple stores. *Semantic Web*, páginas 1–17, 2021. http://www.semantic-web-journal.net/content/crafts-configurable-rest-apis-triple-stores-0.

[Vir] OpenLink Software. Virtuoso Open-Source Edition. [Website]. Último acceso julio 2021. http://vos.openlinksw.com/owiki/wiki/VOS.

Apéndice A

Esquema de configuración de una API CRAFTS

```
{
  "apiId": "string",
  "endpoints": [
    {
      "id": "string",
      "sparqlURI": "string",
      "graphURI": "string",
      "httpMethod": "GET",
      "authInfo": {
        "user": "string",
        "password": "string",
        "type": "basic"
      },
      "sparqlUpdate": {
        "id": "string",
        "sparqlURI": "string",
        "graphURI": "string",
        "httpMethod": "GET",
        "authInfo": {
          "user": "string",
          "password": "string",
          "type": "basic"
        }
      }
    }
  ],
  "model": [
    {
      "id": "string",
      "types": [
        {
          "label": "string",
          "endpoint": "string",
```

```
          "inferred": true,
          "restrictions": [
            "string"
          ],
          "targetId": "string",
          "embed": true,
          "writeonly": true
        }
      ],
      "dprops": [
        {
          "label": "string",
          "endpoint": "string",
          "iri": "string",
          "restrictions": [
            "string"
          ],
          "writeonly": true
        }
      ],
      "oprops": [
        {
          "label": "string",
          "endpoint": "string",
          "iri": "string",
          "inv": true,
          "restrictions": [
            "string"
          ],
          "targetId": "string",
          "embed": true,
          "writeonly": true
        }
      ]
    }
  ],
  "queryTemplates": [
    {
      "id": "string",
      "endpoint": "string",
      "description": "string",
      "template": "string",
      "variables": [
        "string"
      ],
      "parameters": [
        {
          "label": "string",
          "type": "iri",
```

```
          "optional": true
        }
      ]
    }
  ]
}
```

