Grado en Ingeniería Informática

Trabajo Final de Grado

Herramienta de documentación de infraestructuras informáticas

Autor:
Daniel Varela Bellot

Supervisor:
Alfredo Cadroy Esteban
Tutor académico:
Sergio Barrachina Mir

Fecha de lectura: 18 de Julio de 2017
Curso académico 2016/2017

Resumen

En el presente documento se presenta el desarrollo de una herramienta de documentación de infraestructuras informáticas que permite registrar, modificar y consultar información referente a los elementos típicos de una red, tales como switches, servidores, conexiones, etc., de una forma gráfica. El desarrollo de esta herramienta tiene el objetivo de centralizar y homogeneizar la forma en la cual se lleva a cabo la documentación de las redes de los clientes de la empresa donde se ha realizado la estancia en prácticas. Para ello, se ha implementado una aplicación web que se apoya en una tecnología de gestión de direcciones IP, denominada Netbox, donde se almacena el modelo de datos. La interfaz se comunica con esta tecnología mediante llamadas asíncronas y representa toda la información a través de diagramas. De este modo, el usuario puede realizar acciones sobre el modelo de datos mediante la interacción con los diagramas.

Palabras clave

IPAM, aplicación web, diagrama, infraestructura informática, documentación.

Keywords

IPAM, web application, diagram, computer infrastructure, documentation.

Índice general

Capítulo 1

Introducción

Contenidos

1.1. Contexto y motivación del proyecto

En el presente apartado se define el proyecto y su contexto. En primer lugar, con el objetivo de entender mejor este contexto, se describe la empresa que ha planteado el proyecto y las motivaciones que han propiciado tal decisión. Una vez planteado el contexto, se explica en qué consiste el proyecto y se enumeran los objetivos que debe cumplir.

1.1.1. Descripción de la empresa

El proyecto se ha desarrollado en la empresa de base tecnológica 720Tec, cuyo logotipo se puede apreciar en la Figura 1.1. La empresa empezó su actividad en Castellón en diciembre de 2014. A pesar de ser una empresa muy joven, está respaldada por un equipo de profesionales con muchos años de experiencia en el sector tecnológico.

Sus principales líneas de negocio se basan en ofrecer servicios tecnológicos, *outsourcing*, consultoría estratégica y diseño organizativo a otras organizaciones. Básicamente, su función consiste en el análisis y utilización de la tecnología y ver como ésta se puede adaptar, de una forma práctica y eficiente, a cada uno de sus clientes para la consecución de objetivos estratégicos y ventajas competitivas. Dicha empresa pone mucho énfasis en darse a conocer por la calidad de sus productos y servicios, su orientación hacia la investigación e innovación y por la excelencia y valores de sus empleados. Es por ello que su modelo de gestión está basado en ITIL, compuesto por un conjunto de buenas prácticas y recomendaciones sobre cómo llevar a cabo la gestión de las TIC. Este tipo de gestión supone una garantía de calidad de servicio, ajustado a unas

Figura 1.1: Logotipo de 720Tec

(Fuente: http://www.720tec.es/)

necesidades y presupuestos, para las organizaciones. Los productos y servicios que ofrece 720Tec para estas organizaciones se podrían estructurar en los cuatro bloques indicados a continuación.

- **Servicios gestionados.** Servicios avanzados de mantenimiento y soporte de infraestructuras, de carácter proactivo, de forma que se solucionen problemas actuales como la itinerancia, dispersión y ubicuidad del puesto de trabajo.
- **Soluciones en infraestructuras para entornos industriales.** Tecnologías y servicios orientados al entorno industrial, donde la robustez, el coste de operación y propiedad adquieren más importancia.
- **Diseño, provisión y mantenimiento de infraestructuras de comunicaciones y colaboración avanzadas.** Despliegues rápidos de infraestructuras a un coste reducido para hacerlo accesible a las pequeñas y medianas empresas. Su objetivo es permitir a las pymes acceder a modelos de gestión, de base tecnológica, que utilizan las grandes corporaciones.
- **Desarrollo de proyectos especiales.** Uso de la tecnología para la creación de productos tecnológicos que se adapten a las necesidades de los clientes y que permitan obtener ventajas competitivas derivadas de la diferenciación e innovación.

1.1.2. Descripción del proyecto

Este proyecto consiste en el desarrollo de una herramienta de documentación de infraestructuras informáticas que permita registrar, modificar y consultar información de los elementos de una red, de una forma gráfica.

Esta herramienta hace uso de una tecnología de gestión de direcciones IP denominada Netbox, la cual se ha instalado en la parte del servidor. Netbox proporciona una API REST para poder realizar operaciones sobre la base de datos a partir de llamadas HTTP. Por otra parte, la interfaz de la aplicación se ha implementado a través de tecnologías web como JavaScript y hace uso de bibliotecas de dibujos de diagramas con el fin de poder representar toda la infraestructura de una forma gráfica.

La herramienta permite presentar de una forma visual, mediante diagramas, cada elemento de la red de cada uno de los clientes de la empresa. Aparte, proporciona la posibilidad de

documentar y consultar información sobre los dispositivos de red. Los usuarios de la herramienta pueden editar los diagramas, introduciendo, eliminando o modificando los vértices (dispositivos de red) y/o las aristas (conexiones), de forma que todos los cambios realizados quedan reflejados en la base de datos.

Con ello, el sistema gestiona todos los eventos producidos por los diferentes elementos, por ejemplo, al generar un nuevo elemento en el diagrama, la aplicación capta el evento, lanza una llamada asíncrona a la API y almacena los nuevos datos. Lo mismo ocurre para modificaciones o borrado. Además, también permite almacenar y recuperar el diagrama, entre la interfaz y el *backend*, codificándolo en formato XML. Además, la aplicación se encarga de gestionar las credenciales, de forma segura, para poder establecer conexiones remotas desde algunos puntos de acceso.

Una herramienta como ésta proporcionará muchas ventajas al equipo de soporte de 720Tec. Por una parte, permitirá centralizar y homogeneizar la documentación de las infraestructuras de red de sus clientes. Además, facilitará el proceso de establecer conexiones remotas, pues ya no será necesario buscar las credenciales manualmente, el sistema lo realizará de forma automática. Todo esto se verá traducido en una reducción del tiempo necesario para realizar las funciones típicas de soporte. Aparte, al homogeneizar la forma de documentar, se mejorará la comprensión de los diagramas por parte de los integrantes del equipo. Todas estas reducciones de tiempo a la hora de dar soporte diario al cliente se resumirán en un aumento de la calidad de servicio proporcionada por la empresa.

1.2. Objetivos del proyecto

El objetivo principal del proyecto consiste en facilitar y agilizar el proceso de soporte técnico de la empresa, de forma que los integrantes del equipo puedan aumentar su productividad y eficiencia a la hora de realizar sus respectivas funciones. Este objetivo principal se podría desglosar en los subobjetivos enumerados a continuación.

