Universidad de la Habana

Open Latino Server: Nuevo tipo de mapa temático e interfaz visual para configuración.

Autor: Javier Alejandro Campos Matanzas
Tutora: Msc. Joanna Campbell Amos

Trabajo de Diploma
presentado en opción al título
de Licenciado en Ciencia de la Computación.

Facultad de Matemática y Computación
Departamento de programación, ingeniería de software y base de datos
2022

Agradecimientos

Opinión del Tutor

Resumen

En el presente trabajo de diploma se implementarán mejoras y nuevas funcionalidades para el servidor Open Latino, relacionadas con los mapas temáticos y con el manejo de su configuración. Basados en la investigación sobre otros sistemas y librerías que usan mapas temáticos, se decidió que lo más conveniente fuera la implementación de un nuevo tipo de temático, al que llamaremos mapa temático por clasificación de tipos o de categorías, que es más intuitivo para su utilización por parte de los usuarios. También, como parte de este trabajo, se encuentra la creación de un nuevo pedido que devuelve información acerca de los temáticos creados en el sistema. Por último, luego de analizar las diferentes interfaces visuales para configuración de servidores, se llegó a la conclusión de que es necesario una implementación propia que responda a las necesidades de este servidor en particular, usando poderosos frameworks para este fin, como Angular.

En los capítulos finales se realizan varias pruebas con el fin de verificar el correcto funcionamiento del servidor y de sus nuevas funcionalidades, brindando las conclusiones a las que se arribaron y se propone el trabajo futuro con Open Latino Server (OLS).

Palabras Clave

Mapa temático, protocolo WMS, React, interfaz de usuario, ASP .Net Core, sistema, servidor, backend, frontend, framework, servidor de mapas, Sistemas de Información Geográfica (SIG), SIG Web

Abstract

This diploma project is proposed to implement improvements and new functionalities for the Open Latino server, related to the thematic maps and managing server configuration. Based on research on other systems and libraries that use thematic maps, it was decided that the most convenient solution to the problem would be the implementation of a new type of thematic, which we will call thematic map by type classification, which is more intuitive for users. Also, as part of this work, the creation of a new request that returns information about the thematics that are created in the system. Finally, after analyzing the different settings visual interfaces of servers, it was reached the conclusion of an own implementation, that responds to the needs of this server in particular, is necessary, using powerful frameworks, like Angular.

Several tests are performed in the final chapters to verify the correct operation of the server and its new functionalities, providing the conclusions that we arrived and future work with Open Latino Server(OLS).

KeyWords

Thematic map, WMS protocol, React, user interface, ASP .Net Core, system, server, backend, frontend, framework, Internet Map Server (IMS), Geografic Information System (GIS), Web GIS

Índice general

Índice de figuras

Capítulo 1

Introducción

Hasta hace algunas décadas, los mapas se realizaban obteniendo las coordenadas de los puntos por métodos geodésicos, topográficos y astronómicos para que después mediante métodos cartográficos se elaborara el mapa correspondiente de acuerdo con la proyección y la escala seleccionada; sin embargo, a partir de la década de los 70, con el lanzamiento de los satélites Landsat y con el establecimientos de los procesos cartográficos digitales, la forma de hacer mapas paulatinamente se volvió más dinámica. En la actualidad, usando los diferentes satélites, GPS [1] y los Sistemas de Información Geográfica (SIG) [2], el ser humano elabora diferentes mapas que permiten analizar con mayor profundidad la información contenida en un territorio.

Un mapa es la representación gráfica de un territorio sobre una superficie bidimensional. Se define también como un dibujo o trazado esquemático que representa las características de un territorio determinado, tales como sus dimensiones, coordenadas, accidentes geográficos u otros aspectos relevantes. La función principal de los mapas es brindar información sintetizada sobre puntos de localización y coordenadas de orientación, así como también sobre rutas disponibles, características de la superficie terrestre (relieves, redes fluviales, recursos, etc.), clima regional, límites político-territoriales, puntos de interés, distribución de la población, etc.[12]

El siglo XX fue testigo de un avance extraordinario en el desarrollo de la cartografía, decisivo en este sentido fue el desarrollo de la fotografía aérea y el despliegue

[1] Sistema de navegación y localización mediante satélites.

[2] Sistema de información capaz de integrar, almacenar, editar, analizar, compartir y mostrar la información geográficamente referenciada.

satelital hasta llegar al uso de Sistemas de Información Geográfica [22] (SIG o GIS, por sus siglas en inglés, Geografic Information System).

En las últimas cinco décadas, los Sistemas de Información Geográfica han evolucionado desde un concepto a una ciencia. Esta magnífica evolución hace que los SIG pasen de ser una herramienta rudimentaria a convertirse en una poderosa plataforma para comprender y planificar nuestro mundo. El campo de los Sistemas de Información Geográfica está marcado por diversos hitos, comenzó en los años sesenta, mientras emergían las computadoras y los primeros conceptos de geografía cuantitativa y computacional. El trabajo pionero de Roger Tomlinson [13] para iniciar, planificar y desarrollar el Sistema de Información Geográfica de Canadá, dio como resultado el primer SIG computarizado del mundo, en 1963.

El futuro de SIG, junto con su pasaje a la web y computación en la nube y la integración con la información en tiempo real a través del internet de las cosas(IoT) [3], se convirtió en una plataforma relevante a casi toda actividad humana, el sistema nervioso del planeta. Mientras que el mundo enfrenta desafíos, tales como la expansión de la población, desforestación y contaminación, los SIG [11] jugarán un papel cada vez más importante en cómo entendemos y abordamos estos problemas y proporcionan el medio para comunicar soluciones utilizando el lenguaje común de los mapas.

El acelerado desarrollo de internet hizo más fácil compartir y actualizar la información, lo que motivó el surgimiento de los Web GIS, un patrón para la implementación de un GIS moderno, impulsado por un web-servicio estandar que entrega datos, capacidades y conecta componentes. Un Web GIS puede ser implementado en la nube con permisos o, más común, como una combinación híbrida obteniendo lo mejor de ambos mundos. Dentro de sus componentes más importantes se encuentra el Servidor de Mapas(en inglés conocido como IMS: Internet Map Server) [15], de gran utilidad para manejar grandes volúmenes de información provenientes de diversos proveedores y realizar diversas consultas u operaciones (Queries en inglés) sobre los datos que maneja en solo unos milisegundos, incluso si se trata de millones de datos.

A raíz de estos avances tecnológicos, surgieron nuevas herramientas que facilitan

[3] es el proceso que permite conectar elementos físicos cotidianos al Internet [14]

el estudio científico a partir de variables diversas y complejas; como ejemplo, se pueden mencionar los mapas de edificios en una ciudad que constituyan hospitales. Su diferencia radica principalmente en la escala del mapa; es decir, generales o detallados, dependiendo del nivel de especialización al que desee alcanzar el investigador. Esto no es más que un mapa temático.

Una de las funcionalidades que también tienen los IMS es la de dar, precisamente, mapas temáticos, cuyo objetivo es reflejar un aspecto particular de la zona geográfica sobre la que se definen. Pueden centrarse en variables físicas, sociales, políticas, culturales, económicas, sociológicas y cualquier otra relacionada con un territorio concreto. Los mapas temáticos recogen y aportan información sobre temas geográficos peculiares. Pueden ser analíticos si representan un único elemento gráfico, o sintéticos si reúnen datos de diferentes mapas. [24]

Una diferenciación necesaria los clasifica en cualitativos (aquellos que representan fenómenos sin tener ninguna precisión numérica) y cuantitativos (los que representan el valor numérico de un fenómeno). Así pues, nos encontramos ante una gran variedad de tipos de mapas temáticos. Entre ellos, destacan los de isolíneas, que usan líneas curvas para unir puntos de igual valor de un fenómeno, los mapas de flujos, que consisten en líneas de diferente espesor para representaciones dinámicas y los mapas anamórficos, que dependen de la magnitud del fenómeno representado [24]. Es decir, cambian el tamaño real de los países para hacerlo proporcional al hecho que cartografían. Los mapas temáticos utilizan los mapas topográficos [4] como mapas base para la representación gráfica de datos de diversa índole, lo que se conoce como cartografía temática.

Los servidores de mapas son los encargados de servir los Datos Espaciales [5] de los territorios, los cuales son agrupados en conjuntos, denominados Capas, que al combinarse dan como resultado los mapas, además si se le agregan a estas capas un grupo de restricciones se obtiene un mapa temático [21]. Estas capas pueden contener dos tipos de datos: Raster que son cualquier tipo de imagen digital representada en mallas o Vectorial que están destinados a almacenar con gran exactitud los elementos geográficos. Cada uno de estos está vinculado a una fila en una base de datos que

[4] una imagen, generalmente del relieve de la superficie terrestre, a una escala definida.

[5] dato que tiene asociada una referencia geográfica de tal modo que se puede localizar exactamente dónde sucede dentro de un mapa.

describe sus atributos y polígonos.

El desarrollo exponencial, a pasos acelerados, de la humanidad y su relación directa con el perfeccionamiento de los SIG constituyen la razón de ser de esta investigación, con énfasis en los mapas temáticos, haciéndolos más útiles y fáciles de manejar por los usuarios, llevándolos a un nivel superior, donde se haga más simple su manejo pero a la vez más complejo y específico su contenido, mejorando su eficacia.

1.1. Formulación del problema, motivación y justificación

La Casa del Software de la Facultad de Matemática y Computación de la Universidad de La Habana ha venido desarrollando un Sistema de Información Geográfica llamado Open Latino GIS. Entre los componentes de este sistema se encuentra Open Latino Server (OLS). Este servidor fue desarrollado usando las buenas prácticas de la programación orientada a objetos [19] y los principios SOLID [3] por lo que su código es fácil de entender y permite incorporar mejoras fácilmente. Fue creado usando la tecnología de ASP.NET Framework [16], en consecuencia, solo podía ejecutarse en entorno Windows y publicarse usando IIS [10].

Teniendo como objetivo que fuera multiplataforma [6], se desarrolló una versión 2.0 utilizando NET CORE [18]. La nueva versión de OLS completó la implementación del protocolo WMS [7] (Web Map Service) [23] que se encontraba pendiente en su versión anterior. En este se encuentran una serie de pedidos básicos entre los que se puede destacar GetMap, GetCapabilities,etc. Este servidor cuenta, además, con un pedido para visualizar mapas temáticos.

A pesar de las muchas funcionalidades con las que se cuenta, se presenta el problema de que este servidor carece de una interfaz de usuario[8] amigable, para poder configurarlo es necesario hacerlo a nivel de código, mediante el uso de programas para modificar las bases de datos. OLS tampoco cuenta con una funcionalidad que permita

[6] proyectos que operan en múltiples plataformas informáticas.

[7] El protocolo estándar WMS permite, a grandes rasgos, servir imágenes georeferenciadas a través de internet

[8] Punto de interacción y comunicación humano-computadora en un dispositivo.

conocer los detalles acerca de los temáticos que están definidos en este, actualmente se deben realizar consultas sql [9] para conocer esa información. Por último, los temáticos que existen son por consulta, es decir, la definición de nuevos temáticos se realiza usando consultas de sql.