Apéndice B

Ontología para la anotación social de árboles

```
@prefix sta: <http://educawood.gsic.uva.es/sta/ontology/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dc: <http://purl.org/dc/terms/> .
@prefix ifn: <http://crossforest.eu/ifn/ontology/> .
@prefix vann: <http://purl.org/vocab/vann/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix w3cgeo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .


sta: a owl:Ontology ;
    rdfs:label "Social tree annotation ontology"@en , "Ontología de anotación
    social de árboles"@es ;
    rdfs:comment "Ontology for socially annotation of trees, logs, and plant
    microhabitats"@en , "Ontología para la anotación social de árboles,
    troncos caídos y microhábitats de plantas"@es ;
    owl:versionInfo 0.3 ;
    dc:creator <https://www.gsic.uva.es/members/cmayo>,
    <https://www.gsic.uva.es/members/guiveg> ;
    dc:date "2021-04-30"^^xsd:date ;
    vann:preferredNamespacePrefix "sta" . #The preferred namespace
    prefix to use when using terms from this vocabulary in an XML
    document.


#################################################################
#    Classes
#################################################################

sta:SpatialEntity a owl:Class , rdfs:Class ;
  rdfs:subClassOf w3cgeo:SpatialThing ;
```

```
  rdfs:label "Spatial entity"@en , "Entidad espacial"@es ;
  rdfs:isDefinedBy sta: .

sta:Plant a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:SpatialEntity ;
  rdfs:label "Plant"@en , "Planta"@es ;
  rdfs:isDefinedBy sta: .

sta:Tree a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:Plant ;
  rdfs:label "Tree"@en , "Árbol"@es ;
  rdfs:comment "Woody perennial plant with an elongated stem
  supporting branches"@en , "Planta vivaz (que vive más de dos años,
  de tallo leñoso, que se ramifica a cierta altura del suelo"@es ;
  rdfs:isDefinedBy sta: .

sta:Log a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:Plant ;
  rdfs:label "Log"@en , "Tronco caído"@es ;
  rdfs:comment "Felled trunk not rooted in the ground"@en , "Tronco
  caído sin raíces en la tierra"@es ;
  rdfs:isDefinedBy sta: .

sta:Microhabitat a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:SpatialEntity ;
  rdfs:label "Microhabitat"@en , "Microhábitat de una planta"@es ;
  rdfs:comment "Plant microhabitat"@en , "Microhábitat de una planta"@es ;
  rdfs:isDefinedBy sta: .


# anotaciones
sta:Annotation a owl:Class, rdfs:Class ;
  rdfs:label "Annotation"@en , "Anotación"@es ;
  rdfs:comment "Tree annotation"@en , "Anotación de un árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:SpeciesAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Species annotation"@en , "Anotación de especie"@es ;
  rdfs:comment "Annotation about a tree species"@en , "Anotación
  sobre la especie de un árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:PositionAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Position annotation"@en , "Anotación de posición"@es ;
  rdfs:comment "Annotation about a tree position"@en , "Anotación
  sobre la posición de un árbol"@es ;
  rdfs:isDefinedBy sta: .
```

```
sta:DiameterAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Diameter annotation"@en , "Anotación de diámetro"@es ;
  rdfs:comment "Annotation about a tree diameter (typically at breast
  height)"@en , "Anotación sobre el díametro de un árbol (típicamente a
  la altura del pecho)"@es ;
  rdfs:isDefinedBy sta: .

sta:HeightAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Height annotation"@en , "Anotación de altura"@es ;
  rdfs:comment "Annotation about a tree height"@en , "Anotación
  sobre la altura de un árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:ImageAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Image annotation"@en , "Anotación de imagen"@es ;
  rdfs:comment "Annotation about a tree image"@en , "Anotación
  sobre la imagen de un árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:Image a owl:Class , rdfs:Class ;
  rdfs:subClassOf foaf:Image ;
  rdfs:label "Image"@en , "Imagen"@es ;
  rdfs:isDefinedBy sta: .


# anotaciones estado árbol
sta:TreeStatusAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Tree status annotation"@en , "Anotación del estado
  de un árbol"@es ;
  rdfs:comment "Annotation about tree status"@en , "Anotación
  sobre el estado de un árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:AliveTreeAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:TreeStatusAnnotation ;
  rdfs:label "Alive tree annotation"@en , "Anotación de árbol vivo"@es ;
  rdfs:comment "Annotation about alive tree status"@en , "Anotación
  sobre el estado de árbol vivo"@es ;
  rdfs:isDefinedBy sta: .

sta:DecliningTreeAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:TreeStatusAnnotation ;
  rdfs:label "Declining tree annotation"@en , "Anotación de árbol en
  declive"@es ;
```

```
  rdfs:comment "Annotation about declining tree status"@en ,
  "Anotación sobre el estado de árbol en declive"@es ;
  rdfs:isDefinedBy sta: .

sta:DeadTreeAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:TreeStatusAnnotation ;
  rdfs:label "Dead tree annotation"@en , "Anotación de árbol
  muerto"@es ;
  rdfs:comment "Annotation about dead tree status"@en , "Anotación
  sobre el estado de árbol muerto"@es ;
  rdfs:isDefinedBy sta: .

sta:DeadTreeWithBarkAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:DeadTreeAnnotation ;
  rdfs:label "Dead tree with bark annotation"@en , "Anotación de
  árbol muerto con corteza"@es ;
  rdfs:comment "Annotation about dead tree with bark status.
  Bark is still OK"@en , "Anotación sobre el estado de árbol muerto
  con corteza. La corteza está todavía intacta"@es ;
  rdfs:isDefinedBy sta: .

sta:DeadTreeWithLooseBarkAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:DeadTreeAnnotation ;
  rdfs:label "Dead tree with loose bark annotation"@en , "Anotación de
  árbol muerto con corteza suelta"@es ;
  rdfs:comment "Annotation about dead tree with loose bark status.
  Bark is scarce or missing, stem is still hard"@en , "Anotación sobre el
  estado de árbol muerto con corteza suelta. Queda poco o nada de
  corteza, la superficie está dura"@es ;
  rdfs:isDefinedBy sta: .

sta:DeadTreeWithNoBarkAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:DeadTreeAnnotation ;
  rdfs:label "Dead tree with no bark annotation"@en , "Anotación de
  árbol muerto sin corteza"@es ;
  rdfs:comment "Annotation about dead tree with no bark status.
  No bark, stem is wet and falls when cut"@en , "Anotación sobre el
  estado de árbol muerto sin corteza. No hay corteza, la superficie
  está húmeda y se cae cuando se corta"@es ;
  rdfs:isDefinedBy sta: .

sta:DecomposedTreeAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:DeadTreeAnnotation ;
  rdfs:label "Decomposed dead tree annotation"@en , "Anotación de
  árbol descompuesto"@es ;
  rdfs:comment "Annotation about decomposed dead tree status.
  Stem is soft, wet, and easily breaks"@en , "Anotación sobre el estado
  de árbol descompuesto. El tronco se aplasta o rompe fácilmente y
  está húmedo"@es ;
```

```
  rdfs:isDefinedBy sta: .

sta:StumpAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:DeadTreeAnnotation ;
  rdfs:label "Stump annotation"@en , "Anotación de tocón"@es ;
  rdfs:comment "Annotation about stump status"@en , "Anotación
  sobre el estado tocón"@es ;
  rdfs:isDefinedBy sta: .


# anotaciones estado tronco caído
sta:LogStatusAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Log status annotation"@en , "Anotación del estado
  de un tronco caído"@es ;
  rdfs:comment "Annotation about log status"@en , "Anotación
  sobre el estado de un tronco caído"@es ;
  rdfs:isDefinedBy sta: .

sta:HardLogAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:LogStatusAnnotation ;
  rdfs:label "Hard log annotation"@en , "Anotación de tronco
  caído duro"@es ;
  rdfs:comment "Annotation about hard log status"@en ,
  "Anotación sobre el estado de tronco caído duro"@es ;
  rdfs:isDefinedBy sta: .

sta:MildLogAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:LogStatusAnnotation ;
  rdfs:label "Mild log annotation"@en , "Anotación de tronco
  caído de dureza media"@es ;
  rdfs:comment "Annotation about mild log status"@en ,
  "Anotación sobre el estado de tronco caído de dureza media"@es ;
  rdfs:isDefinedBy sta: .

sta:SoftLogAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:LogStatusAnnotation ;
  rdfs:label "Soft log annotation"@en , "Anotación de tronco caído
  blando"@es ;
  rdfs:comment "Annotation about soft log status"@en , "Anotación
  sobre el estado de tronco caído blando"@es ;
  rdfs:isDefinedBy sta: .

sta:BuriedLogAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:LogStatusAnnotation ;
  rdfs:label "Buried log annotation"@en , "Anotación de tronco
  caído enterrado"@es ;
  rdfs:comment "Annotation about buried log status"@en , "Anotación
  sobre el estado de tronco caído enterrado"@es ;
```

```
  rdfs:isDefinedBy sta: .

sta:IntegratedLogAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:LogStatusAnnotation ;
  rdfs:label "Integrated log annotation"@en , "Anotación de tronco
  caído integrado"@es ;
  rdfs:comment "Annotation about integrated log status"@en ,
  "Anotación sobre el estado de tronco caído integrado"@es ;
  rdfs:isDefinedBy sta: .


# anotaciones tipo microhábitat
sta:MicrohabitatTypeAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:Annotation ;
  rdfs:label "Microhabitat type annotation"@en , "Anotación de
  tipo de microhábitat"@es ;
  rdfs:comment "Annotation about microhabitat type"@en ,
  "Anotación sobre el tipo de microhábitat"@es ;
  rdfs:isDefinedBy sta: .

sta:CavityAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Cavity annotation"@en , "Anotación de cavidad"@es ;
  rdfs:comment "Annotation about a microhabitat of type cavity"@en ,
  "Anotación de un microhábitat de tipo cavidad"@es ;
  rdfs:isDefinedBy sta: .

sta:BirdCavityAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:CavityAnnotation ;
  rdfs:label "Bird cavity annotation"@en , "Anotación de cavidad de
  pájaro"@es ;
  rdfs:comment "Annotation about a microhabitat of type bird cavity"@en ,
  "Anotación de un microhábitat de tipo cavidad de pájaro"@es ;
  rdfs:isDefinedBy sta: .

sta:InsectGalleryAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:CavityAnnotation ;
  rdfs:label "Insect gallery annotation"@en , "Anotación de galería
  de insectos"@es ;
  rdfs:comment "Annotation about a microhabitat of type insect gallery
  cavity"@en , "Anotación de un microhábitat de tipo cavidad de galería
  de insectos"@es ;
  rdfs:isDefinedBy sta: .

sta:IrregularCavityAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:CavityAnnotation ;
  rdfs:label "Irregular cavity annotation"@en , "Anotación de galería
  de insectos"@es ;
  rdfs:comment "Annotation about a microhabitat of type irregular
```

```
  cavity"@en , "Anotación de un microhábitat de tipo cavidad irregular"@es ;
  rdfs:isDefinedBy sta: .

sta:TreeInjuryAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Tree injury annotation"@en , "Anotación de lesión de
  árbol"@es ;
  rdfs:comment "Annotation about a microhabitat of type tree injury"@en ,
  "Anotación de un microhábitat de tipo lesión de árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:CrownDeadwoodAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Crown deadwood annotation"@en , "Anotación de
  madera muerta en copa"@es ;
  rdfs:comment "Annotation about a microhabitat of type crown
  deadwood"@en , "Anotación de un microhábitat de tipo madera muerta
  en copa"@es ;
  rdfs:isDefinedBy sta: .

sta:ExcrescenceAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Excrescence annotation"@en , "Anotación de excrecencia"@es ;
  rdfs:comment "Annotation about a microhabitat of type excrescence, e.g.
  twig tangles, burrs and cankers"@en , "Anotación de un microhábitat de
  tipo excrecencia, ej. aglomeraciones de ramas y cancros"@es ;
  rdfs:isDefinedBy sta: .

sta:RootExcrescenceAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:ExcrescenceAnnotation ;
  rdfs:label "Root excrescence annotation"@en , "Anotación de excrecencia
  en raíz"@es ;
  rdfs:comment "Annotation about a microhabitat of type root
  excrescence"@en , "Anotación de un microhábitat de tipo excrecencia
  en raíz"@es ;
  rdfs:isDefinedBy sta: .

sta:StemExcrescenceAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:ExcrescenceAnnotation ;
  rdfs:label "Stem excrescence annotation"@en , "Anotación de excrecencia
  en tronco"@es ;
  rdfs:comment "Annotation about a microhabitat of type stem
  excrescence"@en , "Anotación de un microhábitat de tipo excrecencia
  en el tronco"@es ;
  rdfs:isDefinedBy sta: .

sta:BranchExcrescenceAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:ExcrescenceAnnotation ;
  rdfs:label "Branch excrescence annotation"@en , "Anotación de
```

```
  excrecencia en rama"@es ;
  rdfs:comment "Annotation about a microhabitat of type branch
  excrescence"@en , "Anotación de un microhábitat de tipo excrecencia
  en rama"@es ;
  rdfs:isDefinedBy sta: .

sta:ExudateAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Exudate annotation"@en , "Anotación de exudado"@es ;
  rdfs:comment "Annotation about a microhabitat of type exudate"@en ,
  "Anotación de un microhábitat de tipo exudado"@es ;
  rdfs:isDefinedBy sta: .

sta:NestAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Nest annotation"@en , "Anotación de nido"@es ;
  rdfs:comment "Annotation about a microhabitat of type nest"@en ,
  "Anotación de un microhábitat de tipo nido"@es ;
  rdfs:isDefinedBy sta: .

sta:BirdNestAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:NestAnnotation ;
  rdfs:label "Bird nest annotation"@en , "Anotación de nido de ave"@es ;
  rdfs:comment "Annotation about a microhabitat of type bird nest"@en ,
  "Anotación de un microhábitat de tipo nido de ave"@es ;
  rdfs:isDefinedBy sta: .

sta:InsectNestAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:NestAnnotation ;
  rdfs:label "Insect nest annotation"@en , "Anotación de nido de
  insectos"@es ;
  rdfs:comment "Annotation about a microhabitat of type insect nest"@en ,
  "Anotación de un microhábitat de tipo nido de insectos"@es ;
  rdfs:isDefinedBy sta: .

sta:EpiphyticAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:MicrohabitatTypeAnnotation ;
  rdfs:label "Epiphytic annotation"@en , "Anotación de epífitos"@es ;
  rdfs:comment "Annotation about a microhabitat of type epiphytic, e.g.
  fungi, mosses, lichens, climbing plants, ferns, mistletoe"@en , "Anotación
  de un microhábitat de tipo epífitos, ej. hongos, musgos, líquenes, plantas
  trepadoras, helechos, muérdago"@es ;