1. **Homogeneizar la documentación de infraestructuras en la organización.** Se pretende estandarizar la forma en la cuál se representan y se documentan las infraestructuras informáticas por parte de los diferentes miembros del equipo.
2. **Centralizar la documentación en un solo servicio.** Toda la documentación se centralizará en un único punto, accesible a través de un cliente web desde la red interna de la empresa.
3. **Automatizar y agilizar tareas que actualmente se realizan de forma manual.** Automatización de las conexiones remotas, establecimiento de VPN y búsqueda de credenciales.
4. **Aumentar la seguridad, consistencia e integridad de los datos.** Existencia de un backend que respalda la gestión de direccionamiento IP y redes.
5. **Representar las infraestructuras informáticas de una forma más realista y desde diferentes perspectivas.** Diferentes formas de visualización del diagrama en base a las necesidades del usuario en un momento determinado.

1.3. Estructura de la memoria

La presente memoria técnica está dividida en 6 capítulos: Introducción, Descripción del proyecto, Planificación del proyecto, Análisis y diseño del sistema, Detalles de implementación y Conclusiones.

A lo largo de esta introducción se ha pretendido ofrecer una visión global del contexto en el cual se ha desarrollado el proyecto. Incluyendo una descripción de la empresa, del proyecto y de los objetivos que se pretenden conseguir con su materialización.

En el segundo capitulo se describe más detalladamente las características del proyecto. Para ello, se define su alcance, tanto funcional como organizacional, y las principales tecnologías que se han utilizado para su desarrollo.

En el tercer capítulo se define la planificación inicial del proyecto. Esta planificación incluye la lista de tareas planteadas en la etapa inicial del proyecto, reflejando su coste temporal mediante un diagrama de Gantt. Además, se presenta el coste económico que supone para la empresa llevar a cabo el desarrollo de la aplicación.

En el cuarto capítulo se define el análisis y diseño del sistema sobre el cual se apoya la aplicación. Para ello se muestra el análisis de requisitos del sistema necesarios para que éste pueda cubrir todos los objetivos planteados. Además, se explica su arquitectura y se presentan diferentes prototipos de la interfaz de usuario que se realizaron al inicio del proyecto.

En el quinto capítulo se explica cómo se han implementado algunos de los aspectos más importantes del sistema. Básicamente, consiste en un capítulo donde se detalla la lógica más importante que hay detrás de la aplicación.

Finalmente, en el último capítulo se plantean las impresiones y conclusiones que han surgido tras la realización del proyecto. Además, se exponen qué cambios se podrían realizar de cara al futuro con el objetivo de mejorar la calidad y la funcionalidad de la aplicación.

Capítulo 2

Descripción del proyecto

Contenidos

En este capitulo se realiza una descripción más detallada del proyecto. Por una parte, se estudia el alcance, tanto organizacional como funcional, de la herramienta. Por otra parte, se analizan las diferentes tecnologías utilizadas por la aplicación y se describen los motivos por los cuales han sido seleccionadas.

2.1. Alcance del proyecto

El alcance organizacional del sistema incluye a todo el equipo de soporte de 720Tec. A pesar de basarse en una aplicación web, esta herramienta será únicamente accesible por los empleados de la empresa, por lo que no va a estar diseñada como un servicio que deba soportar grandes cantidades de conexiones concurrentes.

Respecto al alcance funcional esbozado al inicio del proyecto, se podría definir de la forma en la que se muestra en los siguientes subapartados.

2.1.1. Gestión de usuarios

A pesar de que el servicio esté pensado para ser utilizado únicamente dentro del ámbito de la empresa, es necesario asegurarse de que los usuarios que acceden están debidamente autorizados. Netbox permite autenticación LDAP y a través de Active Directory, por lo cual solo podrán acceder los usuarios que se encuentren autenticados en el directorio activo de la empresa.

2.1.2. Gestión de clientes

El sistema debe almacenar información referente a cada uno de los clientes de la empresa con el fin de poder asociarlos con sus respectivos diagramas. De esta forma, los usuarios que utilicen el servicio deben poder realizar acciones típicas como insertar, modificar o borrar clientes en el sistema.

2.1.3. Gestión de elementos de red

El servicio debe permitir a los usuarios crear, modificar y borrar información referente a los elementos de red que se encuentren dentro de la infraestructura del cliente. Esta información se debe almacenar en la base de datos de Netbox mediante llamadas a la API.

Estos elementos pueden representar cualquier parte o dispositivo dentro de la red, es decir, puede tratarse de subredes, conmutadores, servidores, zonas, etc. El usuario debe poder definir tipos de dispositivos y asociarlos con los distintos elementos. La interacción con éstos se debe realizar a través del diagrama.

Los elementos deben poder conectarse entre sí mediante las aristas del diagrama. Estos enlaces representan las conexiones físicas o lógicas que poseen los elementos dentro de la infraestructura. Por ejemplo, las conexiones entre las interfaces de un switch y las interfaces de red de un equipo se representan mediante estos enlaces. Por otra parte, el usuario debe ser capaz de documentar cada uno de los elementos de la red y modificar su información a través de la interfaz.

2.1.4. Visualización de la red

La herramienta deber ser capaz de mostrar visualmente, mediante diagramas, la infraestructura de red de un cliente. Estos diagramas deben ser interactivos, de forma que el usuario pueda realizar acciones sobre sus elementos. Los tipos de elementos que puede contener un diagrama deben estar predefinidos por el sistema (routers, switches, hosts, etc.) y sus características deben poderse definir con más detalle por parte del usuario.

Además, la herramienta debe permitir mover y modificar los diferentes elementos del diagrama y éste debe ser almacenado en el servidor utilizando la codificación XML, para poder ser recuperado posteriormente. El sistema debe almacenar el diagrama en el servidor en forma de plantilla, de modo que no debe contener información de configuración, solamente aspectos relacionados con la visualización. Cuando el diagrama es recuperado en el cliente, la plantilla se debe rellenar con la información que contiene la base de datos de Netbox.

El servicio también debe permitir una visualización de los elementos del diagrama regido por un sistema de capas. Debe existir una capa por defecto en la cual se definen todos los elementos del diagrama, pero posteriormente el usuario debe ser capaz de definir sus capas y asignar cada elemento a éstas. De esta forma, la visualización del diagrama se debe poder filtrar, mostrando solamente los elementos incluidos en la capa seleccionada.

Además, el sistema también debe mostrar la información asociada a los diferentes elementos de la red mediante un sistema de niveles o capas. Por ejemplo, en el diagrama se debe poder visualizar información básica como el tipo, nombre y dirección IP del dispositivo, si se pulsa encima de éste se debe mostrar información más detallada y si se entra en los detalles del dispositivo, se visualizará toda su información. El sistema también debe permitir fijar el foco en algún dispositivo u ocultar elementos de niveles inferiores en la red.