Resulta de importancia agregar las funcionalidades anteriores al servidor ya que al usuario final se le dificulta el trabajo con el sistema, debido a que debe tener nociones elementales de programación y conociminetos básicos de sql para manipular la configuración de OLS e interactuar con las funcionalidades referentes a los mapas temáticos. Se hace necesario, entonces, la implementación de una interfaz de usuario amigable para el trabajo con la configuración del servidor y los mapas temáticos, así como el desarrollo de un nuevo tipo de mapas temáticos, más intuitivo, de clasificación por tipos o de categorías. Además, de la creación de un nuevo pedido que devuelva toda la información relacionada a los temáticos definidos en OLS.

Para dar solución a las cuestiones planteadas anteriormente, es necesario dar respuesta a las siguientes preguntas:

- ¿Existen en la actualidad soluciones que permitan representar la información estadística con la que cuentan las personas en un área geoespacial determinada?
- ¿Existen soluciones para la representación de la información estadística que poseen los usuarios sobre el espacio geográfico de su interés que sea capaz de integrar disponibilidad, accesibilidad y robustez, para que los usuarios cuenten con un mecanismo eficiente que les apoye en la formación de elementos de juicio para la toma de decisiones?
- ¿Es posible crear un software que sea de código abierto y que además esté disponible estando en línea, que tenga una herramienta para brindar visualización sobre el problema en cuestión?
- ¿Qué soluciones y tecnologías existentes son apropiadas para la implementación de la presente investigación?

Es posible dar solución a las preguntas anteriores gracias a los conocimientos, herramientas y habilidades de investigación adquiridas en el transcurso de los estudios

[9] Structured Query Language, lenguaje de programación que te permite manipular y descargar datos de una base de datos

universitarios en la carrera de Ciencias de la Computación de la Universidad de La Habana. Se cuenta, además, con las tecnologías necesarias, ya se han puesto en uso en la plataforma OLS que se presta como antecedente a los nuevos cambios propuestos, además se cuenta con aplicaciones precedentes en el propio Departamento de Programación, Ingeniería de Software y Base de Datos como son C-sharp, ASP.Net Core, SQL Server entre otras que evidencian una rica experiencia de base.

1.2. Objetivos

Para resolver el problema anunciado anteriormente se plantean los siguientes objetivos principales:

1. Desarrollar una herramienta sobre la plataforma de código abierto OLS que presente un mecanismo para visualizar y configurar un nuevo tipo de mapas temáticos clasificados por valores de forma extensible.

2. Implementar una herramienta que permita consultar información acerca de los temáticos existentes en el servidor.

3. Desarrollar una interfaz visual intuitiva para manipular la configuración del servidor y los mapas temáticos.

Para cumplir con los objetivos principales explicados previamente, se plantean los siguientes objetivos secundarios:

1. Estudiar precedentes de otras aplicaciones que tengan implementado una herramienta para visualizar y configurar mapas temáticos.

2. Investigar las ventajas, desventajas y analizar la factibilidad de las mejores herramientas para implementar la interfaz visual.

3. Extender la arquitectura de clases con la que ya cuenta OLS que cumpla con los principios del SOLID, que sea extensible y mantenible para continuar con las buenas prácticas de la plataforma OLS para que cualquier implementación posterior en este tema se pueda utilizar sin necesidad de modificar, aumentando la eficacia y sencillez de futuras mejoras al proyecto.

1.3. Estructura del trabajo

Este trabajo de diploma está compuesto por 5 capítulos y se estructura de la siguiente manera:

- **Capítulo 1 :** El primer capítulo presenta una breve introducción al tema, la motivación, formulación del problema y justificación del mismo, los objetivos principales y secundarios, concluyendo este capítulo con la estructura del trabajo de diploma.

- **Capítulo 2 :** Este capítulo está dedicado al estado del arte, donde se analizarán del uso de los diferentes IMS que implementan los mapas temáticos por categorías, así como de las diferentes interfaces visuales que existen para configurar un servidor. Se expondrán las ventajas y desventajas de su uso.

- **Capítulo 3 :** En este capítulo se expone el marco teórico-conceptual del presente trabajo, donde se explican las soluciones que se dieron a los objetivos, tanto conceptualmente como algunos de sus detalles de implementación. Además, se aborda acerca del mecanismo que se implementa para la recuperación de temáticos y la integración del nuevo tipo de mapa, así como se expondrá la arquitectura usada en la implementación de la interfaz visual. Por último se da una explicación de las tecnologías empleadas y dónde se usaron.

- **Capítulo 4 :** El capítulo 4 está dedicado a exponer un conjunto de pruebas para mostrar el buen funcionamiento de la nueva versión de OLS y el cumplimineto de los objetivos planteados. En estas pruebas se van planteando los objetivos que se quieren verificar con estas y luego se presentan los resultados alcanzados con cada una.

- **Capítulo 5 :** En el quinto y último capítulo se proponen las conclusiones y recomendaciones, temas para trabajo futuro, es decir, temas que quedaron pendientes o se pueden realizar de una mejor manera. En este capítulo aparece, además, la bibliografía utilizada.

Capítulo 2

Estado del Arte

A lo largo de este capítulo se explica el punto más avanzado en que se encuentra hoy en día los SIG y los servidores de mapas, haciendo incapié en aquellos que implementan mapas temáticos, sin pasar por alto la interfaz de usuario que utilizan para la configuración de servidor. Por último, basado en la investigación de las herramientas existentes que utilizan mapas temáticos y de la interfaz visual para su configuración, se decide la mejor estrategia para utilizar sobre el servidor. No sin antes acercar al lector a la historia del origen y evolución de los mapas temáticos.

2.1. Origen y evolución de los mapas temáticos

Claudio Tolomeo (siglo II), griego o egipcio, es mejor conocido como el astrónomo autor de la idea errónea del Universo, sostenida durante catorce siglos, según la cual la Tierra ocupa el centro y los planetas giran a su alrededor. Sin embargo, el gran mérito de Tolomeo radica en la geografía.

Los mapas temáticos tienen su antecedente en Tolomeo, quien los elaboró de tipo histórico. En forma aislada aparecieron desde el siglo XVIII mapas específicos para representar algún fenómeno de la naturaleza, además de los históricos que fueron los más comunes. En la segunda mitad de este siglo se popularizaron los términos mapa y cartografía temáticos y, en esta época, se han multiplicado en grado superlativo. Los primeros mapas temáticos fueron muy simples, sin embargo, ameritaron su publicación en las revistas geológicas de mayor prestigio, aunque presentaban una

información muy general y pobre en extremo, nadie puede negar el inmenso valor de esa información.[26]

Si los mapas alcanzan un grado, digamos cercano a la perfección, puede pensarse que el tema de investigación queda clausurado. Esto es cierto sólo parcialmente. En la medida que los mapas que representaban rasgos físicos de la superficie terrestre se fueron perfeccionando, surgió la necesidad de expresar otros fenómenos y objetos: los suelos, las comunidades de flora y fauna, las rocas, los climas, la estructura profunda de la Tierra. De la cartografía general se pasó a la temática.

El mapa ha sido siempre un reflejo del estado de desarrollo de determinadas disciplinas científicas. Si actualmente hay decenas o cientos de mapas temáticos diversos, esto da una idea del estado actual de las geociencias. Uno de los más conocidos es el publicado en 1936 sobre la agricultura de Estados Unidos. Destacó por su originalidad. Posteriormente han sido editados mapas complejos en diversos países, resultado de investigaciones prolongadas e incluso multidisciplinarias, apoyadas por instituciones científicas y financieras.

Los mapas temáticos de un mismo país o región se hacen periódicamente, pretenden que la información contenida en el mismo sea fácilmente comprendida por el lector o usuario. Si esta es correcta y valiosa, pero mal expresada por no usar los colores o símbolos adecuados, la lectura del mapa se vuelve labor tortuosa. Por esto, el diseño final queda a cargo de un especialista altamente calificado, quien define colores, símbolos, tamaños de letras, grosor de líneas, distribución de la leyenda y otros problemas semejantes. Es la parte artística de la cartografía. [26]

En los últimos quince años asistimos a una auténtica revolución en el amplio campo de la cartografía, y muy especialmente en la caracterización tridimensional del territorio. Se ha pasado en unas décadas de una cartografía casi secreta, en manos de los ejércitos o de los estados, y muy limitada, a una enorme disponibilidad e incluso a la gratuidad de los materiales. Con el tiempo se han creado servidores que facilitan cartografía temática a cualquier usuario. Los servidores permiten visualizar mapas, la localización, la identificación de atributos, las consultas sencillas e incluso la conexión a bases de datos remotas para poder crear mapas temáticos. [20]

2.2. Herramientas SIG que implementan mapas temáticos

Hoy en día existen servicios en línea que son capaces de generar mapas temáticos a partir de ciertos parámetros de entrada. Sin embargo es pequeña la cantidad de técnicas de representación de datos estadísticos que se pueden manejar, en general los datos estadísticos y geográficos disponibles son los que se encuentran en los servidores de quienes administran el servicio y, en la mayoría de los casos, su código no puede ser descargado ni modificado de acuerdo a las necesidades de los usuarios. Existen herramientas de escritorio y, además, plataformas que usan servidores de mapas, lo cuales cuentan con módulos que permiten la creación de mapas temáticos y dejan al usuario utilizar los datos estadísticos que posee, sin embargo, estas son aplicaciones que no se centran en el trabajo con mapas temáticos, sino en la edición y gestión de datos geográficos por lo que en su mayoría el número de técnicas de representación de datos estadísticos con el que se trabaja no es muy amplio; además, en no todos los casos el código fuente está disponible.

2.2.1. QGis

QGis es un sistema de información geográfica de software libre[1] y de código abierto[2], multiplataforma, que funciona junto con un servidor web [25]. Esta herramienta cuenta con un pequeño módulo para crear mapas temáticos. La ayuda del software hace una descripción acerca del uso de los mismos, en esta se plantea que están disponibles cuatro modos para crear diferentes tipos de estos mapas.

Como ventajas, este sistema multiplataforma de código abierto presenta funcionalidades para la creación de los mapas temáticos, que permiten seleccionar, en un menú deplegable, varias opciones; la que nos concierne es la de Categorized, que permite asignar un color diferente a cada capa, y así, crear un mapa temático, donde separa por colores la información pertinente a estas (figura 2.1).

[1] significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software

[2] software cuyo código fuente se ha puesto a disposición de todo el mundo de manera gratuita

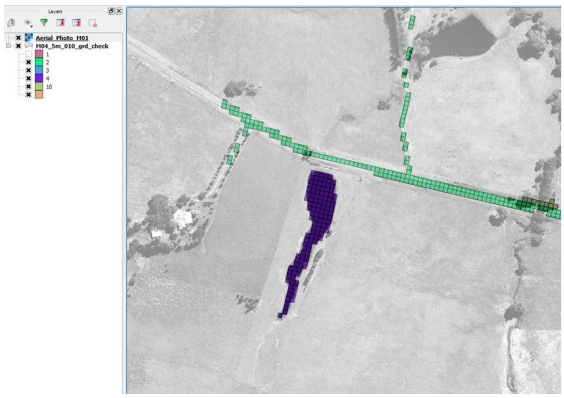

Figura 2.1: Mapa temático creado con QGis.

Además, una de las mayores ventajas de esta herramienta es la posibilidad de usar Quantum GIS como GUI (Interfaz gráfica de usuario) del SIG GRASS [7], utilizando toda la potencia de análisis de este último en un entorno de trabajo más amigable (figura 2.2). QGIS está desarrollado en C++[3], usando la biblioteca Qt[4] para su Interfaz gráfica de usuario.