  rdfs:isDefinedBy sta: .

sta:FungiAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:EpiphyticAnnotation ;
  rdfs:label "Fungi annotation"@en , "Anotación de hongos"@es ;
  rdfs:comment "Annotation about a microhabitat of type fungi"@en ,
  "Anotación de un microhábitat de tipo hongos"@es ;
```

```
  rdfs:isDefinedBy sta: .

sta:MossAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:EpiphyticAnnotation ;
  rdfs:label "Moss annotation"@en , "Anotación de musgo"@es ;
  rdfs:comment "Annotation about a microhabitat of type moss"@en ,
  "Anotación de un microhábitat de tipo musgo"@es ;
  rdfs:isDefinedBy sta: .

sta:LichenAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:EpiphyticAnnotation ;
  rdfs:label "Lichen annotation"@en , "Anotación de liquen"@es ;
  rdfs:comment "Annotation about a microhabitat of type lichen"@en ,
  "Anotación de un microhábitat de tipo liquen"@es ;
  rdfs:isDefinedBy sta: .

sta:ClimbingPlantAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:EpiphyticAnnotation ;
  rdfs:label "Climbing plant annotation"@en , "Anotación de planta
  trepadora"@es ;
  rdfs:comment "Annotation about a microhabitat of type climbing
  plant"@en , "Anotación de un microhábitat de tipo planta trepadora"@es ;
  rdfs:isDefinedBy sta: .

sta:FernAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:EpiphyticAnnotation ;
  rdfs:label "Fern annotation"@en , "Anotación de helecho"@es ;
  rdfs:comment "Annotation about a microhabitat of type fern"@en ,
  "Anotación de un microhábitat de tipo helecho"@es ;
  rdfs:isDefinedBy sta: .

sta:MistletoeAnnotation a owl:Class, rdfs:Class ;
  rdfs:subClassOf sta:EpiphyticAnnotation ;
  rdfs:label "Mistletoe annotation"@en , "Anotación de muérdago"@es ;
  rdfs:comment "Annotation about a microhabitat of type mistletoe"@en ,
  "Anotación de un microhábitat de tipo muérdago"@es ;
  rdfs:isDefinedBy sta: .

# fotos de partes de una planta y microhábitats
sta:PlantPartPhoto a owl:Class , rdfs:Class ;
  rdfs:label "Plant part photo"@en , "Parte de la planta de la foto"@es ;
  rdfs:comment "Part of a plant that it is shown in the image"@en , "Parte
  de la planta que se muestra en la imagen"@es ;
  rdfs:isDefinedBy sta: .

sta:LeafPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Leaf photo"@en , "Foto de una hoja"@es ;
  rdfs:isDefinedBy sta: .
```

```
sta:FruitPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Fruit photo"@en , "Foto de un fruto"@es ;
  rdfs:comment "Photo of a fruit which grows on a tree"@en , "Foto de
  un fruto que crece en un árbol"@es ;
  rdfs:isDefinedBy sta: .

sta:FlowerPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Flower photo"@en , "Foto de una flor"@es ;
  rdfs:isDefinedBy sta: .

sta:StemPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Stem photo"@en , "Foto de un tronco"@es ;
  rdfs:isDefinedBy sta: .

sta:BranchPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Branch photo"@en , "Foto de una rama"@es ;
  rdfs:isDefinedBy sta: .

sta:CrownPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Crown photo"@en , "Foto de una copa"@es ;
  rdfs:isDefinedBy sta: .

sta:RootPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Root photo"@en , "Foto de una raíz"@es ;
  rdfs:isDefinedBy sta: .

sta:StumpPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Stump photo"@en , "Foto de un tocón"@es ;
  rdfs:isDefinedBy sta: .

sta:LogPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Log photo"@en , "Foto de un tronco caído"@es ;
  rdfs:isDefinedBy sta: .

sta:MicrohabitatPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Microhabitat photo"@en , "Foto de un microhábitat"@es ;
  rdfs:isDefinedBy sta: .

sta:GeneralViewPhoto a owl:Class , rdfs:Class ;
```

```
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "General view photo"@en , "Foto general"@es ;
  rdfs:isDefinedBy sta: .

sta:OtherPartPhoto a owl:Class , rdfs:Class ;
  rdfs:subClassOf sta:PlantPartPhoto ;
  rdfs:label "Other part photo"@en , "Foto de otra parte"@es ;
  rdfs:isDefinedBy sta: .


###  http://crossforest.eu/ifn/ontology/Taxon
ifn:Taxon rdf:type owl:Class, rdfs:Class ;
  rdfs:label "Taxon"@en ;
  rdfs:label "Taxón"@es .

###  http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing
w3cgeo:SpatialThing rdf:type owl:Class, rdfs:Class ;
  rdfs:label "Spatial thing"@en ;
  rdfs:label "Cosa en el espacio"@es .

###  http://xmlns.com/foaf/0.1/Image
foaf:Image rdf:type owl:Class, rdfs:Class ;
  rdfs:label "Image"@en ;
  rdfs:label "Imagen"@es .


#################################################################
#    Object Properties
#################################################################

sta:isInPlant a owl:ObjectProperty , rdf:Property ;
  rdfs:label "is in plant"@en , "está en la planta"@es ;
  rdfs:comment "microhabitat is in plant"@en , "microhábitat está en
  la planta"@es ;
  rdfs:domain sta:Microhabitat ;
  rdfs:range sta:Plant ;
  rdfs:isDefinedBy sta: .

sta:hasAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:label "has annotation"@en , "tiene anotación"@es ;
  rdfs:comment "annotation of a tree"@en , "anotación de un árbol"@es ;
  rdfs:domain sta:SpatialEntity ;
  rdfs:range sta:Annotation ;
  rdfs:isDefinedBy sta: .

sta:hasImageAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has image annotation"@en , "tiene anotación de imagen"@es ;
  rdfs:comment "annotation which includes a image"@en , "anotación que
```

```
  incluye una imagen"@es ;
  rdfs:domain sta:SpatialEntity ;
  rdfs:range sta:ImageAnnotation ;
  rdfs:isDefinedBy sta: .

sta:hasImage a owl:ObjectProperty , rdf:Property ;
  rdfs:label "has image"@en , "tiene imagen"@es ;
  rdfs:comment "image of an image annotation"@en , "imagen de una
  anotación de imagen"@es ;
  rdfs:domain sta:ImageAnnotation ;
  rdfs:range sta:Image ;
  rdfs:isDefinedBy sta: .

sta:hasSpeciesAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has species annotation"@en , "tiene anotación de especie"@es ;
  rdfs:comment "annotation of plant species"@en , "anotación de una
  especie de una planta"@es ;
  rdfs:domain sta:Plant ;
  rdfs:range sta:SpeciesAnnotation ;
  rdfs:isDefinedBy sta: .

#Propiedad más general que especie (no es subpropiedad del ifn:hasTaxon
porque tiene distinto dominio)
sta:hasTaxon a owl:ObjectProperty , rdf:Property ;
  rdfs:label "has taxon"@en , "tiene taxón"@es ;
  rdfs:domain sta:SpeciesAnnotation ;
  rdfs:range ifn:Taxon ;
  rdfs:isDefinedBy sta: .

sta:hasPositionAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has position annotation"@en , "tiene anotación de posición"@es ;
  rdfs:comment "annotation of a spatial entity position"@en , "anotación de
  una posición de una entidad espacial"@es ;
  rdfs:domain sta:SpatialEntity ;
  rdfs:range sta:PositionAnnotation ;
  rdfs:isDefinedBy sta: .

sta:hasDiameterAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has diameter annotation"@en , "tiene anotación de
  diámetro"@es ;
  rdfs:comment "annotation of plant diameter"@en , "anotación de
  diámetro de una planta"@es ;
  rdfs:domain sta:Plant ;
  rdfs:range sta:DiameterAnnotation ;
  rdfs:isDefinedBy sta: .
```

```
sta:hasHeightAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has height annotation"@en , "tiene anotación de altura"@es ;
  rdfs:comment "annotation of plant diameter"@en , "anotación de altura
  de una planta"@es ;
  rdfs:domain sta:Plant ;
  rdfs:range sta:HeightAnnotation ;
  rdfs:isDefinedBy sta: .

sta:hasTreeStatusAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has tree status annotation"@en , "tiene anotación de estado
  de árbol"@es ;
  rdfs:comment "annotation of tree status"@en , "anotación de estado
  de árbol"@es ;
  rdfs:domain sta:Tree ;
  rdfs:range sta:TreeStatusAnnotation ;
  rdfs:isDefinedBy sta: .

sta:hasLogStatusAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has log status annotation"@en , "tiene anotación de estado
  de tronco caído"@es ;
  rdfs:comment "annotation of log status"@en , "anotación de estado
  de tronco caído"@es ;
  rdfs:domain sta:Log ;
  rdfs:range sta:LogStatusAnnotation ;
  rdfs:isDefinedBy sta: .

sta:hasMicrohabitatTypeAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has microhabitat type annotation"@en , "tiene anotación de
  tipo de microhábitat"@es ;
  rdfs:comment "annotation of microhabitat type"@en , "anotación de
  tipo de microhábitat"@es ;
  rdfs:domain sta:Microhabitat ;
  rdfs:range sta:MicrohabitatTypeAnnotation ;
  rdfs:isDefinedBy sta: .


# anotaciones primarias
sta:hasPrimaryAnnotation a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasAnnotation ;
  rdfs:label "has primary annotation"@en , "tiene anotación primaria"@es ;
  rdfs:comment "primary annotation of a spatial entity"@en , "anotación
  primaria de una entidad espacial"@es ;
  rdfs:domain sta:SpatialEntity ;
  rdfs:range sta:Annotation ;
  rdfs:isDefinedBy sta: .
```

```
# Propiedad proxy, es la posición que se considera buena para una
# entidad espacial en cada momento
sta:hasPrimaryPosition a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary position annotation"@en , "tiene anotación
  primaria de posición"@es ;
  rdfs:comment "primary position annotation of a spatial entity"@en ,
  "anotación primaria de posición de una entidad espacial"@es ;
  rdfs:domain sta:SpatialEntity ;
  rdfs:range sta:PositionAnnotation ;
  rdfs:isDefinedBy sta: .

# Propiedad proxy, es el diámetro que se considera bueno para una
# planta en cada momento
sta:hasPrimaryDiameter a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary diameter annotation"@en , "tiene anotación
  primaria de diámetro"@es ;
  rdfs:comment "primary diameter annotation of a plant"@en , "anotación
  primaria de diámetro de una planta"@es ;
  rdfs:domain sta:Plant ;
  rdfs:range sta:DiameterAnnotation ;
  rdfs:isDefinedBy sta: .

# Propiedad proxy, es la altura que se considera buena para una
# planta en cada momento
sta:hasPrimaryHeight a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary height annotation"@en , "tiene anotación primaria
  de altura"@es ;
  rdfs:comment "primary height annotation of a plant"@en , "anotación
  primaria de altura de una planta"@es ;
  rdfs:domain sta:Plant ;
  rdfs:range sta:HeightAnnotation ;
  rdfs:isDefinedBy sta: .

# Propiedad proxy, es la especie que se considera buena para una
# planta en cada momento
sta:hasPrimarySpecies a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary species annotation"@en , "tiene anotación primaria
  de especie"@es ;
  rdfs:comment "primary species annotation of a plant"@en , "anotación
  primaria de una especie de una planta"@es ;
  rdfs:domain sta:Plant ;
  rdfs:range sta:SpeciesAnnotation ;
  rdfs:isDefinedBy sta: .
```

```
# Propiedad proxy, es el estado de árbol que se considera bueno
# para un árbol en cada momento
sta:hasPrimaryTreeStatus a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary tree status annotation"@en , "tiene anotación
  primaria de estado de árbol"@es ;
  rdfs:comment "primary tree status annotation of a tree"@en , "anotación
  primaria de estado de árbol"@es ;
  rdfs:domain sta:Tree ;
  rdfs:range sta:TreeStatusAnnotation ;
  rdfs:isDefinedBy sta: .

# Propiedad proxy, es el estado de tronco caído que se considera bueno
# para un tronco caído en cada momento
sta:hasPrimaryLogStatus a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary log status annotation"@en , "tiene anotación
  primaria de estado de tronco caído"@es ;
  rdfs:comment "primary log status annotation of a log"@en , "anotación
  primaria de estado de tronco caído"@es ;
  rdfs:domain sta:Log ;
  rdfs:range sta:LogStatusAnnotation ;
  rdfs:isDefinedBy sta: .

# Propiedad proxy, es la especie que se considera buena para
# un tronco caído en cada momento
sta:hasPrimaryMicrohabitatType a owl:ObjectProperty , rdf:Property ;
  rdfs:subPropertyOf sta:hasPrimaryAnnotation ;
  rdfs:label "has primary microhabitat type annotation"@en , "tiene
  anotación primaria de tipo de microhábitat"@es ;
  rdfs:comment "primary microhabitat type annotation of a log"@en ,
  "anotación primaria de tipo de microhábitat"@es ;
  rdfs:domain sta:Microhabitat ;
  rdfs:range sta:MicrohabitatTypeAnnotation ;
  rdfs:isDefinedBy sta: .



######################################################################
#    Data Properties
######################################################################

sta:hasDiameterInMillimeters rdf:type owl:DatatypeProperty ;
  rdfs:domain sta:DiameterAnnotation ;
  rdfs:range xsd:decimal ;
  rdfs:label "has diameter in millimeters"@en ;
  rdfs:label "tiene diámetro en milímetros"@es ;
```

```
  rdfs:isDefinedBy sta: .

sta:hasHeightInMeters rdf:type owl:DatatypeProperty ;
  rdfs:domain sta:HeightAnnotation ;
  rdfs:range xsd:decimal ;
  rdfs:label "has height in meters"@en ;
  rdfs:label "tiene altura en metros"@es ;
  rdfs:isDefinedBy sta: .

# no es una subpropiedad de w3cgeo:lat porque los dominios no
# son compatibles
sta:lat rdf:type owl:DatatypeProperty ;
  rdfs:domain sta:PositionAnnotation ;
  rdfs:range xsd:float ;
  rdfs:label "latitude"@en, "latitud"@es ;
  rdfs:comment "WGS84 latitude of a PositionAnnotation (decimal
  degrees)"@en, "Latitud WGS84 de una PositionAnnotation (grados
  decimales)"@es ;
  rdfs:isDefinedBy sta: .

# no es una subpropiedad de w3cgeo:long porque los dominios no
# son compatibles
sta:long rdf:type owl:DatatypeProperty ;
  rdfs:domain sta:PositionAnnotation ;
  rdfs:range xsd:float ;
  rdfs:label "longitude"@en, "longitud"@es ;
  rdfs:comment "WGS84 longitude of a PositionAnnotation (decimal
  degrees)"@en, "Longitud WGS84 de una PositionAnnotation (grados
  decimales)"@es ;
  rdfs:isDefinedBy sta: .

###  http://www.w3.org/2003/01/geo/wgs84_pos#lat
w3cgeo:lat rdf:type owl:DatatypeProperty ;
  rdfs:domain w3cgeo:SpatialThing ;
  rdfs:range xsd:float ;
  rdfs:label "latitude"@en ;
  rdfs:label "latitud"@es ;
  rdfs:comment "The WGS84 latitude of a SpatialThing (decimal
  degrees)"@en .

###  http://www.w3.org/2003/01/geo/wgs84_pos#long
w3cgeo:long rdf:type owl:DatatypeProperty ;
  rdfs:domain w3cgeo:SpatialThing ;
  rdfs:range xsd:float ;
  rdfs:label "longitude"@en ;
  rdfs:label "longitud"@es ;
  rdfs:comment "The WGS84 longitude of a SpatialThing (decimal
  degrees)"@en .
```

```
#################################################################
#    Annotation properties
#################################################################

###  http://purl.org/dc/terms/created
dc:created rdf:type owl:AnnotationProperty ;
rdfs:label "created"@en ;
rdfs:label "creado"@es .

###  http://purl.org/dc/terms/creator
dc:creator rdf:type owl:AnnotationProperty ;
rdfs:label "creator"@en ;
rdfs:label "creador"@es .
```