2.1.5. Conexiones remotas

Finalmente, el servicio debe permitir a los usuarios realizar conexiones remotas a algunos equipos de los clientes. El tipo de conexión puede variar según el tipo de dispositivo con el que se quiere establecer la conexión. Por ejemplo, una conexión ssh para un sistema basado en Linux, o un escritorio remoto para una máquina Windows. Aparte, debe permitir establecer VPN por algunos puntos de acceso de los clientes. En este caso, para establecer la conexión se debe ejecutar un cliente residente en la máquina local. Respecto a las credenciales, el sistema debe almacenarlas en el servidor, de forma cifrada, y deben ser transmitidas al cliente sin dejar rastro en la máquina local ni desvelando ninguna información sensible. Además, los diferentes puntos de acceso deben estar resaltados o diferenciados de alguna forma para ser fácilmente identificables por parte de los usuarios del servicio.

2.2. Tecnologías utilizadas en la interfaz

En este apartado se describen las principales tecnologías que se han utilizado para desarrollar la interfaz web del proyecto, explicando los motivos por los cuales han sido seleccionadas.

2.2.1. Vue.js

Vue.js, cuyo logotipo se muestra en la Figura 2.1, es un «framework progresivo» que se utiliza para el desarrollo de interfaces de usuario. Con Vue.js se puede generar, fácilmente, vistas reactivas a las operaciones del usuario. Esto significa que la vista puede referenciar datos de un modelo local. Cuando se produce un cambio en el modelo de datos, éste se ve reflejado en todos los lugares de la vista donde se referencia [14]. De esta forma, se permite mantener un almacenamiento local de los datos utilizados por la aplicación.

Vue.js proporciona, aparte de la reactividad, muchas otras funciones útiles para la interfaz de usuario, como transiciones, control de eventos o reutilización de componentes. Así se facilita notablemente la implementación de código por parte de los desarrolladores web. Aún así, una de las características que se ha tenido más en cuenta para elegir Vue.js es su gran flexibilidad, pues a diferencia de otros *frameworks* su núcleo se centra únicamente en la vista, haciéndolo compatible con cualquier otra biblioteca. De esta forma, se puede evitar problemas de compatibilidad a la hora de integrarlo con mxGraph, la biblioteca que se ha utilizado para dibujar los diagramas.

Además, también hay que tener en cuenta su gran ligereza y rendimiento, pues Vue.js ha

Figura 2.1: Logotipo de Vue.js

(Fuente: https://vuejs.org/)

desmostrado ser, en diferentes tests de rendimiento, uno de los *frameworks* más rápidos [9]. Es por ello que gracias a su flexibilidad, ligereza, funcionalidad y fácil aprendizaje, Vue.js es una solución muy óptima para el desarrollo de la aplicación. De hecho, según un análisis, de JavaScript Rising Stars, véase la Figura 2.2, Vue.js es la biblioteca de JavaScript que más se utilizó en proyectos de desarrollo web durante el año pasado [11].

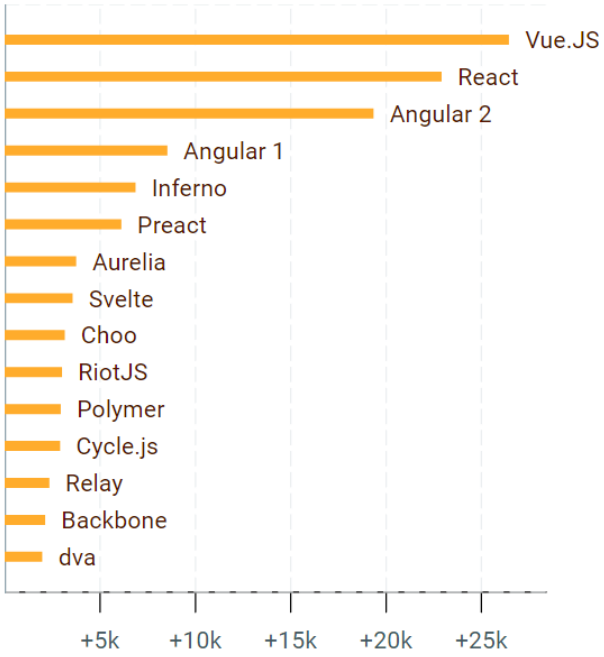

Figura 2.2: Cantidad de proyectos en los cuales han sido utilizados los *frameworks* JavaScript más conocidos durante el 2016

(Fuente: https://risingstars2016.js.org/#framework)

2.2.2. Axios

Axios es una biblioteca JavaScript basada en promesas que se utiliza para realizar llamadas HTTP asíncronas desde un cliente [13]. La ventaja de Axios es que permite realizar llamadas

asíncronas al servidor con una sintaxis muy sencilla. Además, es totalmente independiente de otras bibliotecas, por lo tanto, los componentes que integren Axios son totalmente reutilizables.

Esta biblioteca se ha utilizado para realizar todas las llamadas necesarias para consultar, eliminar y modificar datos en Netbox de una forma asíncrona.

2.2.3. Bootstrap

Bootstrap es un *framework* desarrollado por Twitter orientado a facilitar la implementación del diseño web. Su principal ventaja es que permite crear interfaces web adaptables de una forma sencilla, tal como se muestra en la Figura 2.3. Esto significa que los elementos de la interfaz cambian de tamaño automáticamente según el tipo de dispositivo que se esté utilizando para visualizarlos.

Bootstrap ayuda a construir páginas web dotándolas de un estilo profesional. Además, incorpora un sistema de rejilla que permite situar correctamente los diferentes componentes de la interfaz. Todo esto se consigue gracias a un conjunto de bibliotecas CSS y JavaScript que facilitan la tarea del desarrollador y proporcionan un tiempo de respuesta óptimo.

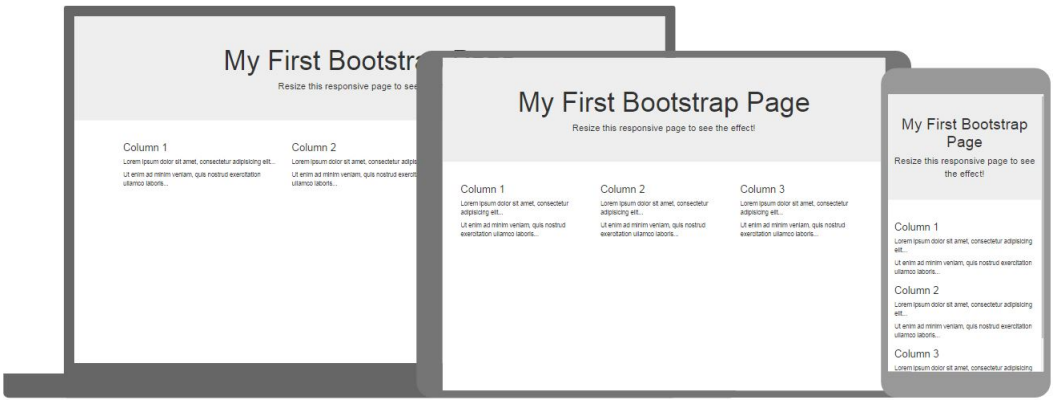

Figura 2.3: Diseño adaptable de Bootstrap

(Fuente: https://www.w3schools.com/bootstrap/)

Por otra parte, este *framework* también incluye multitud de componentes y *pluggins*, como desplegables, galerías de fotos, ventanas, etc., que pueden encontrarse frecuentemente en cualquier sitio web profesional. A todo esto, cabe sumarle la gran comunidad que existe detrás de Bootstrap, pues al ser código abierto y al ser muy utilizado, se puede encontrar solución para cualquier tipo de problema.