A pesar de las ventajas expuestas, QGis presenta varias desventajas, que no perminten su uso en Open Latino Sever. Entre estas se encuentran que no son muchos los tipos de mapas temáticos que pueden ser creados utilizando QGis.

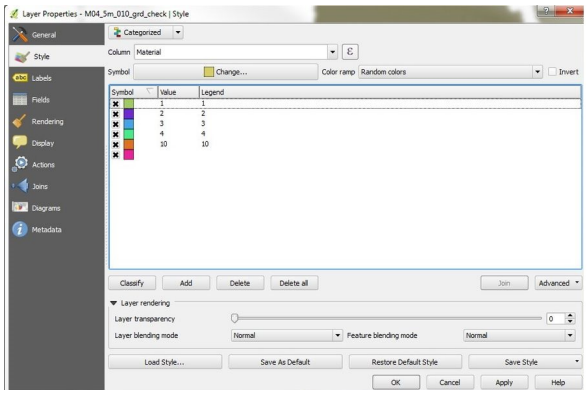

Figura 2.2: Interfaz visual QGis.

Tampoco se puede integrar al proyecto OLS con facilidad debido a que está en otro lenguaje y, para añadirlo, habría que migrar todo el código a C-sharp, que sería tan o más costoso que implementar la herramienta de mapas temáticos desde cero, además, esto no garantizará que la migración quede libre de errores. También se puede decir que esta herramienta no permite realizar restricciones sobre una misma capa para tematizar la información de esta. Por último, a pesar de que existen *bindings* [5] de la librería Qt de interfaz de usuario para C-sharp, se dificulta su uso por la misma problemática de la migración de código, sin dejar de mencionar que es necesario modificar, por no decir, cambiar por completo el código de la interfaz, puesto que el manejo de la configuración de QGis se diferencia, en gran medida, con la de OLS,

[3] lenguaje de programación multiparadigma diseñado en 1979 por Bjarne Stroustrup.

[4] framework multiplataforma orientado a objetos ampliamente usado para desarrollar programas que utilicen interfaz gráfica de usuario

[5] adaptación de una biblioteca para ser usada en un lenguaje de programación distinto de aquel en el que ha sido escrita. Esta palabra no tiene traducción al español

osea que no podemos integrar directamente el frontend al servidor, sino que se tiene que adaptar, en cuyo caso es preferible hacer una interfaz propia con un framework que se ajuste a las necesidades de OLS.

2.2.2. ArcGis

ArcGis es un completo sistema que permite recopilar, organizar, administrar, analizar, compartir y distribuir información geográfica. Permite crear y utilizar Sistemas de Información Geográfica. Una de las principales funcionalidades de ArcGIS es la creación y diseño de cartografía. Los mapas generados pueden ser de diversa tipología, siendo una de las más destacadas la de mapas temáticos (figura 2.3). Estos mapas permiten captar el interés de los usuarios y proporcionarles información de forma muy visual, ofreciendo un método ideal para mostrar los resultados del trabajo SIG.

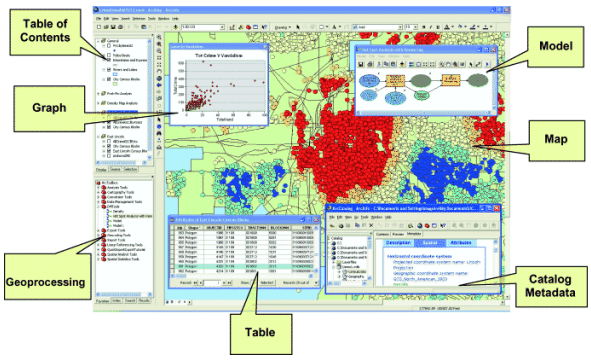

Figura 2.3: ArcGis. Interfaz visual y mapa temático.

Las ventajas que presenta esta herramienta radican en que proporciona una amplia posibilidad de recursos relacionados con los mapas temáticos. Con esta herramienta se pueden crear, consultar y analizar datos; combinar varias capas; aplicar funciones matemáticas, construir y obtener nueva información a partir de temáticos ya existentes, etc. En ArcGis, la finalidad de un mapa temático es la de representar de una o varias características de fenómenos geográficos, pudiendo dividir a estos en objetos reales: distribución, densidad, relación, etc; y conceptos abstractos: indicadores de desarrollo económico, calidad de vida, etc. Estos mapas pueden ser publicados en web permitiendo, mediante ventanas emergentes, la visualización de una o varias de estas características, fotografías de los fenómenos y acceso a otra información en la web. [25]

La principal desventaja que presenta este software es que no es código abierto, sino que se distribuye comercialmente bajo tres niveles de licencia, por lo tanto se descarta por no ser de código abierto y no ser gratuito. Además, la interfaz de

usuario que presenta es un poco complicada, debido a que cada documento diferente en ArcGIS utiliza una GUI separada, cada GUI se compone por barra de botones, herramientas, menús, estados y líneas de comando, lo que dificulta su uso por usuarios poco preparados, y no es lo que se quiere en OLS.

2.2.3. MapInfo

Las soluciones que proporciona MapInfo [17] para la creación de mapas permiten llevar a cabo análisis geográficos sencillos y complejos, acceso a datos remotos y creación de mapas temáticos que revelen patrones en los datos. Se pueden visualizar los datos como puntos, regiones zonificadas temáticamente, como gráficos de tartas o de barras , etc.

Figura 2.4: Mapa temático creado con MapInfo.

Como parte de las características ventajosas se encuentran que, con MapInfo, se pueden crear distintos mapas temáticos asignándoles a estos colores, patrones o símbolos para establecer una coincidencia con objetos según valores específicos de la tabla. La característica Mapa temático utiliza un asistente compuesto de una secuencia de tres cuadros de diálogo que facilitan la selección del tipo de mapa temático que se desea, las tablas y los campos que se utilizarán para construir el mapa y diversas opciones para personalizarlo. Las plantillas temáticas de MapInfo facilitan el comienzo de la creación de un tema. Se selecciona una plantilla que represente el tipo de mapa temático que se desea. Las plantillas se pueden personalizar completamente y pueden guardarse como nuevas plantillas en caso de que se precisen para la creación de mapas temáticos más adelante. Con MapInfo se suministran más de 40 plantillas. MapInfo cuenta con una potente y flexible interfaz de usuario, interactiva para mapas temáticos, esta interfaz permite a los usuarios anclar las barras de herramientas en los cuatro lados de la aplicación, ayudando a mejorar la eficiencia, reducir la conglomeración sobre la pantalla, y ahorrar tiempo. [17]

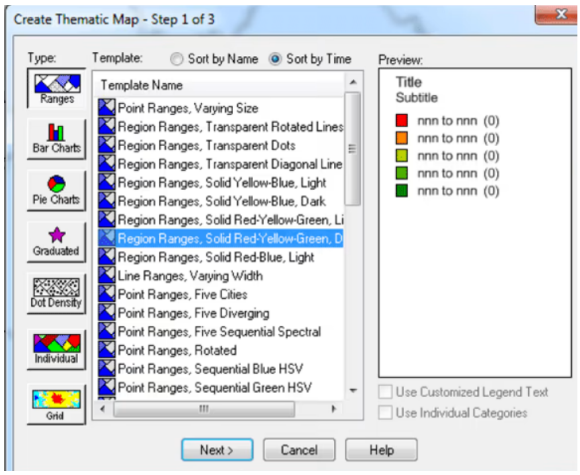

Figura 2.5: Interfaz visual de configuración de MapInfo.

Como parte de las desventajas que nos inclinan a descartar a MapInfo se encuentra que no es multiplataforma, solo está disponible para Windows[6]. Además, es un software privado de la compañía Precisely, antiguamente Pitney Bowes Inc, lo que no se corresponde con las políticas de migración a software libre e imposibilita su adquisición. Finalmente, es un software basado en Python[7], lo cual entorpecería su integración a OLS por la problemática antes mencionada de migración de código.

2.2.4. SharpMap

SharpMap [1] es una biblioteca de clases para crear aplicaciones web. Con esta librería se pueden realizar consultas a los datos espaciales para el manejo y análisis de los mismos.

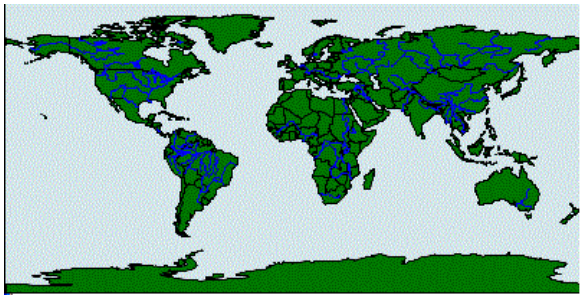

Figura 2.6: Mapa temático creado con SharpMap.

Sería ventajoso el uso de SharMap ya que es un sistema SIG escrito totalmente en C# .NET 4.0, y admite múltiples lenguajes de desarrollo .Net (C#, C++, etc). También, presenta la clase *CustomTheme*, que es usada para definir un temático propio, esta presenta dos métodos públicos: *CustomTheme*, para crear la clase, y *GetStyle*, para obtener un estilo para pintar el temático. Además, con la herramienta *SharpMap.Rendering.Thematics.CategoryTheme* se crean categorías

[6]sistema operativo, es decir, un conjunto de programas que posibilita la administración de los recursos de una computadora

[7]lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning

usando rangos de valores para comparar con el campo elegido. [1]

Una notale desventaja de esta biblioteca es que al no ser multiplataforma daría un paso atrás en el desarrollo de OLS, que ya fue mejorado a un proyecto multiplataforma. Por otro lado, al estar basado en el framework .Net 4.0, y este no presentar una versión compatible con ASP.NET Core, entra en conflicto con la última versión de OLS, que ya fue migrado a ASP.NET Core. SharpMap carece, además, de una interfaz visual, por lo que es necesario usar código, lo cual dificulta a usuarios que no estén familiarizados con la programación hacer uso de esta, y es una de las cosas que se quiere evitar en OLS. Por último, otra restricción que presenta esta biblioteca, es que no presenta muchos tipos de temáticos.

2.3. Frameworks y librerías de JavaScript que se pueden utilizar en la interfaz visual de OLS.

Como se dijo anteriormente, se hace necesario la creación de una interfaz propia que responda a las necesidades particulares de OLS. Los Frameworks de interfaz gráfica son soluciones completas que contemplan herramientas de apoyo a la construcción de software para la capa de presentación, brindando al usuario aplicaciones atractivas. Las bibliotecas, y los marcos de trabajo de JavaScript[8], facilitan el desarrollo de sitios web y aplicaciones con características y funcionalidades muy variadas, todo ello gracias a las características dinámicas, flexibles y atractivas de JavaScript. Según una encuesta de StackOverflow[9] de 2020, JavaScript sigue siendo el lenguaje de programación más utilizado (por octavo año), con un 67,7 % de los encuestados que lo utilizan. Su versatilidad favorece el desarrollo del front-end, además de las pruebas. Como resultado, se pueden encontrar muchas bibliotecas y frameworks de JavaScript que sirven para varios propósitos. En el mercado existe una amplia gama de Frameworks y librerías que proveen soluciones para el desarrollo de aplicaciones web, y la creación de interfaces gráficas. En este epígrafe se estudiarán aquellos que, por sus características y su uso en el mundo de la programación web, se consideran más importantes.