Apéndice C

Fichero de configuración CRAFTS API

```
{
    "apiId": "educawood",
    "endpoints": [
        {
            "id": "crossforest",
            "sparqlURI": "https://crossforest.gsic.uva.es/sparql/",
            "graphURI": "http://crossforest.eu",
            "httpMethod": "GET"
        },
        {
            "id": "educawood",
            "sparqlURI": "https://crossforest.gsic.uva.es/pruebas/sparql",
            "graphURI": "http://educawood.gsic.uva.es",
            "httpMethod": "GET",
            "sparqlUpdate": {
                "sparqlURI": "https://crossforest.gsic.uva.es/pruebas/sparql-auth",
                "id": "educawood_update",
                "graphURI": "http://educawood.gsic.uva.es",
                "httpMethod": "GET"
            }
        },
        {
            "id": "dbpedia",
            "sparqlURI": "http://dbpedia.org/sparql",
            "graphURI": "http://dbpedia.org",
            "httpMethod": "GET"
        }
    ],
    "model": [
        {
            "id": "Tree",
            "oprops": [],
            "dprops": [
```

```
            {
                "label": "latWGS84",
                "iri": "http://crossforest.eu/position/ontology/hasPosition>/
                <http://epsg.w3id.org/ontology/axis/1",
                "endpoint": "crossforest"
            },
            {
                "label": "lngWGS84",
                "iri": "http://crossforest.eu/position/ontology/hasPosition>/
                <http://epsg.w3id.org/ontology/axis/2",
                "endpoint": "crossforest"
            },
            {
                "label": "dbh1mm",
                "iri": "https://datos.iepnb.es/def/sector-publico/medio-
                ambiente/ifn/hasDBH1InMillimeters",
                "endpoint": "crossforest"
            },
            {
                "label": "dbh2mm",
                "iri": "https://datos.iepnb.es/def/sector-publico/medio-
                ambiente/ifn/hasDBH2InMillimeters",
                "endpoint": "crossforest"
            },
            {
                "label": "heightM",
                "iri": "https://datos.iepnb.es/def/sector-publico/medio-
                ambiente/ifn/hasTotalHeightInMeters",
                "endpoint": "crossforest"
            }
        ],
        "types": [
            {
                "label": "species",
                "targetId": "Species",
                "restrictions": [
                    "?type a/rdfs:subClassOf
                    <https://datos.iepnb.es/def/sector-publico/medio-ambiente/
                    ifn/Taxon> ."
                ],
                "embed": true,
                "endpoint": "crossforest"
            }
        ]
    },
    {
        "id": "Patch",
        "oprops": [
            {
```

```
            "label": "polygon",
            "targetId": "Polygon",
            "iri": "http://crossforest.eu/position/ontology/hasPolygon",
            "embed": true,
            "endpoint": "crossforest"
        },
        {
            "label": "infoSpecies",
            "targetId": "InfoPatchSpecies",
            "iri": "https://datos.iepnb.es/def/sector-publico/medio-
            ambiente/mfe/containsSpecies",
            "restrictions": [
                "?value <https://datos.iepnb.es/def/sector-publico/medio-
                ambiente/mfe/hasPercentageOfSpecies> ?spperc .\nFILTER
                (?spperc > 0.4) ."
            ],
            "embed": true,
            "endpoint": "crossforest"
        }
    ],
    "dprops": [
        {
            "label": "province",
            "iri": "http://vocab.linkeddata.es/datosabiertos/def/sector-
            publico/territorio#provincia>/<http://www.w3.org/2000/01/
            rdf-schema#label",
            "endpoint": "crossforest"
        },
        {
            "label": "canopyCoverTotalPercent",
            "iri": "https://datos.iepnb.es/def/sector-publico/medio-
            ambiente/mfe/hasCanopyCoverTotalPercent",
            "endpoint": "crossforest"
        },
        {
            "label": "canopyCoverTreesPercent",
            "iri": "https://datos.iepnb.es/def/sector-publico/medio-
            ambiente/mfe/hasCanopyCoverTreesPercent",
            "endpoint": "crossforest"
        }
    ],
    "types": [
        {
            "label": "soilUse",
            "targetId": "SoilUse",
            "embed": true,
            "restrictions": [
                "?type rdfs:subClassOf* <https://datos.iepnb.es/def/sector-
                publico/medio-ambiente/ifn/Use> ."
```

```
                ],
                "endpoint": "crossforest"
            }
        ]
    },
    {
        "id": "Polygon",
        "oprops": [
            {
                "label": "layer",
                "iri": "http://crossforest.eu/position/ontology/isInLayer",
                "endpoint": "crossforest"
            }
        ],
        "dprops": [
            {
                "label": "westBoundWGS84",
                "iri": "http://epsg.w3id.org/ontology/hasLeftBound107",
                "endpoint": "crossforest"
            },
            {
                "label": "eastBoundWGS84",
                "iri": "http://epsg.w3id.org/ontology/hasRightBound107",
                "endpoint": "crossforest"
            },
            {
                "label": "northBoundWGS84",
                "iri": "http://epsg.w3id.org/ontology/hasUpperBound106",
                "endpoint": "crossforest"
            },
            {
                "label": "southBoundWGS84",
                "iri": "http://epsg.w3id.org/ontology/hasLowerBound106",
                "endpoint": "crossforest"
            },
            {
                "label": "areaM2",
                "iri": "http://crossforest.eu/position/ontology/
                hasAreaInSquareMeters",
                "endpoint": "crossforest"
            },
            {
                "label": "wkt",
                "iri": "http://www.opengis.net/ont/geosparql#asWKT",
                "endpoint": "crossforest"
            }
        ],
        "types": []
    },
```

```
{
    "id": "SoilUse",
    "oprops": [
        {
            "label": "subClassOf",
            "targetId": "SoilUse",
            "iri": "http://www.w3.org/2000/01/rdf-schema#subClassOf",
            "endpoint": "crossforest"
        }
    ],
    "dprops": [
        {
            "label": "label",
            "iri": "http://www.w3.org/2000/01/rdf-schema#label",
            "endpoint": "crossforest"
        },
        {
            "label": "comment",
            "iri": "http://www.w3.org/2000/01/rdf-schema#comment",
            "endpoint": "crossforest"
        }
    ],
    "types": []
},
{
    "id": "InfoPatchSpecies",
    "oprops": [
        {
            "label": "species",
            "targetId": "Species",
            "iri": "https://datos.iepnb.es/def/sector-publico/medio-
            ambiente/mfe/hasSpecies",
            "embed": true,
            "endpoint": "crossforest"
        }
    ],
    "dprops": [
        {
            "label": "percOccupation",
            "iri": "https://datos.iepnb.es/def/sector-publico/medio-
            ambiente/mfe/hasPercentageOfSpecies",
            "endpoint": "crossforest"
        }
    ],
    "types": []
},
{
    "id": "EducaTree",
    "oprops": [
```

```
{
    "label": "creator",
    "iri": "http://purl.org/dc/terms/creator",
    "endpoint": "educawood"
},
{
    "label": "position",
    "targetId": "PositionAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasPrimaryPosition",
    "embed": true,
    "endpoint": "educawood"
},
{
    "label": "positionAnnotations",
    "targetId": "PositionAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasPositionAnnotation",
    "embed": false,
    "endpoint": "educawood"
},
{
    "label": "species",
    "targetId": "SpeciesAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasPrimarySpecies",
    "embed": true,
    "endpoint": "educawood"
},
{
    "label": "speciesAnnotations",
    "targetId": "SpeciesAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasSpeciesAnnotation",
    "embed": false,
    "endpoint": "educawood"
},
{
    "label": "diameter",
    "targetId": "DiameterAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasPrimaryDiameter",
    "embed": true,
    "endpoint": "educawood"
},
{
    "label": "diameterAnnotations",
    "targetId": "DiameterAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
```

```
    hasDiameterAnnotation",
    "embed": false,
    "endpoint": "educawood"
},
{
    "label": "height",
    "targetId": "HeightAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasPrimaryHeight",
    "embed": true,
    "endpoint": "educawood"
},
{
    "label": "heightAnnotations",
    "targetId": "HeightAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasHeightAnnotation",
    "embed": false,
    "endpoint": "educawood"
},
{
    "label": "treeStatus",
    "targetId": "TreeStatusAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasPrimaryTreeStatus",
    "embed": true,
    "endpoint": "educawood"
},
{
    "label": "treeStatusAnnotations",
    "targetId": "TreeStatusAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasTreeStatusAnnotation",
    "embed": false,
    "endpoint": "educawood"
},
{
    "label": "imageAnnotations",
    "targetId": "ImageAnnotation",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/
    hasImageAnnotation",
    "embed": true,
    "endpoint": "educawood"
},
{
    "label": "hasMicrohabitats",
    "targetId": "EducaMicrohabitat",
    "iri": "http://educawood.gsic.uva.es/sta/ontology/isInPlant",
    "embed": false,
```

```
                "inv": true,
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "types",
                "endpoint": "educawood",
                "writeonly": true
            }
        ]
    },
    {
        "id": "PositionAnnotation",
        "oprops": [
            {
                "label": "creator",
                "iri": "http://purl.org/dc/terms/creator",
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            },
            {
                "label": "latWGS84",
                "iri": "http://www.w3.org/2003/01/geo/wgs84_pos#lat",
                "endpoint": "educawood"
            },
            {
                "label": "lngWGS84",
                "iri": "http://www.w3.org/2003/01/geo/wgs84_pos#long",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "types",
                "endpoint": "educawood",
```

```
            "writeonly": true
        }
    ]
},
{
    "id": "DiameterAnnotation",
    "oprops": [
        {
            "label": "creator",
            "iri": "http://purl.org/dc/terms/creator",
            "endpoint": "educawood"
        }
    ],
    "dprops": [
        {
            "label": "created",
            "iri": "http://purl.org/dc/terms/created",
            "endpoint": "educawood"
        },
        {
            "label": "millimeters",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/
            hasDiameterInMillimeters",
            "endpoint": "educawood"
        }
    ],
    "types": [
        {
            "label": "types",
            "endpoint": "educawood",
            "writeonly": true
        }
    ]
},
{
    "id": "HeightAnnotation",
    "oprops": [
        {
            "label": "creator",