Básicamente, se ha decidido integrar Bootstrap en el proyecto para dotar a la aplicación de un estilo visual correcto, así como mejorar la usabilidad. De esta forma, se ha reducido en gran medida el tiempo de desarrollo dedicado al estilo de la interfaz web.

2.2.4. JQuery

JQuery es una biblioteca de JavaScript creada con el objetivo de facilitar la programación web orientada a la manipulación de elementos HTML. Gracias a esta biblioteca se puede manipular, fácilmente, elementos del DOM, manejar eventos y realizar llamadas asíncronas a un servidor web. Además, incluye funcionalidades relacionadas con animaciones y efectos que resultan costosos de implementar únicamente con JavaScript y CSS [5].

JQuery es de código abierto y únicamente se necesita integrar un único fichero JavaScript que incluye todo el código necesario para proporcionar las funcionalidades mencionadas anteriormente. En suma, existe una gran comunidad y soporte detrás de biblioteca donde se puede encontrar documentación y componentes ya implementados que otorgan interactividad a los sitios web.

Principalmente se ha decido incluir JQuery al proyecto por que es un requisito para poder utilizar Boostrap. Sin embargo, se han aprovechada algunas de sus funcionalidades, sobretodo aquellas orientadas a la manipulación del DOM y sus atributos.

2.2.5. MxGraph

MxGraph es una biblioteca JavaScript que permite integrar grafos y diagramas interactivos en las aplicaciones web. De esta forma, el usuario puede crear elementos gráficos e interactuar con ellos mediante la interfaz. Además, estos elementos pueden asociarse a un modelo de datos definido por el desarrollador, permitiendo realizar cambios en el modelo a través del diagrama [1].

Otra ventaja es que esta biblioteca es totalmente gratuita y libre. Gracias a su gran flexibilidad y robustez, mxGraph es utilizado por muchas grandes empresas, por ejemplo el servicio web Draw IO está implementado con esta tecnología.

Por otra parte, mxGraph proporciona una API completa que permite a los desarrolladores crear sus aplicaciones de una forma consistente y en cualquier entorno. También cabe mencionar que está biblioteca está implementada únicamente con JavaScript, sin bibliotecas de terceros, de forma que la compatibilidad con otros *frameworks* está asegurada. Esto resulta de vital importancia, pues en la elección de las tecnologías de desarrollo se ha tenido en cuenta la flexibilidad y compatibilidad entre éstas.

MxGraph utiliza un lenguaje de gráficos de vectores, normalmente SVG, y HTML para dibujar los diagramas, por ésta razón puede ejecutarse en cualquier tipo de navegador.

La visualización de diagramas se basa en las teorías matemáticas de redes y grafos. En consecuencia, tal y como se muestra en la Figura 2.4, el diagrama posee dos tipos de elementos gráficos: los vértices y las aristas que los conectan. De este modo, es necesario analizar y definir como integrar el modelo de datos con estos elementos gráficos.

Ahora bien, un mismo grafo puede ser representado de muchas formas diferentes. Por esta

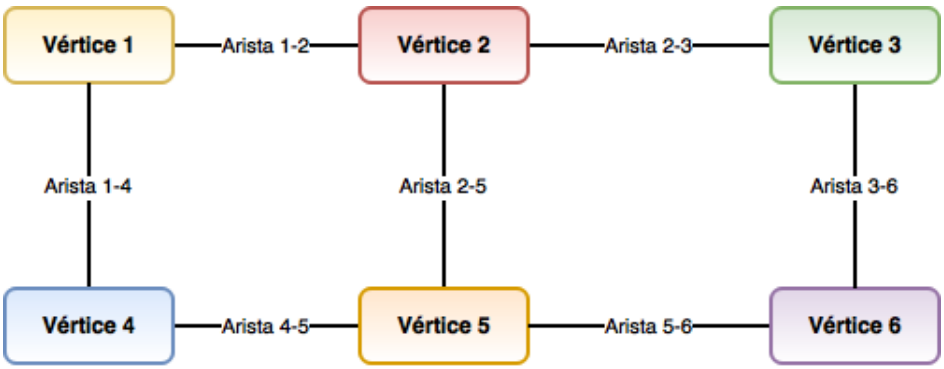

Figura 2.4: Ejemplo de un grafo simple con Draw IO

razón, uno de los puntos fuertes de mxGraph es el gran alcance y flexibilidad respecto a la visualización de grafos. Esta biblioteca proporciona un gran abanico de posibilidades y características orientadas a la visualización de vértices y diagramas, fijando el límite a las habilidades del programador. Por ejemplo, un vértice puede ser una animación, imagen, figura o incluso puros elementos HTML.

Por otra parte, mxGraph permite interaccionar con los elementos gráficos de muchas formas diferentes. Por ejemplo, a través de ésta se puede clonar, borrar, arrastrar, conectar, modificar el tamaño y forma o cambiar el estilo de los diferentes elementos gráficos. La gracia recae en la gran flexibilidad a la hora de programar la interactividad.

MxGraph se estructura en 8 paquetes que engloban toda su funcionalidad y que se detallan a continuación.

- El paquete **«editor»** que proporciona las clases necesarias para crear un editor de diagramas.
- Los paquetes **«view»** y **«model»** que implementan todos los componentes necesarios para crear y dibujar los diferentes elementos de un diagrama.
- Los paquetes **«handler»**, **«layout»** y **«shape»** que integran los manejadores de eventos, la forma en la cual se distribuyen los elementos gráficos y las formas o figuras de éstos.
- El paquete **«util»** que implementa utilidades generales para los desarrolladores, como la capacidad de arrastrar y soltar un elementos gráficos.
- El paquete **«io»** que proporciona diferentes clases para codificar y descodificar los diagramas en XML.

Como se ha podido observar, mxGraph proporciona todas las funcionalidades necesarias para representar una infraestructura informática a partir de un diagrama. La posibilidad de poder asociar cualquier modelo de datos a los elementos gráficos supone una gran ventaja para alcanzar los objetivos de este proyecto. Además, su gran flexibilidad permite integrarlo junto a otros *frameworks*, como Vue.js. Todas estas razones, más el hecho de que sea totalmente gratuita, han propiciado la elección de esta tecnología para dibujar los diagramas.

2.3. Tecnologías utilizadas en el servidor

En este apartado se describen las principales tecnologías utilizadas en la parte del servidor, así como los motivos por los cuales se han escogido.