[8] lenguaje de programación que se utiliza para hacer páginas web interactivas

[9] sitio de preguntas y respuestas para desarrolladores diseñado para resolver problemas y aprender

2.3.1. React

ReactJS (figura 2.7) es una librería escrita en JavaScript de código abierto enfocada a la visualización para facilitar la creación de componentes interactivos y reutilizables para interfaces de usuario [8]. Esta librería fue lanzada en el año 2013 y desarrollada por Facebook, quienes también la mantienen actualmente junto a una comunidad de desarrolladores independientes y compañías.

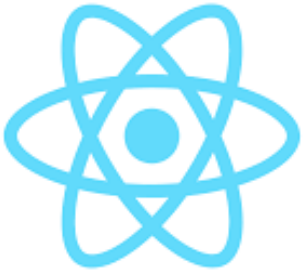

Figura 2.7: Logo React.

La característica más importante de ReactJS es el componente, una pieza de interfaz de usuario. Al diseñar una App con React, lo que se crean son componentes independientes y reusables para crear interfaces de usuario más complejas. De esta manera, ReactJS está basado en un paradigma llamado programación orientada a componentes en el que cada componente es una pieza con la que el usuario puede interactuar. Estas piezas se crean usando una sintaxis llamada JSX [8] permitiendo escribir HTML[10] (y opcionalmente CSS[11]) dentro de objetos JavaScript.

Entre las muchas ventajas que tiene el uso de React resalta el desarrollo rentable, ofreciendo una vía económica para crear aplicaciones multiplataforma. Además, se necesitan menos esfuerzos, ya que se requiere menos código en comparación con otras plataformas de desarrollo. También, dado el hecho de que ReactJS es una plataforma de código abierto con licencia del MIT, brinda acceso para usar bibliotecas y marcos de forma gratuita.

Como parte de las desventajas que motivaron a no usar esta biblioteca se destacan la ausencia de documentación oficial. La alta velocidad de desarrollo de ReactJS apenas deja lugar a una documentación apropiada, la cual es algo caótica ya que diferentes desarrolladores contribuyen sin un enfoque común. Por otro lado, no existe un estándar de desarrollo, de modo que tenemos demasiadas elecciones a tomar.

[10] lenguaje con el que se define el contenido de las páginas web
[11] lenguaje que maneja el diseño y presentación de las páginas web

Además, el alto ritmo de desarrollo de esta herramienta, provoca volver a aprender nuevas formas de hacer las cosas regularmente, cada vez que se realizan nuevos cambios, y resulta difícil para los desarrolladores adoptar todos los cambios con todas las actualizaciones continuas. Por último, la integración de React con un marco MVC[12] como nuestro servidor requiere una gran cantidad de configuración.

2.3.2. Vue

Vue (figura 2.8) es un marco de trabajo flexible y ligero basado en JavaScript que ofrece potentes herramientas web para desarrollar proyectos web frontales modernistas [9]. Vue también se considera un marco JavaScript flexible y evolutivo, ya que permite realizar cambios en el código de una aplicación sin que ello afecte a ninguna característica fundamental, lo que permite crear una interfaz de usuario progresiva. La gran flexibilidad de Vue también permite añadir módulos a medida y componentes visuales a la funcionalidad de la aplicación web.

Figura 2.8: Logo Vue.

Como parte de las ventajas de esta herramienta podemos mencionar que presenta HTML empoderado, esto significa que Vue.js tiene muchas similitudes con Angular, y esto puede ayudar a optimizar el manejo de bloques HTML usando diferentes componentes. Por otro lado, tiene una documentación muy completa, que puede acelerar la curva de aprendizaje para desarrolladores y ahorrar mucho tiempo en el desarrollo de una aplicación, usando sólo conocimientos básicos de HTML y JavaScript. Se puede decir, además, que este framework se usa para construir tanto aplicaciones de página única (SPA)[13] o complejas interfa-

[12]Modelo Vista Controlador, estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos

[13]aplicaciones o sitios web que cargan todos los recursos necesarios para navegar por los sitios web en la primera carga de la página

ces web de aplicaciones. También, Vue.js puede ocupar cerca de 20KB manteniendo su velocidad y flexibilidad, que permite alcanzar un mejor rendimiento en comparación con otros frameworks. Finalmente, Vue.js ayuda a construir grandes plantillas reutilizables en poco tiempo de acuerdo con su sencilla estructura.

Como desventaja, Vue.js aún tiene poca cuota de mercado comparado con Angular o React, lo que significa que los recursos disponibles de este framework aún están en su fase inicial. Por otra parte, su gran flexibilidad puede ser un contratiempo, puesto que, en ocasiones, Vue.js puede tener problemas para integrarse en grandes proyectos y aún no hay experiencia acerca de posibles soluciones. La barrera del idioma es un contra de esta herramienta ya que el creador es en realidad chino-estadounidense y apoya mucho a la comunidad de desarrollo china. La mayoría de los usuarios son comunidades de habla no inglesa, que son quizás uno de los mayores problemas con este marco. Predominantemente, la mayor parte de la codificación está escrita en chino, esto complica el trabajo de los desarrolladores de habla inglesa con Vue.js. Por estas razones se ha desestimado el uso de este framework.

2.3.3. Angular

Angular (figura 2.9) es una plataforma de desarrollo, construida sobre TypeScript[14], un framework basado en componentes para crear aplicaciones web escalables. Es una colección de bibliotecas bien integradas que cubren una amplia variedad de características, que incluyen enrutamiento, administración de formularios, comunicación cliente-servidor y más [6].

Angular proporciona un conjunto de herramientas que permiten desarrollar, compilar, probar y actualizar el código fuente de la aplicación. Se ha convertido en una alternativa popular para diseñar aplicaciones multiplataforma ya que admite arquitecturas MVC y MVVM[15] del lado del cliente, lo que facilita a los desarrolladores la creación de aplicaciones.

[14]lenguaje de programación construido a un nivel superior de JavaScript, que añade tipados estáticos y objetos basados en clases

[15]patrón Model-View-ViewModel, ayuda a separar limpiamente la lógica empresarial y de presentación de una aplicación de su interfaz de usuario

Figura 2.9: Logo Angular.

Como ventajas, se puede decir que Angular tiene una estructura basada en componentes lo que hace que estos sean altamente reutilizables, y simplifica el proceso de desarrollo. Angular tiene, además, una guía de estilo de documentación y CLI[16], ambas consistencias de unidad. La codificación consistente tiene varios beneficios como plantillas o fragmentos predefinidos, fácil uso de sitios, etc. También, es compatible con arquitectura MVC, y utiliza la fuente HTML para describir la interfaz de usuario de la aplicación porque es un lenguaje intuitivo, declarativo y menos complicado. Por último, cuando se implementa para proyectos de desarrollo web, Angular se integra sin problemas, ofreciendo un marco inteligente y robusto que ahorra mucho tiempo a los desarrolladores.

Este framework también presenta desventajas, entre las que pueden mencionarse su sintaxis compleja, heredada de la primera versión de Angular. Aunque Angular usa TypeScript, que es menos difícil de aprender. También, pueden aparecer problemas con la migración de anteriores versiones pero, si se usa en OLS, no serán necesarias estas migraciones.

Según el análisis de las ventajas y desventajas realizadas en este epígrafe, se decide hacer uso del framework Angular para desarrollar la interfaz visual de configuración de OLS.

2.4. Conclusiones

A lo largo de este capítulo se analizaron varios Sistemas de Información Geográfica que implementan mapas temáticos. Se estudiaron QGis, ArcGis, MapInfo y SharpMap. Los cuales presentan deficiencias si se integran a OLS, entre las causas comunes que llevaron a su desestimación se encuentran que la mayoría de estas herramientas están en otros lenguajes no compatibles con los del servidor OLS, por lo que sería igual, o más complicado, su migración a OLS, que desarrollar una herramienta

[16]herramienta de línea de comandos que se utiliza para inicializar, desarrollar, estructurar y mantener aplicaciones de Angular

propia desde cero, la excepción a esta desventaja la presenta SharpMap, pero se descartó debido a que no es multiplataforma y su versión de .Net Core no es la misma de OLS, además de no presentar una interfaz visual. Otras complicaciones que se analizaron fueron la poca capacidad para generar temáticos, como es el caso de QGis, donde, además, su interfaz visual está implementada en otro lenguaje de programación. Por otro lado, ArcGis no es de código abierto y no es gratuito, y también esta herramienta presenta una interfaz de usuario compleja. Finalmente, a pesar de que MapInfo presenta una potente y flexible interfaz de usuario y funcionalidades útiles para temáticos, se descarta por no ser multiplataforma y ser un software privado.

Por los motivos antes expuestos, se llega a la conclusión de que la mejor opción es implementar una nueva herramienta que permita configurar y visualizar mapas temáticos de clasificación de tipos.

En el segundo epígrafe, se analizaron ventajas y desventajas de frameworks y librerías de JavaScript con el objetivo de seleccionar uno para implementar la interfaz visual de OLS. Se analizaron React, Vue y Angular como marcos de trabajo. Se pudo observar que React y Vue no son tan estructurados y definidos como puede ser Angular. React, además, no cuenta con una fuerte documentación oficial y necesita mucha configuración para integrarlo a proyectos como OLS. Vue, a pesar de que está ganando popularidad, no cuenta con un fuerte apoyo de la comunidad de programadores como React o Angular, por otro lado, a pesar de tener una gran documentación, esta está en su mayoría en chino, lo que dificulta su aprendizaje. Finalmente, debido a que Angular esta basado en TypeScript, su modularidad y jerarquía de componentes, además del excelente soporte de la comunidad y de Google, entre otras ventajas previamente expuestas, se decide usar este framework para implementar la interfaz visual de OLS.

Capítulo 3

Solución Teórico-Conceptual-Computacional

El presente capítulo se enfocará en el marco teórico-computacional de la propuesta de solución, abordando las acciones que pueden realizar los usuarios de OLS y la estructura de la aplicación, así como la arquitectura y patrón de visualización empleado. Se explicarán, además, las modificaciones realizadas sobre el código fuente del servidor, dando a conocer, finalmente, algunos detalles de implementación de las nuevas herramientas creadas.

3.1. Diagrama de casos de uso en Open Latino Server

Las funcionalidades de OLS son consumidas por usuarios y aplicaciones cliente. Un usuario representa al humano que accede a la interfaz visual y puede hacer determinados cambios como agregar capas, editar temáticos, crear workspaces, etcétera. Un cliente constituye una aplicación registrada para usar los servicios del servidor de mapas[1].

[1]Los servicios son los tipos de pedidos que se pueden hacer en OLS, por ejemplo: GetMap, GetCapabilities, entre otros

3.1.1. Usuarios

En la aplicación existen dos roles de usuario: Usuario Regular y Admin. Un Usuario Regular es un usuario que se registra en el sistema, y luego puede ingresar mediante nombre de usuario y contraseña, para tener acceso a algunas de las funcionalidades de la aplicación. Admin es un administrador del sistema, tiene permisos para realizar operaciones CRUD[2] sobre las entidades para configurar el servidor. La figura 3.1 representa un diagrama de casos de uso[3] de las funcionalidades generales que tienen los usuarios en OLS.

Un usuario regular puede realizar las siguientes acciones:

- Registrarse en el sistema.
- Iniciar sesión mendiante nombre de usuario y contraseña.
- Registrar sus aplicaciones cliente y crear, editar o eliminar Workspaces para las mismas.