            "iri": "http://purl.org/dc/terms/creator",
            "endpoint": "educawood"
        }
    ],
    "dprops": [
        {
            "label": "created",
            "iri": "http://purl.org/dc/terms/created",
            "endpoint": "educawood"
        },
```

```
            {
                "label": "meters",
                "iri": "http://educawood.gsic.uva.es/sta/ontology/
                hasHeightInMeters",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "types",
                "endpoint": "educawood",
                "writeonly": true
            }
        ]
    },
    {
        "id": "TreeStatusAnnotation",
        "oprops": [
            {
                "label": "creator",
                "iri": "http://purl.org/dc/terms/creator",
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "treeStatus",
                "endpoint": "educawood"
            }
        ]
    },
    {
        "id": "ImageAnnotation",
        "oprops": [
            {
                "label": "creator",
                "iri": "http://purl.org/dc/terms/creator",
                "endpoint": "educawood"
            },
            {
                "label": "image",
                "targetId": "Image",
```

```
            "iri": "http://educawood.gsic.uva.es/sta/ontology/hasImage",
            "embed": true,
            "endpoint": "educawood"
        }
    ],
    "dprops": [
        {
            "label": "created",
            "iri": "http://purl.org/dc/terms/created",
            "endpoint": "educawood"
        }
    ],
    "types": [
        {
            "label": "types",
            "endpoint": "educawood",
            "writeonly": true
        }
    ]
},
{
    "id": "Image",
    "oprops": [],
    "dprops": [],
    "types": [
        {
            "label": "plantPart",
            "restrictions": [
                "FILTER (?type NOT IN (
                <http://educawood.gsic.uva.es/sta/ontology/Image> ))"
            ],
            "endpoint": "educawood"
        },
        {
            "label": "types",
            "endpoint": "educawood",
            "writeonly": true
        }
    ]
},
{
    "id": "SpeciesAnnotation",
    "oprops": [
        {
            "label": "creator",
            "iri": "http://purl.org/dc/terms/creator",
            "endpoint": "educawood"
        },
        {
```

```
                "label": "species",
                "targetId": "Species",
                "iri": "http://educawood.gsic.uva.es/sta/ontology/hasTaxon",
                "embed": true,
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "types",
                "endpoint": "educawood",
                "writeonly": true
            }
        ]
    },
    {
        "id": "Species",
        "oprops": [
            {
                "label": "dbpedia",
                "targetId": "DbpediaSpecies",
                "iri": "http://schema.org/sameAs",
                "restrictions": [
                    "FILTER contains(str(?value), \"dbpedia\" )"
                ],
                "embed": true,
                "endpoint": "crossforest"
            }
        ],
        "dprops": [
            {
                "label": "scientificName",
                "iri": "https://datos.iepnb.es/def/sector-publico/medio-
                ambiente/ifn/hasAcceptedName>/<https://datos.iepnb.es/def/
                sector-publico/medio-ambiente/ifn/name",
                "endpoint": "crossforest"
            },
            {
                "label": "vulgarName",
                "iri": "https://datos.iepnb.es/def/sector-publico/medio-
                ambiente/ifn/vulgarName",
                "endpoint": "crossforest"
```

```
            }
        ],
        "types": [
            {
                "label": "types",
                "restrictions": [
                    "FILTER (?type NOT IN (
                    <http://www.w3.org/2002/07/owl#Class>,
                    <http://www.w3.org/2000/01/rdf-schema#Class> ))"
                ],
                "endpoint": "crossforest"
            }
        ]
    },
    {
        "id": "DbpediaSpecies",
        "oprops": [
            {
                "label": "image",
                "iri": "http://dbpedia.org/ontology/thumbnail",
                "endpoint": "dbpedia"
            }
        ],
        "dprops": [
            {
                "label": "comment",
                "iri": "http://www.w3.org/2000/01/rdf-schema#comment",
                "restrictions": [
                    "FILTER (lang(?value) = \"es\" OR lang(?value) = \"en\" OR
                    lang(?value) = \"\")"
                ],
                "endpoint": "dbpedia"
            }
        ],
        "types": []
    },
    {
        "id": "EducaLog",
        "oprops": [
            {
                "label": "creator",
                "iri": "http://purl.org/dc/terms/creator",
                "endpoint": "educawood"
            },
            {
                "label": "position",
                "targetId": "PositionAnnotation",
                "iri": "http://educawood.gsic.uva.es/sta/ontology/
                hasPrimaryPosition",
```

```
        "embed": true,
        "endpoint": "educawood"
    },
    {
        "label": "positionAnnotations",
        "targetId": "PositionAnnotation",
        "iri": "http://educawood.gsic.uva.es/sta/ontology/
        hasPositionAnnotation",
        "embed": false,
        "endpoint": "educawood"
    },
    {
        "label": "species",
        "targetId": "SpeciesAnnotation",
        "iri": "http://educawood.gsic.uva.es/sta/ontology/
        hasPrimarySpecies",
        "embed": true,
        "endpoint": "educawood"
    },
    {
        "label": "speciesAnnotations",
        "targetId": "SpeciesAnnotation",
        "iri": "http://educawood.gsic.uva.es/sta/ontology/
        hasSpeciesAnnotation",
        "embed": false,
        "endpoint": "educawood"
    },
    {
        "label": "logStatus",
        "targetId": "LogStatusAnnotation",
        "iri": "http://educawood.gsic.uva.es/sta/ontology/
        hasPrimaryLogStatus",
        "embed": true,
        "endpoint": "educawood"
    },
    {
        "label": "logStatusAnnotations",
        "targetId": "LogStatusAnnotation",
        "iri": "http://educawood.gsic.uva.es/sta/ontology/
        hasLogStatusAnnotation",
        "embed": false,
        "endpoint": "educawood"
    },
    {
        "label": "hasMicrohabitats",
        "targetId": "EducaMicrohabitat",
        "iri": "http://educawood.gsic.uva.es/sta/ontology/isInPlant",
        "embed": false,
        "inv": true,
```

```
                "endpoint": "educawood"
            },
            {
                "label": "imageAnnotations",
                "targetId": "ImageAnnotation",
                "iri": "http://educawood.gsic.uva.es/sta/ontology/
                hasImageAnnotation",
                "embed": true,
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "types",
                "endpoint": "educawood",
                "writeonly": true
            }
        ]
    },
    {
        "id": "LogStatusAnnotation",
        "oprops": [
            {
                "label": "creator",
                "iri": "http://purl.org/dc/terms/creator",
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "logStatus",
                "endpoint": "educawood"
            }
        ]
    },
```

```
{
    "id": "EducaMicrohabitat",
    "oprops": [
        {
            "label": "creator",
            "iri": "http://purl.org/dc/terms/creator",
            "endpoint": "educawood"
        },
        {
            "label": "inTree",
            "targetId": "EducaTree",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/isInPlant",
            "embed": false,
            "restrictions": [
                "?value a <http://educawood.gsic.uva.es/sta/ontology/Tree> ."
            ],
            "endpoint": "educawood"
        },
        {
            "label": "inLog",
            "targetId": "EducaLog",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/isInPlant",
            "restrictions": [
                "?value a <http://educawood.gsic.uva.es/sta/ontology/Log> ."
            ],
            "embed": false,
            "endpoint": "educawood"
        },
        {
            "label": "microhabitatType",
            "targetId": "MicrohabitatTypeAnnotation",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/
            hasPrimaryMicrohabitatType",
            "embed": true,
            "endpoint": "educawood"
        },
        {
            "label": "microhabitatTypeAnnotations",
            "targetId": "MicrohabitatTypeAnnotation",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/
            hasMicrohabitatTypeAnnotation",
            "embed": false,
            "endpoint": "educawood"
        },
        {
            "label": "imageAnnotations",
            "targetId": "ImageAnnotation",
            "iri": "http://educawood.gsic.uva.es/sta/ontology/
            hasImageAnnotation",
```

```
                "embed": true,
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "types",
                "endpoint": "educawood",
                "writeonly": true
            }
        ]
    },
    {
        "id": "MicrohabitatTypeAnnotation",
        "oprops": [
            {
                "label": "creator",
                "iri": "http://purl.org/dc/terms/creator",
                "endpoint": "educawood"
            }
        ],
        "dprops": [
            {
                "label": "created",
                "iri": "http://purl.org/dc/terms/created",
                "endpoint": "educawood"
            }
        ],
        "types": [
            {
                "label": "microhabitatType",
                "endpoint": "educawood"
            }
        ]
    }
],
"queryTemplates": [
    {
        "id": "indivs",
        "description": "Obtain members (variable \"iri\") of type \"cliri\"
        with an optional offset (parameter \"offset\")",
        "template": "SELECT DISTINCT ?iri\nWHERE {\n?iri a <{{{cliri}}}>
```

```
        .\n}\n{{#limit}}LIMIT {{limit}}{{/limit}}{{^limit}}LIMIT 1000{{/limit}}
        {{#offset}}\nOFFSET {{offset}}{{/offset}}",
        "variables": [
            "iri"
        ],
        "parameters": [
            {
                "label": "cliri",
                "type": "iri",
                "optional": false
            },
            {
                "label": "limit",
                "type": "integer",
                "optional": true
            },
            {
                "label": "offset",
                "type": "integer",
                "optional": true
            }
        ],
        "endpoint": "crossforest"
    },
    {
        "id": "allSpecies",
        "description": "Obtain all the species (variable \"species\") with
        their scientific names (variable \"sciname\"), their Spanish vulgar
        names (variable \"esnames\"), their English vulgar names (variable
        \"ennames\"), and their parent (variable \"parent\") of the forest
        inventory",
        "template": "select ?species ?super ?sciname (group_concat(distinct
        ?esname;separator=\"; \") as ?esnames) (group_concat(distinct ?
        enname;separator=\"; \") as ?ennames) where {\n?species