2.3.1. Netbox

Netbox, cuyo logotipo se muestra en la Figura 2.5, es una aplicación web, de código abierto, diseñada para la gestión y documentación de redes. Netbox fue inicialmente desarrollado por el equipo de DigitalOcean con el fin de cubrir las necesidades de los ingenieros de redes y de infraestructuras informáticas. Esta herramienta está implementada con Django (Python) y utiliza PostgreSQL como base de datos, funcionando de esta forma como un servicio WSGI detrás de cualquier servidor HTTP [2].

Figura 2.5: Logotipo de Netbox

(Fuente: https://github.com/digitalocean/netbox)

Cabe destacar que Netbox funciona como una herramienta de gestión de direcciones IP (IPAM) y como una herramienta de gestión de infraestructura de centros de datos (DCIM). De este modo, el alcance de Netbox no se limita únicamente a la gestión de redes sino que también permite documentar dispositivos. Así pues, esta tecnología se divide en 8 módulos que abarcan los siguientes aspectos relacionados con las gestión de infraestructuras informáticas.

Gestión de direcciones IP: Incluye la gestión de redes, direcciones IP, enrutamiento virtual (VRF) y redes locales virtuales (VLAN).

Gestión de armarios: Permite organizar los armarios en grupos o por ubicaciones, es decir, para cada cliente se puede definir varias ubicaciones o edificios.

Gestión de dispositivos: Permite documentar dispositivos y asociar información a éstos, como el rol que desempeñan, servicios, interfaces, modelos o su ubicación dentro de la infraestructura.

Gestión de conexiones: Permite almacenar y mantener información sobre los diferentes tipos de conexiones que se pueden establecer en una red, así como la interfaces o bocas que estas conexiones conectan.

Gestión de circuitos: Permite la posibilidad de mantener información sobre los proveedores y circuitos de comunicación existentes en una corporación.

Gestión de credenciales: Otorga la posibilidad de almacenar, de forma encriptada y segura, datos sensibles, como las credenciales, y asociarlos a dispositivos.

Como se puede observar, Netbox ofrece muchas características que son totalmente necesarias para el logro de todos los objetivos propuestos en la etapa inicial del proyecto. Su módulo IPAM junto a su módulo DCIM proporcionan toda la funcionalidad necesaria para poder realizar tareas típicas de documentación de infraestructuras y redes. Por otra parte, la posibilidad de poder almacenar credenciales y asociarlas a dispositivos facilita el desarrollo del establecimiento de conexiones remotas. Adicionalmente, Netbox ofrece una API de Transferencia de Estado Representacional (REST), permitiendo realizar todas las operaciones descritas mediante llamadas asíncronas desde una interfaz propia. Es por todo esto que Netbox resulta la solución ideal para mantener el modelo de datos de la aplicación.

2.3.2. Django

Django es un *framework* de Python que facilita enormemente la implementación de aplicaciones web. La base de Django consiste en ofrecer un alto nivel de abstracción de los patrones comúnmente utilizados en el desarrollo web. De esta forma, el programador evita la implementación tediosa de código que suelte utilizarse en cualquier aplicación y puede centrarse en la resolución de problemas.

Este *framework* permite implementar el patrón Modelo-Vista-Controlador (MVC) de una forma rápida y sencilla. La lógica de este patrón se divide en la implementación de 3 ficheros donde se definen los modelos, la lógica y las vistas que deben ser servidas ante una determinada URL. De este modo, los diferentes componentes no dependen uno de otros y cada una de las partes puede ser desarrollada de forma independiente [7].

Esta tecnología se ha integrado en el proyecto debido a que es necesaria para el funcionamiento de Netbox. Además, se ha utilizado para extender Netbox con la creación de nuevos módulos destinados a cubrir las necesidades de la empresa.

2.3.3. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos relacionales (SGBDR) orientada a objetos. Este sistema es gratuito y proporciona una gran número de aspectos avanzados relacionados con la gestión de bases de datos. Además, es multiplataforma por lo que puede ser ejecutado en cualquier tipo de sistema operativo, ya sea Mac OS X, Windows o Linux.

Una de las principales ventajas que ofrece PostgresSQL es su buen sistema de planificación y optimización, permitiendo conseguir un rendimiento óptimo en las consultas. Además, integra un control de concurrencias multiversión (MVCC) que ofrece la posibilidad de realizar transacciones eventualmente consistentes. Estas razones, sumadas a su gran flexibilidad y escalabilidad, han propiciado que PostgresSQL sea uno de los SGBDR más utilizados en las grandes corporaciones [10].

Se ha decidido utilizar PostgreSQL en el proyecto porque Netbox necesita el tipo de campo de dirección de red proporcionado por PostgreSQL.

2.3.4. Apache

Apache es un servidor web HTTP abierto y gratuito que está continuamente desarrollado y mantenido por la Apache Software Foundation. Este servidor web puede ser utilizado en cualquier plataforma y es uno de los servidores web más utilizados en la actualidad.

Su arquitectura se descompone en un núcleo, que incorpora todas las funciones básicas de un servidor web, y en varios módulos que se utilizan para extender su funcionalidad. De esta forma, se pueden instalar o desinstalar funcionalidades según las características y naturaleza de las aplicaciones que debe alojar. Así pues, Apache posee muchas propiedades como configuraciones simples basadas en ficheros, autenticación HTTP, admisión de SSL, capacidad de trabajar como *Proxy* o soporte para servidores virtuales [8].

Por lo tanto, Apache ofrece extensibilidad, gratuidad, flexibilidad, seguridad y robustez. Gracias a sus módulos de autenticación y admisión de SSL, se puede comunicar cualquier dato de la aplicación de forma segura. Por otra parte, la posibilidad de configurar Apache como un *Proxy* puede ser aprovechada para controlar los accesos a la aplicación. Además, si se tiene en cuenta la gran comunidad y gran cantidad de documentación, como tutoriales y manuales, Apache se puede considerar un buen servidor web donde poder desplegar la aplicación.

Capítulo 3

Planificación del proyecto

Contenidos

En este capítulo se detalla la metodología que se ha usado para el desarrollo del proyecto, es decir, se describe la forma en la cual se ha realizado la planificación y se analizan las desviaciones que han surgido durante su transcurso. Además, se presenta una estimación de los recursos y costes que supone llevar a cabo la materialización del proyecto.

3.1. Metodología

La metodología de trabajo ha consistido en presentaciones semanales del estado del proyecto. Un día a la semana se ha realizado una reunión con el supervisor donde se ha explicado todo el trabajo realizado durante la semana anterior. Después de la revisión, el supervisor valora si se ha estado siguiendo un buen camino en el desarrollo del proyecto o, si de lo contrario, es necesaria su intervención para redirigirlo correctamente. En ese momento, se definirán detalladamente las tareas a realizar durante la semana en curso y se realizará un listado de todos los recursos necesarios para su desempeño.