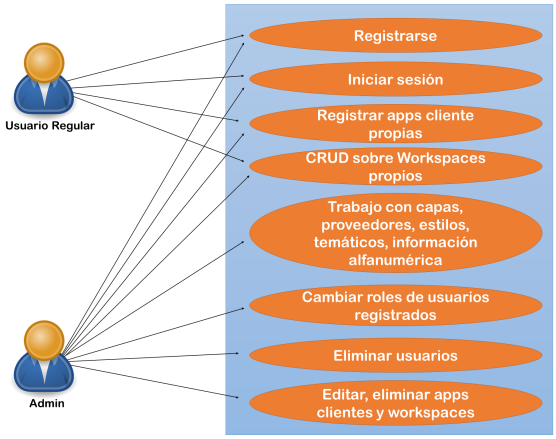

Figura 3.1: Diagrama de casos de uso de los usuarios en OLS.

Un administrador, además de realizar todas las acciones de un usuario regular, también tiene acceso a las siguientes funcionalidades:

- Crear, editar, eliminar capas, proveedores de datos, estilos, temáticos e información alfanumérica.
- Cambiar roles de los usuarios registrados en el sistema o eliminarlos totalmente del servidor.
- Tiene acceso a todos los workspaces creados por los usuarios, así como a todas las aplicaciones cliente registradas, pudiendo realizar cambios sobre estos.

[2]acrónimo de Crear, Leer, Actualizar y Borrar (del original en inglés: Create, Read, Update and Delete), que se usa para referirse a las funciones básicas en bases de datos o a la capa de persistencia en un software.

[3]Permiten visualizar las interacciones que podría tener un usuario o un cliente con un sistema.

3.1.2. Aplicaciones cliente

Los permisos que tienen las aplicaciones cliente para acceder a los servicios de OLS son controlados mediante workspaces. Un Workspace permite restringir las capas y funcionalidades del sistema. Todo cliente que se registra en el nuevo servidor de OpenLatino adquiere acceso a un Workspace llamado *common*. Este espacio de trabajo cuenta con un mínimo de funcionalidades y capas para que el cliente pueda usar. Además, como ya se dijo, el usuario que registró la aplicación puede crear nuevos Workspaces y así darle acceso a más capas y servicios.

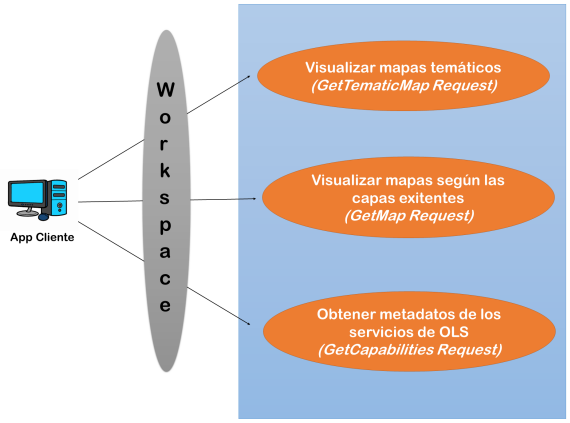

Figura 3.2: Diagrama de casos de uso de las aplicaciones cliente en OLS.

Cada aplicación cliente tiene uno o más Workspaces asignados. Como se observa en la figura 3.2, una aplicación cliente, con todos los permisos posibles permitidos, tiene acceso a las siguientes funcionalidades:

- Visualizar mapas temáticos definidos en el sistema. (Pedido GetTematicMap)
- Visualizar mapas según las capas registradas en el servidor. (Pedido GetMap)
- Obtener metadatos de los servicios que ofrece el servidor. (Pedido GetCapabilities)

3.2. Modificación al modelo de datos

Con el objetivo de crear una herramienta extensible para la creación de otros tipos de mapas temáticos y, para evitar guardar información repetida innecesariamente en la base de datos, se hace necesario la modificación del modelo de entidades relacionales (MER)[4]

[4]Herramienta para el modelo de datos, la cual facilita la representación de entidades de una base de datos.

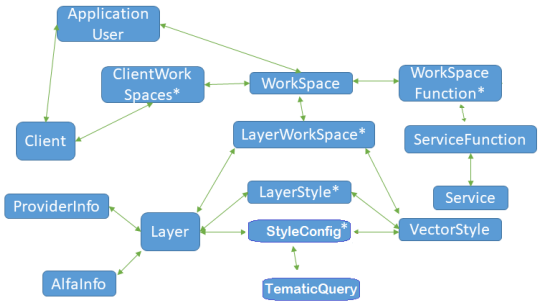

Figura 3.3: Modelo de datos de la última versión de OLS.

Como se observa en la figura 3.3, los temáticos existentes se guardaban en la tabla *TematicQuery*, la cual contiene el nombre del temático, y una función serializada que representa la consulta. De esta forma, en la tabla *StyleConfig* se definen los estilos de cada temático mediante una llave foránea al `Id` de la tabla de temáticos.

En esta nueva versión de OLS se redefinieron los mapas temáticos para que estos pudieran soportar varias consultas, las cuales, a su vez, puedan soportar varios filtros separados por los operadores AND y OR. La figura 3.4 muestra la base de datos modificada, mostrando en color rojo las tablas que sufrieron alguna variación y en color naranja las tablas que se crearon nuevas.

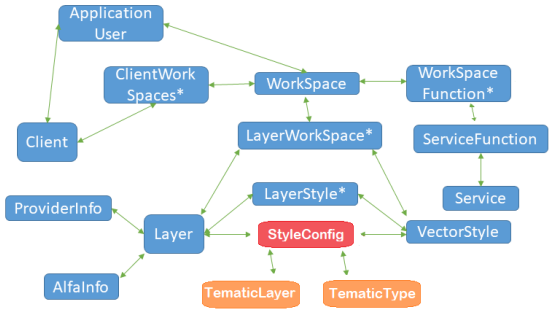

Figura 3.4: Modelo de datos modificado.

La tabla *TematicQuery* se elimina, diviendo su contenido en dos nuevas tablas, contendiendo estas, además, información acerca del tipo de temático.

Se añadió la tabla *TematicLayers*, que representa el concepto de mapa temático, está conformada por los campos `Name`, de tipo string y que representa su nombre, e `Id` que es su identificador de tipo numérico. Esta tabla tiene una relación de uno a muchos con *StyleConfig*, ya que un temático tiene una o varias configuraciones de estilo, una para cada consulta. También, se creó la tabla *TematicTypes* la cual contiene la información de los temáticos, sus filtros, y el tipo de temático que representa. Esta relacionada con la tabla *StyleConfig* en una relación de uno a muchos.

Como resultado de la incorporación de nuevas tablas relacionadas con *StyleConfig*, esta tabla se modificó, conteniendo en esta nueva versión la capa, el estilo, y el filtro

asociado a un temático específico, como llaves foráneas de las respectivas tablas.

3.3. Arquitectura de software

La arquitectura de software es el conjunto de estructuras necesarias para dar sentido a un sistema, lo cual abarca los elementos del software, las relaciones entre ellos y las propiedades de ambos. La arquitectura en capas es una de las más utilizadas, esta se enfoca en la distribución de roles y responsabilidades de forma jerárquica dando una forma muy efectiva de separación de responsabilidades. El rol indica el modo y tipo de interacción con otras capas, y la responsabilidad indica la funcionalidad que está siendo desarrollada. [4]

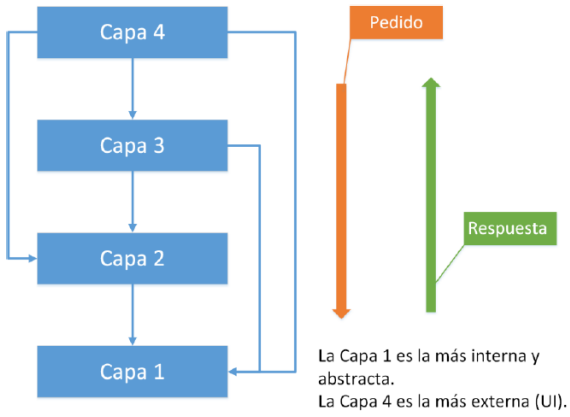

Figura 3.5: Ejemplo de Arquitectura de capas con 4 capas en su forma flexible.

En una arquitectura de capas, todas las capas se colocan de forma horizontal, donde una capa solo puede depender otra que esté por debajo de ella, nunca por encima. En una forma estricta de la arquitectura, solo se puede acceder a la capa que está exactamente debajo. Si se usa un acercamiento más flexible, como se puede observar en la Figura 3.5, una capa puede acceder a todas las capas por debajo de ella. [2]

La arquitectura de capas divide la aplicación con la intención de que cada una tenga un rol muy definido, no define la cantidad de capas que debe tener la aplicación y aplica el principio de separación de preocupaciones. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de tener que hacer algún cambio, solo se realiza en el nivel que le corresponda.

3.3.1. Modificaciones a la arquitectura de OLS

OpenLatino Server implementa *Clean Architecture*[5], la cual es una arquitectura por capas. En la figura 3.6 se muestra la Arquitectura de software *Clean Architecture* implementada en OLS. Se puede observar la separación de las capas y la dependencia que existe entre ellas.

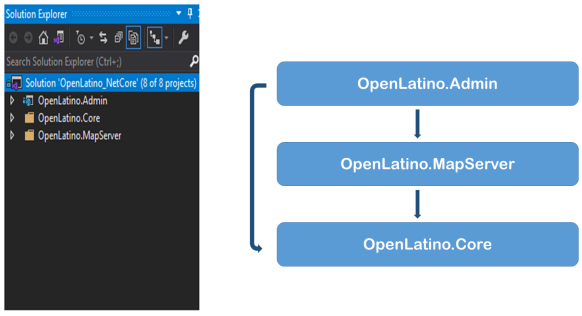

Figura 3.6: Arquitectura de software *Clean Architecture* implementada en OLS en su última versión.

En la última versión, OLS presenta tres capas:

• **OpenLatino.Core**. Es la capa base. No tiene dependencias externas. Posee las entidades y la declaración de las interfaces.

• **OpenLatino.MapServer.** Depende de la capa OpenLatino.Core. Se encarga de procesar los pedidos de las aplicaciones cliente y devolver un response. Posee la implementación de las funciones WMS.

• **OpenLatino.Admin**. Depende de las capas anteriores, posee la implementación de los controladores encargados de recibir las peticiones de las aplicaciones cliente.

En la nueva versión de OLS, con la implementación de la interfaz visual de configuración en Angular, se hizo necesario modificar la arquitectura, manteniendo los principios de *Clean Architecture*, garantizando la extensibilidad, seguridad y buenas prácticas del servidor. La nueva arquitectura de OLS se muestra en la figura 3.7.

Se eliminó la capa más externa de la arquitectura de OLS y, en su lugar, se implementaron tres nuevas capas:

• **OpenLatino.Admin.Infraestructure**. Consiste en una capa de abstracción entre la entre la capa de entidades de dominio y la capa de lógica de negocio. Implementa el patrón Repositorio [6], que consulta la fuente de datos, los asigna a una entidad y realiza cambios en dicha entidad a la fuente de datos.

[5]Arquitectura limpia en español. Presentada por Robert C. Martin (Uncle Bob) en su blog, el 13 de agosto de 2012 [5]

[6]Patrón de diseño que aísla la capa de datos del resto de la aplicación.