        <https://datos.iepnb.es/def/sector-publico/medio-
        ambiente/ifn/hasAcceptedName>/<https://datos.iepnb.es/def/sector-
        publico/medio-ambiente/ifn/name> ?sciname ;\n
        <https://datos.iepnb.es/def/sector-publico/medio-
        ambiente/ifn/vulgarName> ?esname, ?enname ;\n  rdfs:subClassOf
        ?super .\n?super rdfs:subClassOf<https://datos.iepnb.es/def/sector-
        publico/medio-ambiente/ifn/Plantae> .\nFILTER (lang(?esname) =
        'es')\nFILTER (lang(?enname) = 'en')\n}\nORDER BY ?sciname",
        "variables": [
            "species",
            "super",
            "sciname",
            "esnames",
            "ennames"
        ],
```

```
    "parameters": [],
    "endpoint": "crossforest"
},
{
    "id": "infoClasses",
    "description": "Obtain info about all the classes (variable \"class\")
    from an ancestor class (parameter \"ancestor\")",
    "template": "select distinct ?class ?parent ?labes ?laben ?comes ?
    comen where {\n ?class rdfs:subClassOf* <{{{ancestor}}}> ;\n
    rdfs:subClassOf ?parent .\n OPTIONAL {\n  ?class rdfs:label ?labes
    .\n  FILTER (lang(?labes) = \"es\")\n }\n OPTIONAL {\n  ?class
    rdfs:label ?laben .\n  FILTER (lang(?laben) = \"en\")\n }\n
    OPTIONAL {\n  ?class rdfs:comment ?comes .\n  FILTER (lang(?
    comes) = \"es\")\n }\n OPTIONAL {\n  ?class rdfs:comment ?
    comen .\n  FILTER (lang(?comen) = \"en\")\n }\n}",
    "variables": [
        "class",
        "parent",
        "labes",
        "laben",
        "comes",
        "comen"
    ],
    "parameters": [
        {
            "label": "ancestor",
            "type": "iri"
        }
    ],
    "endpoint": "educawood"
},
{
    "id": "lastEducatrees",
    "description": "Obtain the last educatrees (variable \"tree\") of an
    optional primary species (parameter \"species\") and their creation
    date (variable \"date\"). This template query can be paginated with
    the optional parameters \"limit\" and \"offset\"",
    "template": "SELECT DISTINCT ?tree ?date\nWHERE {\n  ?tree a
    <http://educawood.gsic.uva.es/sta/ontology/Tree>{{#species}} .\n  ?
    sp rdfs:subClassOf* <{{{species}}}> .\n  ?tree
    <http://educawood.gsic.uva.es/sta/ontology/hasPrimarySpecies>/
    <http://educawood.gsic.uva.es/sta/ontology/hasTaxon> ?
    sp{{/species}} ;\n    <http://purl.org/dc/terms/created> ?date .
    }\nORDER BY DESC(?date)\n{{#limit}}LIMIT {{limit}}{{/limit}}
    {{^limit}}LIMIT 20{{/limit}}{{#offset}}\nOFFSET {{offset}}{{/offset}}",
    "variables": [
        "tree",
        "date"
    ],
```

```
            "parameters": [
                {
                    "label": "species",
                    "type": "iri",
                    "optional": true
                },
                {
                    "label": "limit",
                    "type": "integer",
                    "optional": true
                },
                {
                    "label": "offset",
                    "type": "integer",
                    "optional": true
                }
            ],
            "endpoint": "educawood"
        },
        {
            "id": "patchesinbox",
            "description": "Obtain patches (variable \"patch\"), polygons
            (variable \"poly\"), their bounds (variables \"west\", \"east\",
            \"north\", \"south\" ), and their area (variable \"area\") in a layer
            (parameter \"layer\") with a minimum size (parameter
            \"areamin\") in a box (parameters \"lngwest\", \"lngeast\",
            \"latsouth\", \"latnorth\") with an optional soil use (parameter
            \"soiluse\"). This template query can be paginated with the
            optional parameters \"limit\" and \"offset\"",
            "template": "SELECT DISTINCT ?patch ?poly ?west ?east ?north ?
            south ?area\nWHERE {\n  ?patch <http://crossforest.eu/
            position/ontology/hasPolygon> ?poly .\n{{#soiluse}}  ?patch a
            <{{{soiluse}}}> .\n{{/soiluse}}  ?poly
            <http://epsg.w3id.org/ontology/hasLeftBound107> ?west ;\n
            <http://epsg.w3id.org/ontology/hasRightBound107> ?east ;\n
            <http://epsg.w3id.org/ontology/hasUpperBound106> ?north ;\n
            <http://epsg.w3id.org/ontology/hasLowerBound106> ?south ;\n
            <http://crossforest.eu/position/ontology/hasAreaInSquareMeters> ?
            area .\n{{#layer}}  ?poly
            <http://crossforest.eu/position/ontology/isInLayer> <{{{layer}}}>
            .\n{{/layer}}{{#latnorth}}  FILTER (?south < {{latnorth}})
            .\n{{/latnorth}}{{#latsouth}}  FILTER (?north > {{latsouth}})
            .\n{{/latsouth}}{{#lngeast}}  FILTER (?west < {{lngeast}})
            .\n{{/lngeast}}{{#lngwest}}  FILTER (?east > {{lngwest}})
            .\n{{/lngwest}}{{#areamin}}  FILTER (?area > {{areamin}})
            .\n{{/areamin}}}\n{{#limit}}LIMIT {{limit}}{{/limit}}{{^limit}}LIMIT
            100{{/limit}}{{#offset}}\nOFFSET {{offset}}{{/offset}}",
            "variables": [
                "patch",
```

```
        "poly",
        "west",
        "east",
        "north",
        "south",
        "area"
    ],
    "parameters": [
        {
            "label": "layer",
            "type": "iri",
            "optional": true
        },
        {
            "label": "soiluse",
            "type": "iri",
            "optional": true
        },
        {
            "label": "lngwest",
            "type": "number",
            "optional": true
        },
        {
            "label": "lngeast",
            "type": "number",
            "optional": true
        },
        {
            "label": "latnorth",
            "type": "number",
            "optional": true
        },
        {
            "label": "latsouth",
            "type": "number",
            "optional": true
        },
        {
            "label": "areamin",
            "type": "number",
            "optional": true
        },
        {
            "label": "limit",
            "type": "integer",
            "optional": true
        },
        {
```

```
                "label": "offset",
                "type": "integer",
                "optional": true
            }
        ],
        "endpoint": "crossforest"
    },
    {
        "id": "treesinbox",
        "description": "Obtain trees (variable \"tree\") of an optional
        species (parameter \"species\") and their GPS coordinates
        (variables \"lat\" and \"lng\") in a bounding box with GPS
        coordinates \"latsouth\", \"latsouth\", \"latsouth\", and
        \"latsouth\". This template query can be paginated with the
        optional parameters \"limit\" and \"offset\"",
        "template": "SELECT DISTINCT ?tree ?lat ?lng\nWHERE {\n  ?tree
        a <https://datos.iepnb.es/def/sector-publico/medio-
        ambiente/ifn/Tree>{{#species}} ; a/rdfs:subClassOf* <{{{species}}}>
        {{/species}} ;\n
        <http://crossforest.eu/position/ontology/hasPosition> ?pos .\n  ?pos
        <http://crossforest.eu/position/ontology/
        hasCoordinateReferenceSystem>
        <http://epsg.w3id.org/data/crs/4326> ;\n
        <http://epsg.w3id.org/ontology/axis/1> ?lat ;\n
        <http://epsg.w3id.org/ontology/axis/2> ?lng .\n{{#latnorth}} FILTER
        (?lat < {{latnorth}}) .\n{{/latnorth}}{{#latsouth}}  FILTER (?lat >
        {{latsouth}}) .\n{{/latsouth}}{{#lngeast}}  FILTER (?lng < {{lngeast}})
        .\n{{/lngeast}}{{#lngwest}}  FILTER (?lng > {{lngwest}})
        .\n{{/lngwest}}}\n{{#limit}}LIMIT {{limit}}{{/limit}}{{^limit}}LIMIT
        100{{/limit}}{{#offset}}\nOFFSET {{offset}}{{/offset}}",
        "variables": [
            "tree",
            "lat",
            "lng"
        ],
        "parameters": [
            {
                "label": "species",
                "type": "iri",
                "optional": true
            },
            {
                "label": "lngwest",
                "type": "number",
                "optional": true
            },
            {
                "label": "lngeast",
                "type": "number",
```

```
            "optional": true
        },
        {
            "label": "latnorth",
            "type": "number",
            "optional": true
        },
        {
            "label": "latsouth",
            "type": "number",
            "optional": true
        },
        {
            "label": "limit",
            "type": "integer",
            "optional": true
        },
        {
            "label": "offset",
            "type": "integer",
            "optional": true
        }
    ],
    "endpoint": "crossforest"
},
{
    "id": "educatreesinbox",
    "description": "Obtain educatrees (variable \"tree\") of an optional
    primary species (parameter \"species\") and their GPS coordinates
    corresponding to their primary location (variables \"lat\" and
    \"lng\") in a bounding box with GPS coordinates \"latsouth\",
    \"latsouth\", \"latsouth\", and \"latsouth\". This template query
    can be paginated with the optional parameters \"limit\" and
    \"offset\"",
    "template": "SELECT DISTINCT ?tree ?lat ?lng\nWHERE {\n  ?tree
    a <http://educawood.gsic.uva.es/sta/ontology/Tree>{{#species}} .\n
    ?sp rdfs:subClassOf* <{{{species}}}> .\n  ?tree
    <http://educawood.gsic.uva.es/sta/ontology/hasPrimarySpecies>/
    <http://educawood.gsic.uva.es/sta/ontology/hasTaxon> ?
    sp{{/species}} ;\n
    <http://educawood.gsic.uva.es/sta/ontology/hasPrimaryPosition> ?
    pos .\n    ?pos <http://www.w3.org/2003/01/geo/wgs84_pos#lat> ?
    lat ;\n    <http://www.w3.org/2003/01/geo/wgs84_pos#long> ?lng
    .\n{{#latnorth}} FILTER (?lat < {{latnorth}}) .\n{{/latnorth}}
    {{#latsouth}}  FILTER (?lat > {{latsouth}}) .\n{{/latsouth}}{{#lngeast}}
    FILTER (?lng < {{lngeast}}) .\n{{/lngeast}}{{#lngwest}}  FILTER (?lng
    > {{lngwest}}) .\n{{/lngwest}}}\n{{#limit}}LIMIT {{limit}}{{/limit}}
    {{^limit}}LIMIT 100{{/limit}}{{#offset}}\nOFFSET {{offset}}{{/offset}}",
    "variables": [
```

```
                "tree",
                "lat",
                "lng"
            ],
            "parameters": [
                {
                    "label": "species",
                    "type": "iri",
                    "optional": true
                },
                {
                    "label": "lngwest",
                    "type": "number",
                    "optional": true
                },
                {
                    "label": "lngeast",
                    "type": "number",
                    "optional": true
                },
                {
                    "label": "latnorth",
                    "type": "number",
                    "optional": true
                },
                {
                    "label": "latsouth",
                    "type": "number",
                    "optional": true
                },
                {
                    "label": "limit",
                    "type": "integer",
                    "optional": true
                },
                {
                    "label": "offset",
                    "type": "integer",
                    "optional": true
                }
            ],
            "endpoint": "educawood"
        }
    ]
}
```