Para la definición, seguimiento y planificación de tareas se ha utilizado la herramienta MeisterTask[1], una herramienta de gestión de tareas y proyectos que resulta muy intuitiva, visual y fácil de utilizar. En esta herramienta se definen las tareas con sus correspondientes subtareas. Cada vez que se ha trabajado en una tarea se ha iniciado un temporizador para calcular el tiempo dedicado a la misma. Además, esta aplicación permite documentar y anotar las acciones que se consideren importantes para la realización de la tarea. La propia herramienta permite

[1] https://www.meistertask.com/es

realizar toda esta funcionalidad, de forma centralizada, por parte del usuario y solo es necesario un navegador web.

Tal como se puede observar en la Figura 3.1, la organización semanal de tareas se ha procedido siguiendo las pautas que se detallan a continuación.

- Las nuevas tareas planteadas en la reunión se abren en la columna «Abrir».
- Las tareas abiertas que van a ser trabajadas durante la semana en curso se mueve a la columna «En progreso».
- Los diferentes errores que surjan durante el desarrollo se abren en la columna «Bugs».
- Cuando se termina una tarea en curso se mueve a la columna «Para revisión».
- Al inicio de cada reunión se revisan las tareas pendientes de revisión, si el supervisor da el visto bueno se mueven a la columna «Finalizadas» y se archivan.

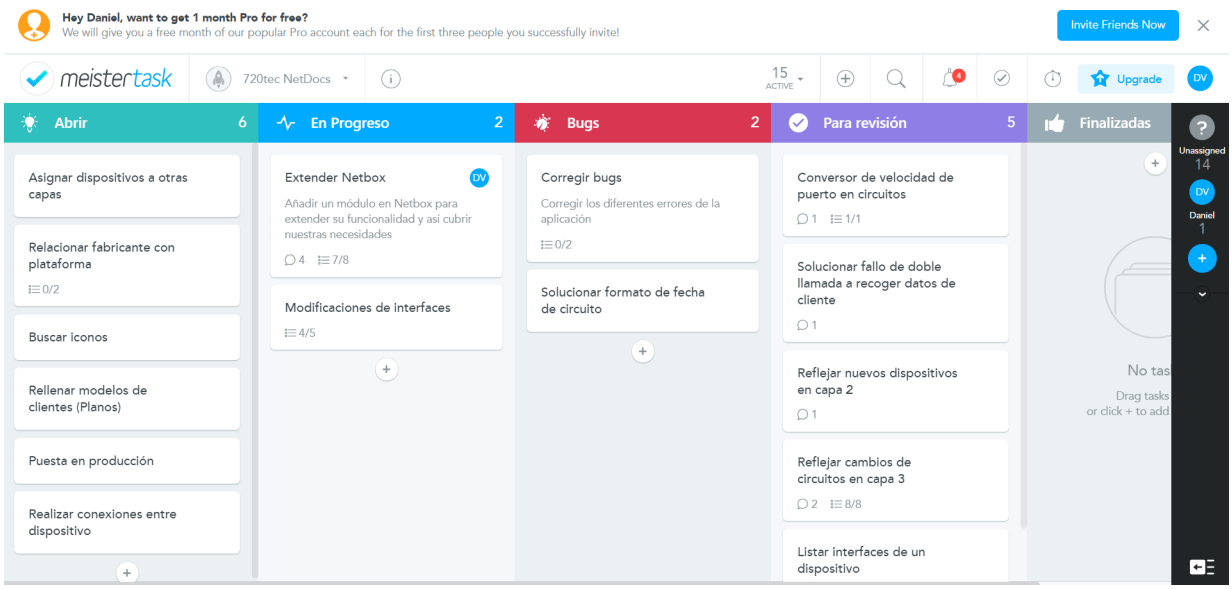

Figura 3.1: Ejemplo de organización de tareas con Meistertask

3.2. Planificación inicial

A pesar de que las tareas se han ido definiendo semanalmente según las peticiones de la empresa, en el inicio del proyecto se realizó una planificación inicial para orientar la implementación de la aplicación. Así pues, en este apartado se detalla el listado de tareas definido al inicio del proyecto y se presenta la planificación inicial, de una forma visual, a través de una diagrama de Gantt.

3.2.1. Lista de tareas

En el Cuadro 3.1 se muestra el listado de tareas definidas al inicio del proyecto, indicando el coste temporal y sus dependencias. Estas tareas fueron diseñadas con el objetivo de cubrir todos los aspectos del proyecto, tanto de implementación como de documentación.

Cuadro 3.1: Lista de tareas planteadas al inicio del proyecto

Nº	**Tarea**	**Horas**	**Dependencias**
1	**Desarrollar propuesta técnica**		
1.1	**Inicio**	**10**	
1.1.1	Definir proyecto con tutor y supervisor	2	
1.1.2	Definir método de trabajo y documentación	4	1.1.1
1.1.3	Definir formato y estándares	4	1.1.1
1.2	**Planificar proyecto**	**23**	
1.2.1	Definir tareas	3	1.1
1.2.2	Diagrama de Gantt	8	1.2.1
1.2.3	Documentar propuesta técnica	12	1.2.2
1.2.4	Entregar propuesta técnica	0	1.2.3
2	**Desarrollo técnico del proyecto**		
2.1	**Elección de las herramientas de desarrollo**	**18**	
2.1.1	Elección de herramienta de gestión de IP	8	
2.1.2	Elección de tecnología del backend	1	
2.1.3	Elección de tecnología de la interfaz	1	
2.1.4	Elección,de biblioteca de diagramas	8	
2.2	**Recopilación y análisis de requisitos**	**46**	
2.2.1	Identificación,de requisitos	5	
2.2.2	Casos de uso	25	2.2.1
2.2.3	Requisitos de datos	12	2.3.3
2.2.4	Requisitos,tecnológicos y de plataforma	4	
2.3	**Diseño de la herramienta**	**56**	
2.3.1	Diseño de la interfaz de usuario	6	
2.3.2	Creación de prototipo	40	
2.3.3	Validación del prototipo	0	
2.3.4	Diseño de la API	10	2.2.3
2.4	**Formación en tecnologías**	**15**	
2.4.1	Formación en Vue.js	10	
2.4.2	Formación en Django	5	
2.5	**Desarrollo del producto**	**105**	

Continúa en la siguiente página...

Lista de tareas planteadas al inicio del proyecto (continuación)

Nº	Tarea	Horas	Dependencias
2.5.1	Preparación de entorno	2	
2.5.2	Gestión de elementos de red	40	2.3
2.5.3	Gestión de conexiones	10	2.3
2.5.4	Visualización del diagrama	30	2.3
2.5.5	Conexiones remotas	18	2.3
2.5.6	Pruebas	5	2.5.2
2.6	**Documentación**	**20**	
2.6.1	Manual web	20	2.5
2.7	**Puesta en marcha**	**7**	
2.7.1	Despliegue	3	2.5
2.7.2	Formación	4	2.7.1
2.7.3	Entrega	0	2.7.2
3	**Documentación y presentación del TFG**	**135**	
3.1	Redacción de informes quincenales	4	
3.2	Redacción de memoria técnica	100	
3.3	Libramiento de memoria técnica	0	
3.4	Preparación de presentación oral	30	
3.5	Presentación oral	1	

3.2.2. Diagrama de Gantt

El diagrama de Gantt es una herramienta que permite mostrar visualmente el tiempo de trabajo previsto para cada una de las tareas en el marco de una planificación. En la figura 3.2 se muestra el diagrama de Gantt correspondiente a la planificación inicial de las tareas descritas en el anterior apartado.