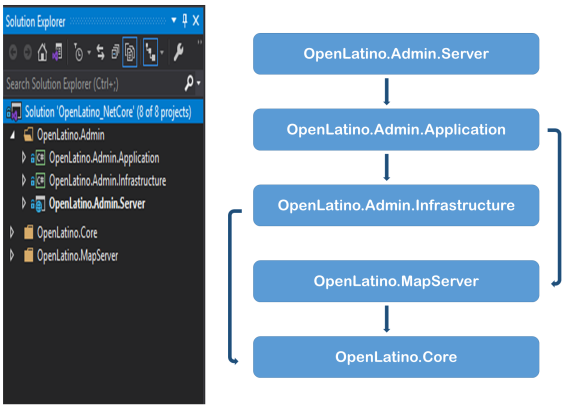

Figura 3.7: Arquitectura de software *Clean Architecture* de OLS modificada.

- **OpenLatino.Admin.Application**. Esta capa contiene interfaces que se utilizan para comunicarse entre la capa OpenLatino.Admin.Server y la capa de repositorio. Contiene los servicios asociados a una entidad específica, ya sean proveedores, temáticos, estilos, etc.
- **OpenLatino.Admin.Server** Es la capa más externa de la aplicación, contiene los controladores que reciben los pedidos de las aplicaciones clientes y de los usuarios de la interfaz visual de configuración. En esta capa se realiza, además, la verificación de seguridad, que valida la identidad de los usuarios y clientes que realizan los pedidos.

3.3.2. Arquitectura de la Interfaz Visual de Configuración.

Las aplicaciones creadas con Angular se separan en plantillas HTML, clases de componentes en TypeScript, escritas para gestionar dichas plantillas, y los servicios que contienen la lógica para la comunicación con el servidor.

Para crear la interfaz visual se generaron templates con HTML, controlando estos con lógica creada en componentes, que serán exportados como clases. Así mismo, se agregan los servicios para manejar el flujo de datos de la aplicación y el servidor OLS. Finalmente, se "encapsulan" dichas componentes y servicios en módulos.

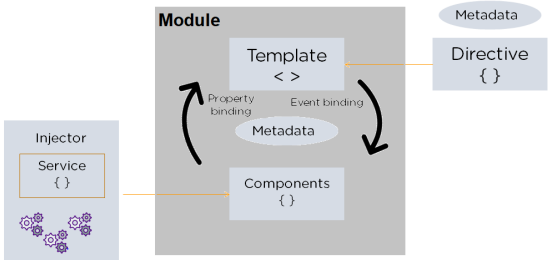

Figura 3.8: Arquitectura de software de la interfaz visual.

La arquitectura de la aplicación desarrollada se muestra en la figura 3.8. Con el objetivo de seguir las buenas prácticas de la programación, la interfaz visual implementa varios módulos cuyas funcionalidades están bien definidas e independientes. Estos gestionan el trabajo con temáticos, estilos, proveedores, información alfanumérica, auntenticación,

etcétera. Cada uno de dichos módulos contiene uno o varios servicios que se encargan de realizar los pedidos al servidor, atendiendo a las peticiones del usuario, y de recibir la respuesta de OLS. Estos servicios son inyectados en las componentes como dependencias.

Las componentes son clases que contienen la plantilla de la vista, y la lógica asociada a esta, de una parte de la interfaz visual. Para gestionar la configuración de OLS se crearon varias componentes, encargadas de crear nuevos temáticos, editar estilos, registrar usuarios, etcétera. En la interfaz visual, las plantillas HTML contienen directivas que le proveen lógica de programación, las cuales realizan *data binding*, es decir, enlazan la lógica de los datos de la aplicación con las vistas. Por otro lado, *event binding*, responde a la acción del usuario al interactuar en la aplicación, actualizando los datos en la lógica.

3.4. Detalles de implementación

En esta sección se explicarán los cambios en el código para la creación del nuevo tipo de mapa temático de categorías, explicando el proceso para crear nuevos tipos de temáticos. Se abordará, también, cómo fue implementada la seguridad de la interfaz visual y del servidor, no sin antes abordar en qué consiste el estándar de seguridad JWT (JSON Web Token por sus siglas en inglés).

3.4.1. Implementación de mapas temáticos

Para crear el nuevo tipo de mapa temático se eliminó la entidad `TematicQuery`, creando en su lugar la nueva entidad `TematicType` (figura 3.9), la cual contiene las propiedas inherentes a todo tipo de temático: Id, nombre, función, que constituye los filtros, y los estilos asociados. Se creó, además, la entidad `TematicLayer` que contiene el nombre y el Id del temático.

```
namespace OpenLatino.Core.Domain.Entities
{
    19 references | 0 changes | 0 authors, 0 changes
    public class TematicType
    {
        7 references | 0 changes | 0 authors, 0 changes | 0 exceptions
        public int Id { get; set; }
        [Display(Name = "TematicType Name")]
        14 references | 0 changes | 0 authors, 0 changes | 0 exceptions
        public string Name { get; set; }
        10 references | 0 changes | 0 authors, 0 changes | 0 exceptions
        public byte[] Function { get; set; }
        6 references | 0 changes | 0 authors, 0 changes | 0 exceptions
        public virtual ICollection<StyleConfig> StyleConfiguration { get; set; }
    }
}
```

Figura 3.9: Entidad que representa los tipos de temáticos en OLS.

Para crear nuevos temáticos se crea una nueva clase que herede de la nueva entidad `TematicType` (figura 3.10).

Al heredar de esta clase, automáticamente la clase queda relacionada con la

entidad `StyleConfig`, de esta manera no hay que modificar nada en la estructura existente: se crea la nueva entidad, se agrega algún campo extra que necesite tener y se genera una nueva migración porque se modifica la base de datos existente al agregar el nuevo tipo.

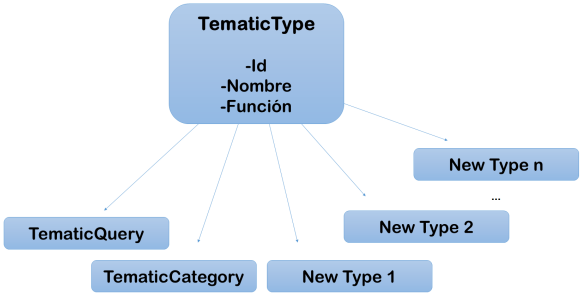

Figura 3.10: Estructura para crear nuevos tipos de mapas temáticos.

Se crea, además, una interfaz `ITematicLayerHelper`, en sustitución de la interfaz existente, que tiene los métodos que permiten el trabajo con mapas temáticos. El servicio `TematicLayerService` implementa esta interfaz.

Para el procesamiento de nuevos tipos de temáticos, se pueden seguir cualquiera de las siguientes vías, dependiendo de lo que se desea implementar:

- **ITematicLayerHelper → TematicLayerService**. Usar el mismo servicio implementado para los temáticos existentes.
- **ITematicLayerHelper → TematicLayerService → NewService**. Crear un nuevo servicio que use algunos métodos del servicio existente, redefina otros y cree algunos nuevos.
- **ITematicLayerHelper → NewService**. Crear un nuevo servicio que tenga una implementación diferente de los métodos actuales.
- **NewInterfaceHelper → NewService**. Crear una nueva interface, y un servicio que la implemente (para mantener la estructura de la aplicación) debido a que el nuevo temático usa métodos diferentes a los definidos en la existente.

La opción a escoger varía de acuerdo a las necesidades del programador, dependiendo de las funcionalidades del nuevo temático a implementar.

3.4.2. Seguridad de la interfaz visual y del servidor

En esta nueva versión de OLS es necesario implementar seguridad a nivel de Usuarios, para controlar las acciones que pueden realizar en la configuración del servidor desde la interfaz visual, y evitar pedidos de usuarios maliciosos. Igualmente se hace necesario implementar seguridad en los componentes de la interfaz visual, ya

que un usuario no autorizado no puede acceder a algunas vistas restringidas en la aplicación. Para lograr lo anterior se decide implementar Json Web Token (JWT).

Json Web Token (JWT) es un estándar para transmitir información de forma segura en internet, por medio de archivos en formato JSON[7]. Este sistema se utiliza en la autenticación de la interfaz visual, siendo su función principal la de validar la identidad de quien ingresa a la página, después de que ya haya iniciado sesión en el pasado (figura 3.11).

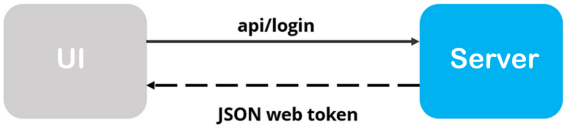

Figura 3.11: Login en OLS desde la interfaz visual.

El contenido del token se encuentra firmado digitalmente, por lo que se puede comprobar su veracidad. Gracias a esta certificación, el servidor web puede aprobar de forma segura todas las peticiones que se hagan desde esa sesión en nombre de ese usuario. Un JWT se divide en tres partes: el *header*, que contiene el tipo de token y la información del algoritmo de codificación[8] usado. El *Payload*, que contiene la información relativa al usuario, si es admin o usuario regular, cuando inició sesión, su Id en el sistema, etcétera.

```
var secretKey = new SymmetricSecurityKey(Encoding.UTF8
                    .GetBytes("super_SecretKEY.@@@$$$"));
var signinCredentials = new SigningCredentials(secretKey,
                            SecurityAlgorithms.HmacSha256);
var claims = new List<Claim>
    {
        new Claim(ClaimTypes.Name, user.UserName),
        new Claim(ClaimTypes.Role, role),
        new Claim(ClaimTypes.NameIdentifier, user.Id)
    };
var tokeOptions = new JwtSecurityToken(
    issuer: "https://localhost:5001",
    audience: "https://localhost:5001",
    claims: claims,
    signingCredentials: signinCredentials
);
return new JwtSecurityTokenHandler().WriteToken(tokeOptions);
```

Figura 3.12: JWT en OLS.

Finalmente, *Signature* contiene la firma digital del token, creada combinando el header y el payload, basado en una clave secreta que solo el servidor conoce. Los JWT se generan en OLS cuando un usuario inicia sesión en el sistema, la figura 3.12 muestra el fragmento de código donde se crea el token y se retorna a la vista. `JwtSecurityToken` contiene importantes parámetros: los dos primeros son para verificar que OLS es el que esta respondiendo los pedidos a la vista, la tercera propiedad contiene información acerca del usuario que inició sesión en el sistema; el último parámetro contiene la

[7]Tipo de archivo de texto plano con el cual se pueden crear parámetros y asignarles un valor

[8]Usualmente HS256, que genera una firma simétrica, esto quiere decir que el secret/key se usa tanto para el firmado como para la verificación de la firma.

firma digital.

Aún con todas las comprobaciones que se realizan, existe un problema grave de seguridad, ya que un agente externo puede interceptar los pedidos que viajan a OLS desde la interfaz visual y obtener un token que no es suyo, esto implica que puede usarlo para realizar pedidos al servidor como si fuera ese cliente y acceder a información confidencial sin ser detectado.

Para resolver este problema se le establece un tiempo de vida limitado para cada token, es decir, cada uno de los tokens que se les entrega a los clientes será válido solo por un tiempo prefijado. Este cambio garantiza que, si un token es robado, una vez que se agote su tiempo de vida, no servirá. En el caso de OLS, se añade la propiedad `expires: DateTime.Now.AddMinutes(30)` al JWT, lo que quiere decir que un usuario se mantendrá registrado en el sistema solo por media hora, pasado ese tiempo debe volver a ingresar mediante su contraseña.

```
@Injectable({
  providedIn: 'root'
})
export class AuthGuard implements CanActivate  {
  constructor(private router:Router,
              private jwtHelper: JwtHelperService,
              private toastr: ToastrService){}
  canActivate(route: ActivatedRouteSnapshot,
              state: RouterStateSnapshot) {
    const token = localStorage.getItem("jwt");
    if (token && !this.jwtHelper.isTokenExpired(token)){
      return true;
    }

    if (token && this.jwtHelper.isTokenExpired(token)){
      this.toastr.info('Login sesion expired');
    }

    this.router.navigate([global['routeTitlePage']]);
    return false; }}
```

Figura 3.13: Seguridad en la interfaz visual.