Apéndice D

Fichero de configuración de Firebase

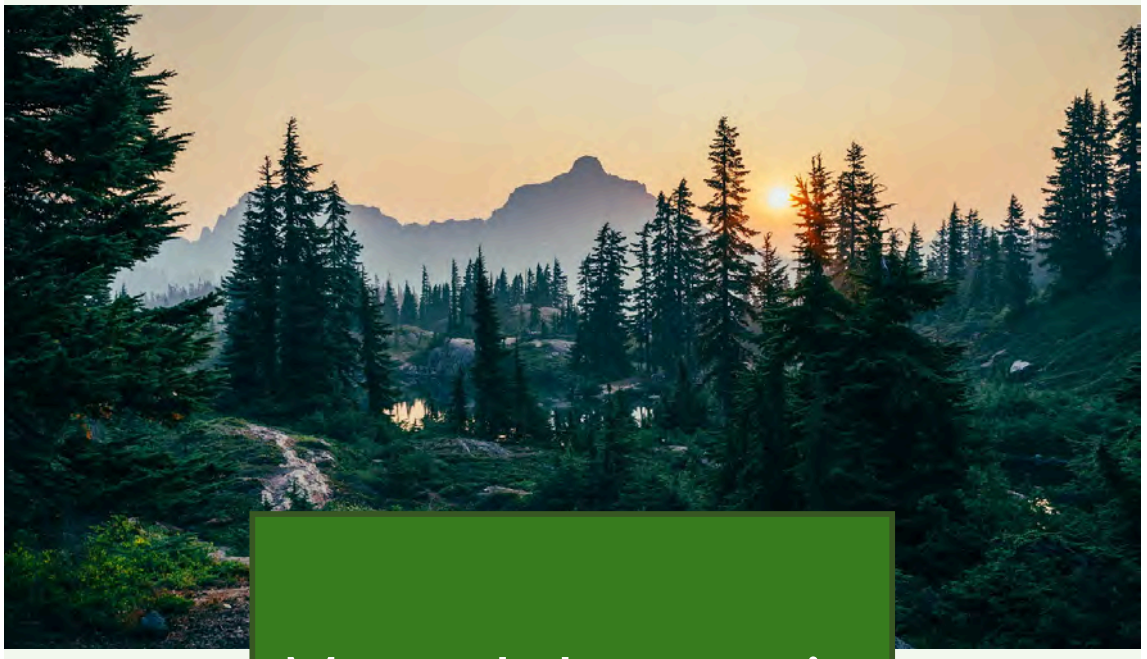
```
1  /* Fichero que almacena las variables de configuracion del proyecto */
2
3  // Variables de configuracion de las credenciales de la cuenta de ...
       Firebase
4  // Se han omitido algunos caracteres por privacidad
5
6  export const firebaseConfig = {
7      apiKey: "***********",
8      authDomain: "educawood-38078.firebaseapp.com",
9      projectId: "educawood-38078",
10     storageBucket: "gs://educawood-38078.appspot.com/",
11     messagingSenderId: "720568845286",
12     appId: "***********",
13     measurementId: "G-SFBHPH5SY5"
14   };
```

Apéndice E

Manual de usuario de EducaWood

En la página siguiente se muestra el manual de usuario de la aplicación web desarrollada.

EducaWood



Manual de usuario

EducaWood

ÍNDICE

INTRODUCCIÓN

Bienvenido a EducaWood, la aplicación web de educación medioambiental con la que podrás disfrutar tanto en el campo como en el aula de los árboles y todas sus características.

Con EducaWood podrás consultar la información forestal de los suelos de todo el territorio español, además de los árboles registrados en el Inventario Forestal Nacional. EducaWood te permitirá añadir nuevos árboles de los que anotar su especie, su localización, su estado, sus medidas, microhábitats e, incluso, fotografías de los mismos.

Cualquier usuario puede navegar por el mapa para consultar la información forestal. Los usuarios registrados podrán realizar nuevas anotaciones.

REQUERIMIENTOS

Los requerimientos mínimos para que EducaWood funcione correctamente son los siguientes:

- Dispositivo móvil, tableta u ordenador con conexión a Internet.
- Navegador web Mozilla Firefox, Google Chrome o Safari.

El lenguaje predeterminado de la aplicación (y el único en la primera versión) es el español.

La aplicación se adapta a cualquier tipo de dispositivo y tamaño de pantalla.

EducaWood no requiere de ninguna instalación adicional. Basta con acceder al enlace: https://educawood.gsic.uva.es/

USO DE LA APLICACIÓN

Página principal

La página principal de EducaWood se muestra en las dos figuras siguientes. A través de ella se puede acceder a todas las funcionalidades de la aplicación. En la parte superior de la pantalla se encuentra el menú de navegación. En la parte inferior de la misma se pueden consultar las últimas anotaciones de árboles realizadas en la aplicación.

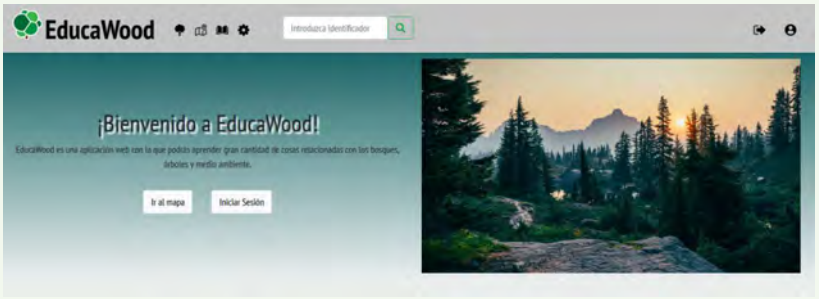


Figura 1: Primera parte de la página principal de EducaWood

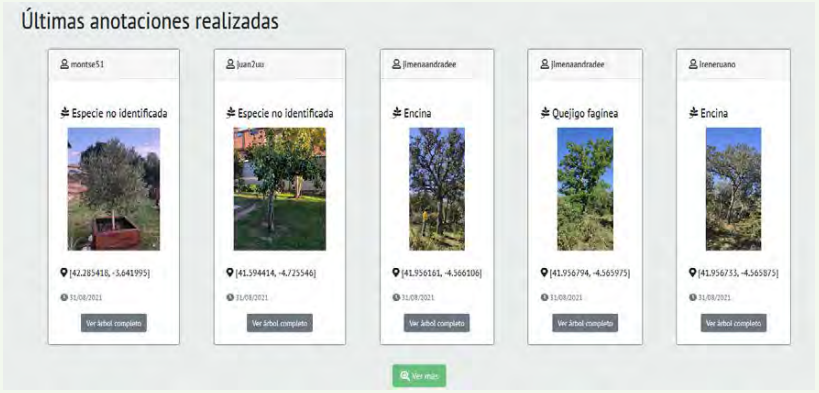


Figura 2: Segunda parte de la página principal de EducaWood

Creación de una cuenta e inicio de sesión

Cualquier usuario puede crearse una cuenta en EducaWood. Para ello debe acceder a la interfaz de **registro** e **inicio de sesión** a través del siguiente icono:

Para crear una cuenta es necesario disponer de un correo electrónico, un nombre de usuario y una contraseña segura. Una vez introducidos los datos en el formulario de registro, se enviará un correo electrónico a su cuenta. Para finalizar el proceso de registro deberá pinchar en el enlace del mensaje recibido.