3.3. Estimación de recursos y costes del proyecto

A continuación, se muestra una estimación del coste que supone la materialización del proyecto. Por una parte, es necesario tener en cuenta el coste en personal, es decir, el coste que le supone a la empresa tener un programador desarrollando el proyecto.

Por una parte, el coste medio por hora que le supone a la empresa mantener a un programador es de 9,4€. Por lo tanto, el coste derivado por las 300 horas necesarias para la elaboración de la aplicación es el siguiente:

$$9{,}4€/hora * 300horas = 2\,820€$$

Por otra parte, también es necesario tener en cuenta el coste de todo el material físico. Para el desarrollo del proyecto se ha utilizado un portátil Dell Inspiron Cn 73402, 2 pantallas Acer K222HQLbid de 21,5 pulgadas, un brazo articulado para las pantallas, una mesa y una silla. La amortización derivada del uso de este material por los 3 meses que ha durado la estancia en prácticas es de 80,44 €.

Aparte, también hay que sumar el coste de la máquina donde se aloja el *backend*. En este caso, se ha utilizado una máquina virtualizada en la nube, lo que supone un coste de 0,2 € la hora (una máquina «m4.xlarge» en Amazon Web Services [2]), sumando un total de:

$$0{,}2€/hora * 300horas = 60€$$

Así pues, todo junto suma un coste total de:

$$2\,820€ + 80{,}44€ + 60€ = 2\,960{,}44€$$

3.4. Desviaciones del proyecto

En este apartado se presentan las principales desviaciones y complicaciones que han surgido durante el desarrollo del proyecto. Además, se explica qué medidas se han llevado a cabo para reducir su impacto en la planificación.

3.4.1. Desviaciones

Aparte de pequeñas desviaciones asociadas al coste temporal de implementar ciertas funcionalidades, han habido dos complicaciones principales que se han reflejado en un mayor coste temporal.

La primera de estas complicaciones ha consistido en el elección de la tecnología encargada de gestionar los datos en la parte del servidor. Inicialmente existían dos alternativas posibles: phpIPAM y Netbox, propuestas que se eligieron por ser totalmente gratuitas y por proporcionar una API REST. Sin embargo, después de un análisis más exhaustivo de las dos tecnologías se descubrió que la API REST de Netbox era solamente de lectura, es decir, no permitía crear, borrar ni modificar información en la base de datos. Por lo tanto, se acabó eligiendo phpIPAM, la cual sí proporcionaba una API REST completa. El problema surgió después de documentar algunos ejemplos con esta herramienta, ya que resultó poseer un modelo de datos muy pobre y

[2] Amazon Web Services es una plataforma de servicios en la nube. Se pueden encontrar los precios de las instancias EC2 en https://aws.amazon.com/es/ec2/pricing/

poco intuitivo. De hecho, únicamente permitía almacenar la información de un solo cliente. Esto supuso un replanteamiento del proyecto, e incluso se empezó a diseñar una API REST propia que trabajara sobre la base de datos de Netbox. Un tiempo después, se publicó una versión *beta* de Netbox que incluía la API REST completa, tanto de lectura como de escritura. En consecuencia, se optó por dejar el diseño de la API REST propia y se decidió utilizar Netbox. Todo esto se vió reflejado en un aumento del tiempo necesario para completar la tarea de elección de la tecnología del servidor, de forma que lo que estaba planeado realizar en pocos días, acabó costando unas tres semanas.

La segunda complicación, que ha supuesto grandes pérdidas, temporales ha sido el desconocimiento de las bibliotecas de la parte del cliente. Se ha necesitado gran cantidad de tiempo para aprender a utilizar Vue.js y mxGraph. Por una parte, la curva de aprendizaje de Vue.js ha sido relativamente rápida, pues una vez se entiende el funcionamiento de este *framework*, resulta muy fácil e intuitivo de utilizar. Aún así, se han tenido que invertir unas 10 horas para poder aprender su funcionamiento básico. Por otra parte, mxGraph ha supuesto todo lo contrario. Realizar el dibujado de diagramas básicos ha sido fácil, pues en la documentación oficial existen múltiples ejemplos que proporcionan el código necesario para implementarlos. Sin embargo, el resto de documentación es muy pobre y cuando se quiere implementar algún tipo de funcionalidad que no aparece en los ejemplos, no queda otro remedio que ir probando diferentes funciones de la API hasta que se consigue el efecto deseado. Esto se debe a que la gran flexibilidad de mxGraph supone una gran complejidad a la hora de implementar algo, pues cuando se desea cambiar alguna de sus funcionalidades se necesita sobrescribir los métodos de las clases que proporciona. Todo esto ha supuesto un mayor coste temporal a la hora de implementar todas las funcionalidades que interactuan con el diagrama, especialmente las relacionadas con reflejar cambios del modelo en el diagrama y las de implementar los editores.

3.4.2. Soluciones

Para reaccionar ante las desviaciones de la planificación inicial, se ha optado por una estrategia de priorización de tareas. Esta estrategia se ha ido desarrollando en cada reunión semanal, donde se iban definiendo el orden y prioridad de cada una de las tareas en curso. En estas reuniones se planteaban los problemas que habían surgido durante la semana en curso y se negociaba, con la empresa, qué tareas se debían priorizar.

Como resultado de todo este proceso, el proyecto se ha acabado limitando en cuanto a requisitos, de forma que las tareas que no se han podido cumplir se han dejado para un desarrollo futuro. Las tareas que no se han podido realizar son:

Conexiones remotas: Se ha decido prescindir de la funcionalidad que permite realizar conexiones remotas con los dispositivos de los clientes. La razón de esta decisión recae en que esta funcionalidad puede ser muy complicada de implementar, pues el cliente VPN que se debe utilizar puede variar en función del dispositivo con el cual se quiere conectar.

Validación y despliegue de la aplicación: Como no se ha llegado a una versión de la aplicación lo suficiente desarrollada, se ha optado por aplazar su despliegue. Antes de llevar a cabo el despliegue final de la aplicación, se ha planeado realizar un despliegue controlado

en el cual se validará la aplicación. Por lo tanto, durante esta fase los usuarios podrán plantear cambios dirigidos a mejorar la usabilidad de la herramienta.

Manual web: Al no realizar el despliegue de la aplicación, se ha decido retrasar el desarrollo de su manual.