Del lado del FrontEnd también se hacen verificaciones de seguridad, mediante el JWT enviado desde el servidor. Se realizan validaciones, permintiendo a los usuarios acceder a las vistas de configuración, según sus roles. Un usuario que no está logeado no puede acceder a ninguna vista de configuración, además, si no es administrador no puede acceder a todas las vistas. En el `app-routing-module`, mediante la propiedad `canActivate`, se reciben, por inyección de dependencia, las clases que verifican, de acuerdo al JWT, si el usuario tiene acceso a la ruta solicitada. En la figura 3.13 se muestra la clase encargada de verificar si un usuario esta logeado y si su sesión no ha expirado.

Finalmente, se incluyen los los JWT en los pedidos del FrontEnd a OLS en el

header de Autorización[9].

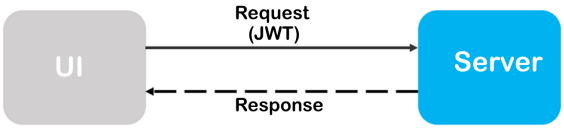

Figura 3.14: Pedido de la interfaz visual a OLS.

Esto se hace con el objetivo de que el servidor valide que el pedido que se está realizando proviene de la interfaz visual de configuración y no de un agente externo. (figura 3.14)

[9]Authorization header, en inglés, contiene las credenciales para autenticar a un usuario en un servidor

Capítulo 4

Pruebas de Funcionalidad

En este capítulo se realizan un conjunto de pruebas de correctitud y optimización para demostrar el buen funcionamiento de las herramientas implementadas y así evidenciar el cumplimiento de los objetivos planteados en el trabajo.

Las pruebas se realizaron en una laptop con las siguientes características:

- Sistema operativo Windows 10 Enterprise LTSC, 64 bits.
- Procesador Intel(R) Core(TM) i3-6100U @ 2.30GHz 2.30GHz.
- Memoria RAM: 8GB (7.82 usable)
- Disco duro 250GB, HDD.

Se cuenta con la cartografía de GeoCuba para las pruebas, las cuales iniciaron con los siguentes datos guardados en la base de datos de Admin:

- Capa **Roads**(Calles), color rojo definido por defecto, con información alfanumérica asociada de cinco columnas: *maxspeed*, *oneway*, *bridge*, *type*, *name*.
- Un usuario con rol de Administrador, *adminopenlatino@gmail.com*.
- Workspace *AdminWorkspace*, asociado al usuario anterior, con acceso a las funciones *GetCapabilities*, *GetMap* y *GetTematicMap*.
- Estilos *RedStyle*, *BlueStyle*, *GreenStyle* y *YellowStyle* que representan los colores, rojo, azul, verde y amarillo respectivamente.

Para la validación de las pruebas se usará Postman, verificando mediante este que el sistema está funcionando correctamente. Postman es una aplicación que permite realizar pruebas API. Es un cliente HTTP[1] que da la posibilidad de realizar *HTTP requests* a través de una interfaz gráfica de usuario, por medio de la cual se obtiene la respuesta del servidor.

4.1. Configuración de temáticos por categorías

El objetivo de esta prueba es crear un nuevo temático en OLS. Se trabajará sobre la capa **Roads**, escogiendo la columna *oneway*[2] como campo alfanumérico. La figura 4.1 muestra la vista de creación con los datos anteriores.

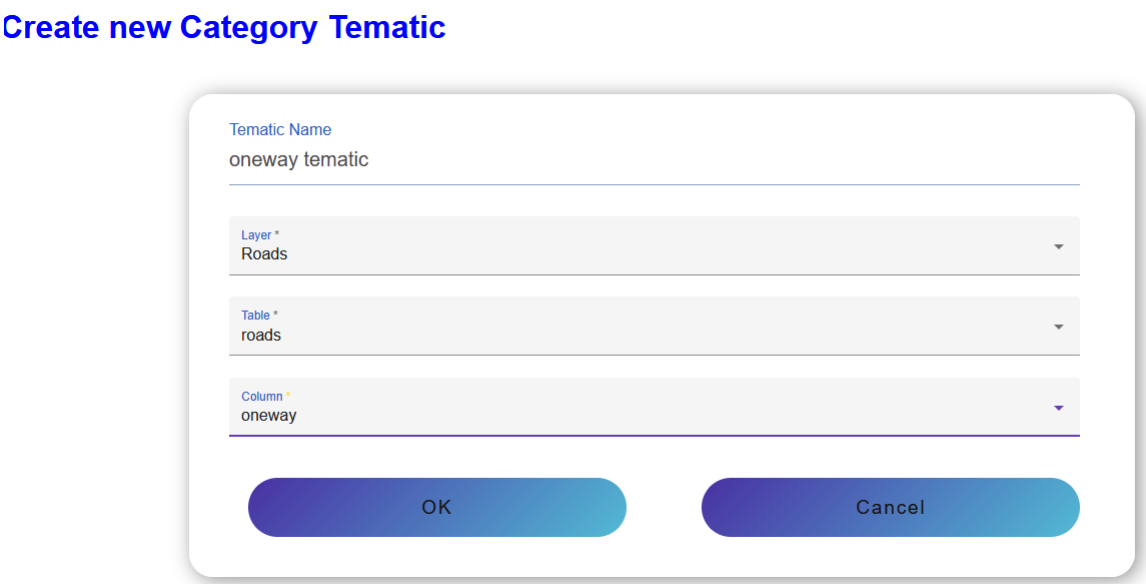

Figura 4.1: Vista de creación de temáticos de categorías.

Se espera que se genere una configuración de temático donde se tienen dos estilos para dos categorías. Efectivamente, se comprueba que dando click en el botón **OK**, se genera la configuración del temático automáticamente, observándose en la figura

[1]HTTP, de sus siglas en inglés: Hypertext Transfer Protocol, es el nombre de un protocolo que permite realizar una petición de datos y recursos.

[2]Según la cartografía de GeoCuba, este campo tiene valor 1 si la calle es de un solo sentido y 0 en caso contrario

4.2 las dos posibles categorías de la columna seleccionada, y la asignación de estilos. Si no hay suficientes estilos para cada categoría el usuario puede generarlos desde esta vista. El usuario también tiene la posibilidad de cambiar los estilos asignados por otros o crear uno nuevo.

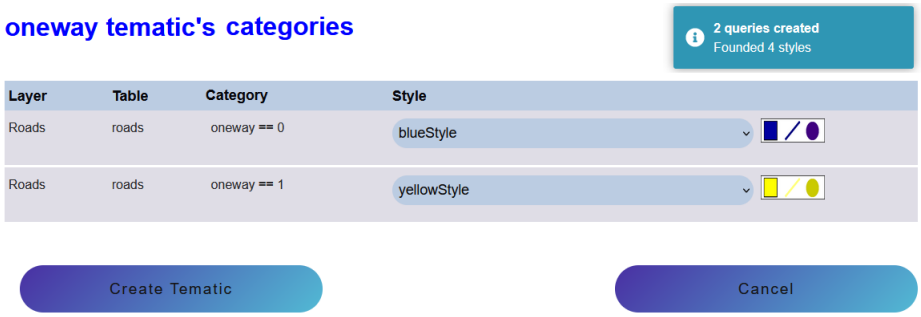

Figura 4.2: Configuración generada a partir de los datos de la vista de creación.

Si se da click sobre el botón **Create Tematic**, se genera el nuevo temático como se observa en la figura 4.3. La cual muestra una lista de todos los temáticos de ese tipo creados, en este caso, solo existe el temático de nombre *oneway tematic*. Además, se observa la información asociada a este: capa, categorías, estilos, etc.

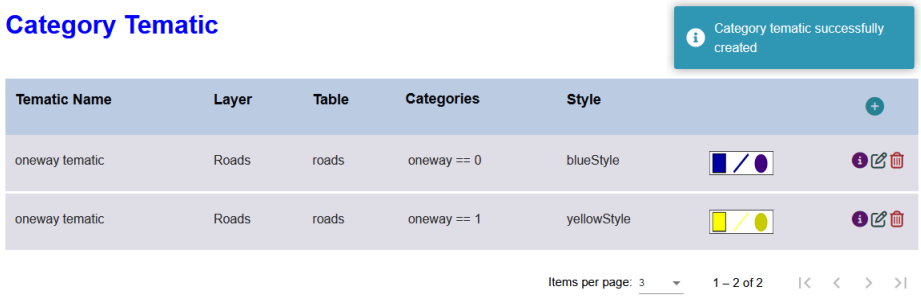

Figura 4.3: Listado de mapas temáticos creados en OLS.

4.2. Generación de mapas temáticos por categorías.

Mediante postman se verifica que el temático creado en la sección anterior se pinta bien sobre el mapa. Se espera que al realizar el pedido, se coloreen las calles

de un solo sentido de azul y las de dos sentidos de amarillo. Si existe alguna calle que no tenga definido sentido en la cartografía, debido a que este valor es NULL, se pinta del valor por defecto de las calles, en este caso, rojo.

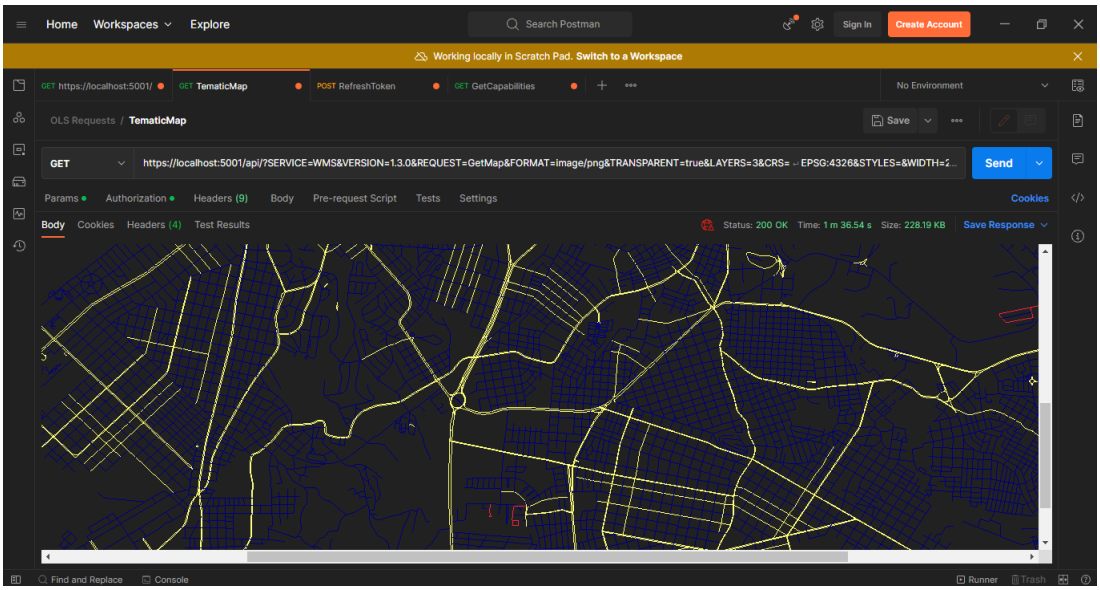

Figura 4.4: Pedido *GetTematicMap*.