Una vez disponga de una cuenta de usuario en EducaWood, podrá iniciar sesión a través del formulario de inicio de sesión.

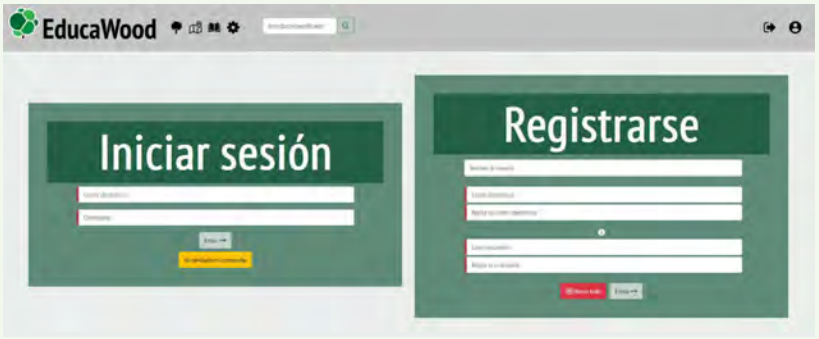
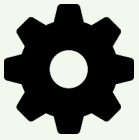


Figura 3: Interfaz de registro e inicio de sesión

Para cerrar sesión, deberá pinchar el icono siguiente:

Ajustes

Para acceder a la interfaz de ajustes debe seleccionar el icono siguiente:



Debe haber iniciado sesión en la aplicación para poder acceder al apartado de ajustes.

Desde esta interfaz podrá modificar sus datos personales y su contraseña a través de los botones disponibles en la parte superior izquierda.

El cambio de contraseña se realizará a través de su correo electrónico.

El apartado de privacidad no está disponible.

En el apartado de información, encontrará la información relativa al proyecto y un contacto en caso de encontrar algún problema en la aplicación.

Figura 4: Interfaz de ajustes

Navegación por el mapa

EducaWood dispone de un mapa interactivo con información forestal muy variada.

Navegue hasta la zona de interés que desee y seleccione el botón *Obtener teselas* para que se cargue la información forestal relativa al uso del suelo de dicha zona. Puede utilizar el botón de geolocalización diponible en la parte izquierda del mapa.

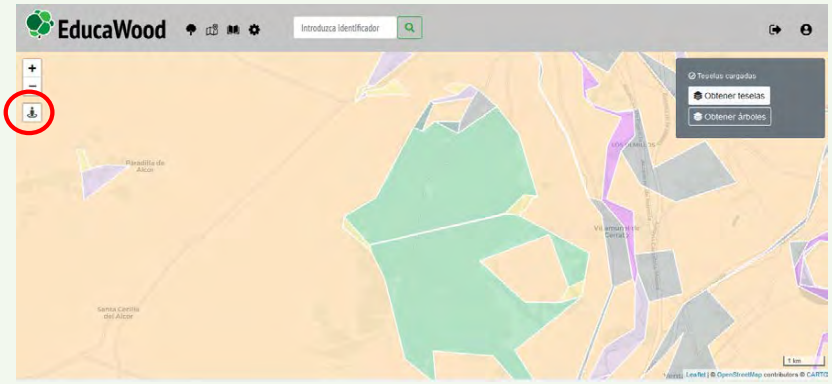


Figura 5: Interfaz del mapa forestal

Cada una de las teselas se pintará de un color, en función del tipo de suelo. Para obtener la información de una tesela concreta, selecciónela. A continuación, aparecerá una ventana informativa con la provincia en la que se encuentra, el área total que abarca, el tipo de suelo y las especies más abundantes en la misma.

Figura 6: Información de una tesela

Si desea obtener los árboles registrados en una zona, seleccione el botón *Obtener árboles* en la leyenda de la derecha.

Se cargarán todos los árboles disponibles en la aplicación. Los árboles de color verde se corresponden con los árboles registrados en el Inventario Forestal Nacional, mientras que los de color morado son los creados por otros usuarios de EducaWood.

Seleccionando cada uno de los árboles puede consultar su información.

Para el caso de los árboles morados, si desea ver su información detallada, debe copiar el identificador del árbol e introducirlo en la barra de búsqueda del encabezado de la aplicación. De esta manera, la aplicación le trasladará a la interfaz con los detalles del árbol deseado.

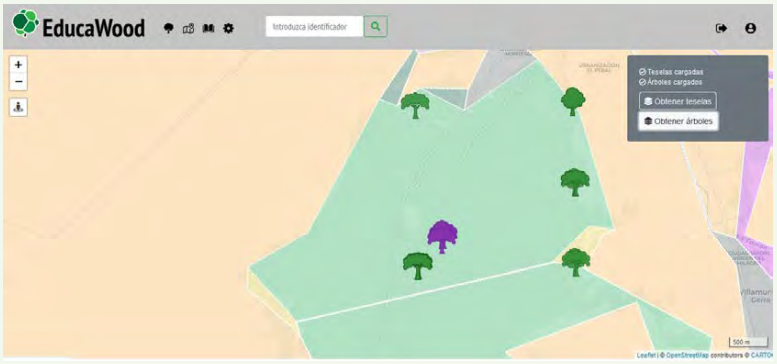


Figura 7: Mapa con los árboles cargados

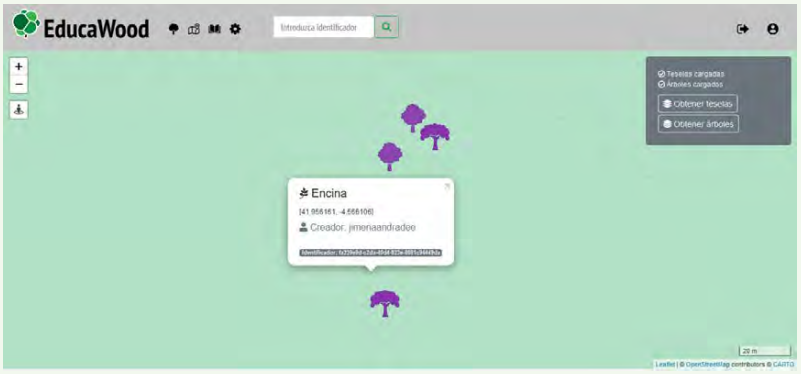


Figura 8: Selección de un árbol

Creación de un nuevo árbol

Para crear un nuevo árbol debe acceder a la interfaz de creación a través del icono siguiente:

Recuerde que es necesario iniciar sesión para poder anotar un árbol nuevo.

El formulario de creación de ilustra en la Figura 9.

Vaya desplegando cada una de las secciones que componen el árbol a través de la flecha disponible en la parte derecha del acordeón.

Complete todos los campos que desee y seleccione *Crear*.

El campo de ubicación es obligatorio para la creación del árbol. Puede permitir a la aplicación utilizar su ubicación en tiempo real para facilitar la obtención de las coordenadas.

Tras la creación, la aplicación le trasladará a la interfaz en la que podrá visualizar los detalles del árbol.

Figura 9: Interfaz de creación de un árbol nuevo

Visualización de los detalles de un árbol

Se puede acceder a los detalles de un árbol tanto si acaba de crear uno nuevo o desde la barra de búsqueda.

Si conoce el identificador del árbol que desea consultar, introdúzcalo en la barra de búsqueda del encabezado de la aplicación.

La interfaz con los detalles de un árbol concreto se muestra en la Figura 10.

En la pestaña de *Datos básicos* encontrará la información relativa a las anotaciones primarias del árbol. Las **anotaciones primarias** se corresponden con las anotaciones que se consideran válidas.

Figura 10: Interfaz con los detalles de un árbol

El resto de anotaciones se pueden ir consultando en las diferentes pestañas de la interfaz. Las anotaciones primarias se diferencian del resto con una estrella de color amarillo.

Añadir anotaciones a un árbol

Si se encuentra observando los detalles de un árbol y desea añadir una nueva anotación al mismo, debe seleccionar el botón *Nueva anotación* disponible en la parte inferior de la pantalla.

Veamos un ejemplo con el árbol de la Figura 10. Se observa que tiene una anotación de imagen (Figura 11).

Figura 11: Anotación de imagen

Seleccionando la opción de *Nueva anotación de imagen* se le abrirá una ventana emergente con un formulario en el que subir la imagen. Además, podrá añadir un título, una descripción y la parte del árbol que se muestra en la imagen.

Figura 12: Formulario para añadir una imagen nueva

Rellene el formulario y seleccione *Crear*.

Tras esperar a que la aplicación registre la nueva imagen, podrá observar la nueva anotación de imagen añadida.

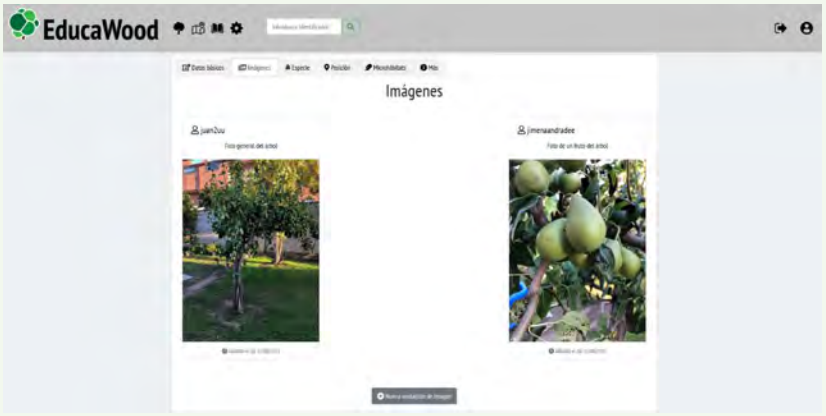


Figura 13: Anotaciones de imagen del árbol

Sección de actividades

A través del icono mostrado arriba se accede a la sección destinada a las actividades de EducaWood.

Esta sección no se encuentra implementada en la primera versión de EducaWood.

El equipo de EducaWood espera que disfrute y aprenda mucho con la aplicación

¡Bienvenido!