La figura 4.4 muestra la respuesta del servidor para el temático recién creado. Se observan los resultados esperados. Ahora se edita el temático, para en lugar de pintar las calles de amarillo, lo haga de verde (figura 4.5).

Figura 4.5: Edición de temáticos.

Si se realiza nuevamente el pedido se observa que ahora las calles de doble sentido están coloreadas de verde.(figura 4.6)

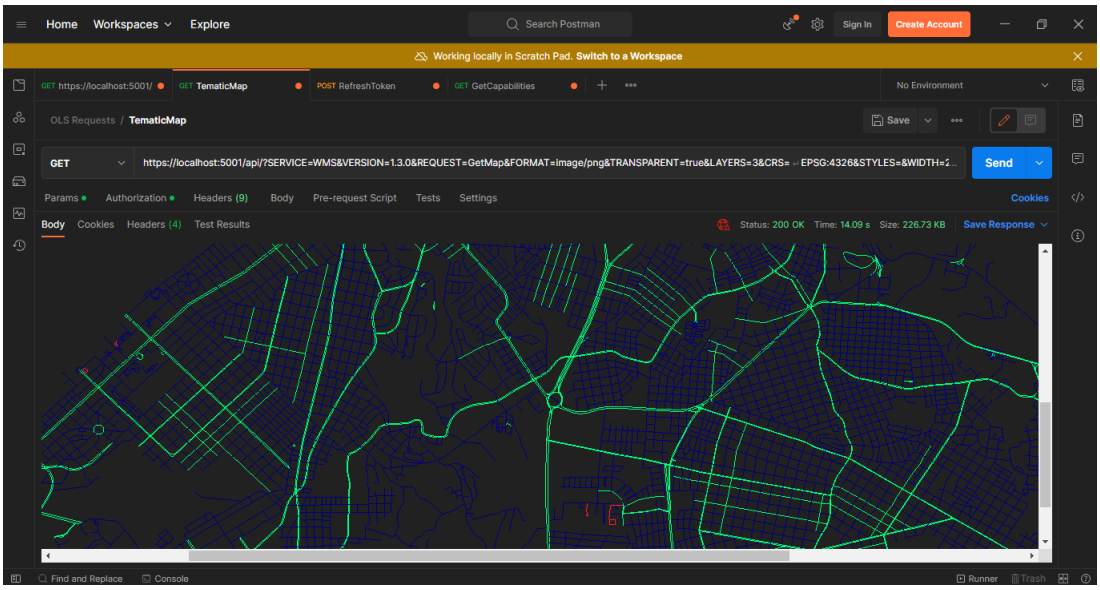

Figura 4.6: Pedido *GetTematicMap* a temático editado.

4.3. Comprobaciones a la interfaz visual

En esta prueba se verifica que la interfaz visual está funcinando correctamente mediante operaciones CRUD. Para ello se trabaja con la entidad *estilos*, y funciona de manera análoga para el resto de las entidades. Primeramente, el estilo de color verde, se editará y pasará a ser de color celeste, eliminándolo finalmente del servidor.

La figura 4.7 muestra la vista de edición, y los listados de estilo después de editado. Se observan los resultados esperados. Finalmente, se eliminará este estilo del sistema. La figura 4.8 muestra que esta operación se realizó satisfactoriamente.

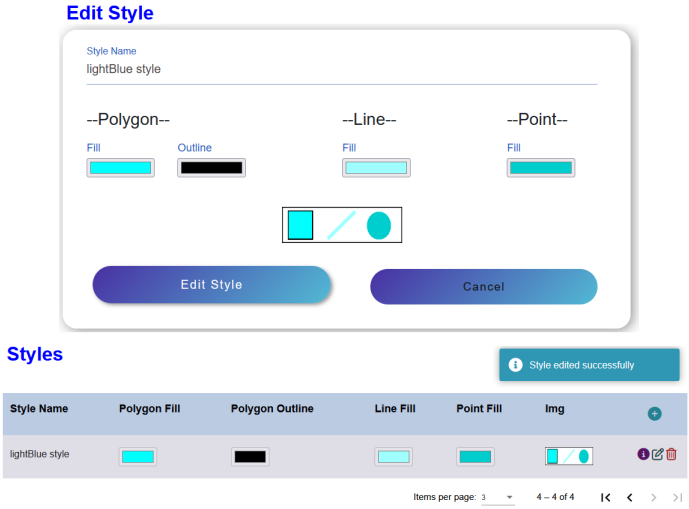

Figura 4.7: Edición de estilos.

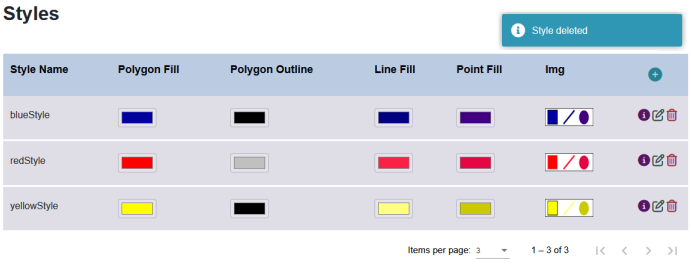

Figura 4.8: Eliminación de estilos.

Capítulo 5

Conclusiones, recomendaciones y trabajo futuro

La implementación de las funcionalidades expuestas en capítulos anteriores da solución a las deficiencias planteadas de OLS, por lo tanto, se cumplieron los objetivos propuestos.

En este capítulo se exponen las conclusiones a las que se arribaron con el desarrollo de la aplicación y se proponen, además, algunas recomendaciones para el trabajo futuro con OLS.

5.1. Conclusiones

El presente trabajo de diploma finaliza con la creación de la nueva interfaz visual de OLS, intuitiva, agradable y segura. Para lograrlo, se realizaron modificaciones sobre el sistema, en su arquitectura, modelo de datos y desarrollo de nuevas funcionalidades, con una implementación extensible y reusable, manteniendo las buenas prácticas que presenta el servidor. Se creó, además, un nuevo tipo de temático, por clasificación de tipos o por categorías, que le permite al usuario la creación de mapas temáticos de forma sencilla, ya que todo el procesamiento de los datos y asignación de estilos se realiza de forma automática, extendiendo, para lograr este propósito, el *Render*(clase encargada de pintar los mapas) de OLS, para que aceptara filtros genéricos, en lugar de consultas sql. Por último, se modificó el pedido *GetCapabili-*

ties para que este muestre también la información acerca de los temáticos existentes, también se creó un nuevo pedido *GetTematicCapabilities*, que devuelve solo la información de los mapas temáticos creados.

Estos cambios mencionados dotan a OLS de mejoras y nuevas funcionalidades, por lo que se puede concluir que los principales aportes de esta investigación son:

- Modificación del pedido *GetCapabilities* para que devuelva información acerca de mapas temáticos y creación de un nuevo pedido que devuelve solo esa información.
- Implementación de una nueva interfaz visual, mediante la cual se puede modificar la configuración de OLS de forma intuitiva.
- Creación de un nuevo tipo de mapa temático que realiza el procesamiento de capas, categorías asociadas y estilos automáticamente.
- Modificación de la arquitectura de OLS para acoplar la nueva interfaz visual, implementada en Angular, y la seguridad de los pedidos de los usuarios.
- Cambio del modelo de datos para la implementación del nuevo temático y hacer extensible el proceso de agregar nuevos tipos.
- Nueva implementación del Render para que acepte filtros genéricos (varias condiciones separadas por operadores lógicos) en lugar de consultas sql.

5.2. Recomendaciones y trabajo futuro

Aunque se cumplió con todos los objetivos planteados al inicio del trabajo, el software implementado puede mejorarse y se pueden agregar nuevas funcionalidades, por ello, se propone:

- Mejorar la arquitectura usada en la interfaz visual. Para su desarrollo se usó la arquitectura propia del framework, sin embargo, puede implementarse arquitectura de capas y patrón MVC.

- Implementar nuevos tipos de temáticos que refuercen las funcionalidades de OLS como servidor de mapas.
- Incorporar técnicas de visualización científica que permita revelar patrones insospechados en los datos de la cartografía.

Bibliografía

[1] SharpMap (2008). *SharpMap: Geospatial Application Framework for the CLR*. URL http://www.codeplex.com/SharpMap.

[2] Leonard J. Bass. *Software Architecture in Practice*. 2012.

[3] Bipin Joshi Beginning. *SOLID Principles and Design Patterns for ASP.NET Developers*. 2016.

[4] Oscar Blancarte. *Introducción a la arquitectura de software*. 2014.

[5] Clean Coder Blog. Consultado el 15 de octubre de 2022. URL https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.

[6] Sitio Oficial de Angular. Consultado el 8 de agosto de 2022. URL https://angular.io/.

[7] QGIS El SIG Lder de Código Abierto para Escritorio. Consultado el 1 de agosto de 2022. URL https://www.qgis.org/es/site/about/index.html.

[8] Sitio Oficial de React. Consultado el 7 de agosto de 2022. URL https://es.reactjs.org/.

[9] Sitio Oficial de Vue. Consultado el 8 de agosto de 2022. URL https://es.vuejs.org/.

[10] Sitio Oficial de Windows. Consultado el 16 de junio de 2022. URL https://www.iis.net/.

[11] Gustavo D. Buzai / Claudia A. Baxendale / María del Rosario Cruz. *Fases de un proyecto de Investigación en estudios de Geografía Aplicada basados en el uso de Sistemas de Información Geográfica*. 2000.

[12] Qué es un mapa. Consultado el 15 de junio de 2022. URL https://www.significados.com/mapa/.

[13] Roger Frank. *The application of electronic computing methods and techniques to the storage, compilation, and assessment of mapped data*. 1974.

[14] Alexander Gillis. *What is internet of things*. 2021.

[15] Web GIS y Open Source GIS Servers. Consultado el 16 de junio de 2022. URL https://www.igismap.com/web-gis-application-development-and-gis-servers/.

[16] Jamie Kurtz. *ASP.NET MVC4 and the Web API*. 2013.

[17] José Eduardo Zalacain LLanes. *Módulo SIG para la generación y consulta de mapas temáticos en la web*. 2011.

[18] Dustin Metzgar. *NET Core in Action*. 2018.

[19] Bertrand Meyer. *Object-Oriented Software Construction*. 1988.

[20] Paula Andrea Fernández / David Alejandro Muoz. *Implementación de los Sistemas de Información Geográfica SIG para la elaboración de mapas temáticos ambientales en el Municipio de Santander de Quilichao*. 2019.

[21] Ferjan Ormeling. *Mapas temáticos*. 1996.

[22] José Miguel Santos Preciado. *Sistemas de Información Geográfica*. 2004.

[23] Geomapik Official Site. Consultado el 16 de junio de 2022. URL https://geomapik.com/webmapping-gis/estandaresogc-wms-wmtswfs-wcs/.

[24] Mapas temáticos. Consultado el 16 de junio de 2022. URL http://pdi.topografia.upm.es/mab/tematica/htmls/inicial.html.

[25] Allexander Rodríguez Torres. *Servicio de Mapas Temáticos*. 2010.

[26] José Ricardo Manzo Yanangomez. *Los mapas temáticos y su apliación como estrategia didáctica para la ense nanza de la Geografía Física en el nivel escolar*. 2020.