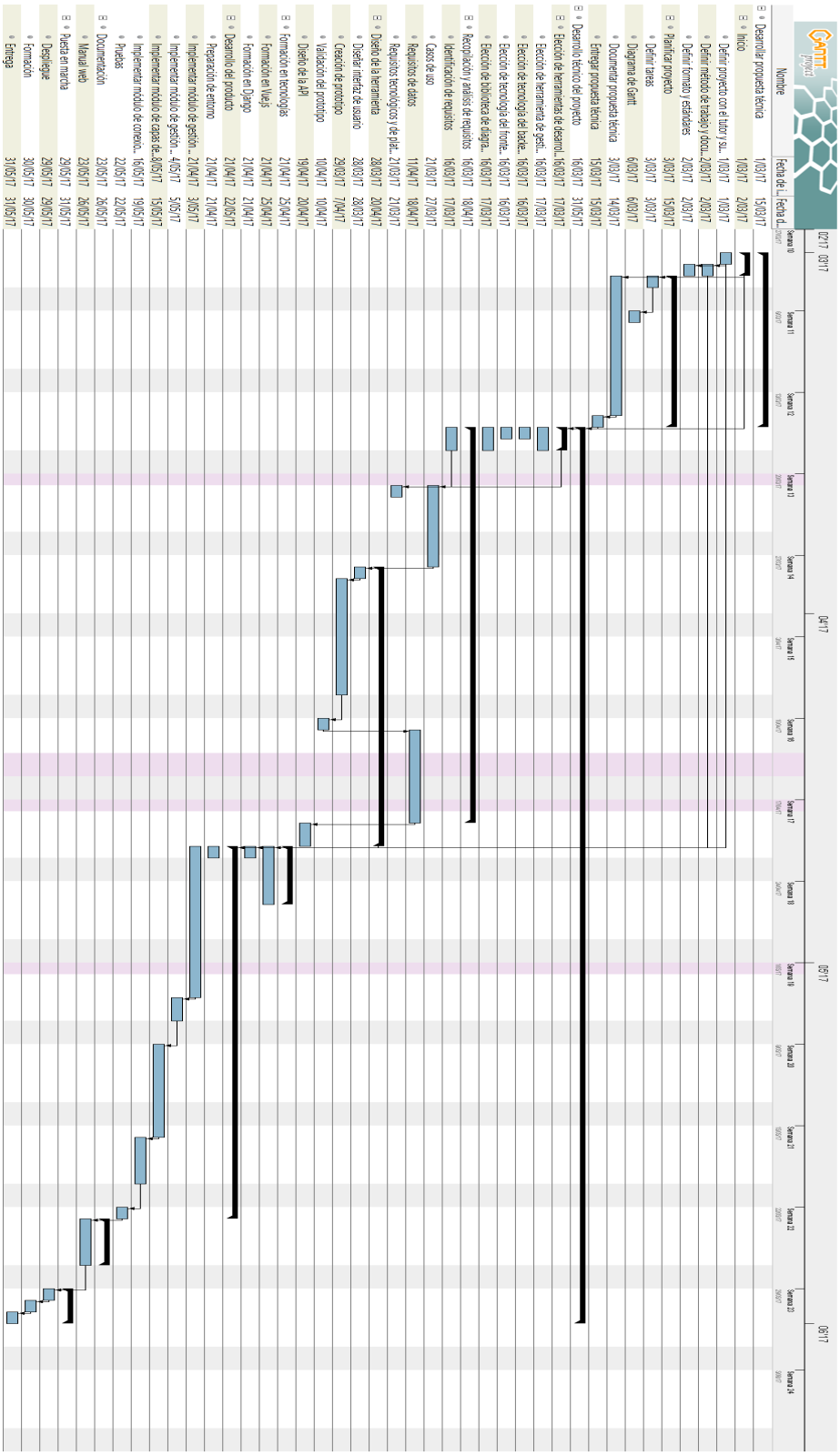

Figura 3.2: Diagrama de Gantt de las tareas del proyecto

Capítulo 4

Análisis y diseño del sistema

Contenidos

4.1. Definición de requisitos

Los requisitos son las condiciones que el software debe cumplir o las características que debe tener para poder conseguir los objetivos y el alcance definidos, teniendo en cuenta todas las limitaciones del proyecto.

Así pues, el primer paso consiste en identificar los actores principales de la herramienta. Cada actor representa un rol que puede adoptar una persona, grupo de personas o incluso dispositivos sobre el sistema. Como en este caso la herramienta va dirigida expresamente al personal de soporte de 720Tec, se puede definir a este grupo como el actor principal. Aparte, también se debe tener en consideración los agentes externos e internos que utilizan la herramienta para llevar a cabo su funcionalidad, es decir, el API que proporciona Netbox para la gestión de direcciones IP, el cliente VPN y los dispositivos externos con los cuales se establece conexión.

Por otra parte, también es necesario identificar, a grandes rasgos, cada una de la funciones que debe poder realizar el sistema para poder atribuirlo a los agentes y generar el diagrama de casos de uso. Teniendo esto en mente, se han definido los siguientes casos de uso para poder cubrir los objetivos planteados al inicio del proyecto.

- **CU01:** Gestionar clientes.
  - **CU01.1:** Añadir cliente.
  - **CU01.2:** Modificar cliente.

    - **CU01.3:** Eliminar cliente.
    - **CU01.4:** Listar clientes.
- **CU02:** Gestionar elementos de red.
    - **CU02.1:** Añadir elemento.
    - **CU02.2:** Modificar elemento.
    - **CU02.3:** Eliminar elemento.
    - **CU02.4:** Consultar información de elemento.
    - **CU02.5:** Documentar elemento.
- **CU03:** Gestionar conexiones de red.
    - **CU03.1:** Añadir conexión.
    - **CU03.2:** Modificar conexión.
    - **CU03.3:** Eliminar conexión.
- **CU04:** Gestionar la visualización de red.
    - **CU04.1:** Modificar diagrama.
    - **CU04.2:** Almacenar diagrama.
    - **CU04.3:** Recuperar diagrama.
    - **CU04.4:** Visualizar capa.
- **CU05:** Conectar con la red cliente.
    - **CU04.1:** Montar VPN.
    - **CU04.2:** Establecer conexiones remotas.

En la Figura 4.1 se muestra el diagrama de casos de uso de la aplicación. Este diagrama consiste en una representación gráfica de la relación existente entre los requisitos, o funcionalidades que debe incluir la herramienta, con cada uno de los actores que intervienen en el sistema. Además, en el Anexo A se muestra un documento que define, de forma más detallada, cada uno de los casos de uso que se plantearon en la planificación inicial del proyecto. Como se puede observar en la Figura 4.1 cuando el equipo de soporte interactúa con el diagrama de red, el sistema realiza las llamadas correspondientes a la API de Netbox, es por ello que éste también interviene como actor de todas estas funciones.

Por otra parte, cuando el equipo de soporte establece una conexión con algunos de los elementos de la red, esta conexión se realiza, si es posible, con la máquina física del cliente. Además, al momento de establecer las conexiones también es necesario considerar los clientes de conexión que intervienen en el proceso.

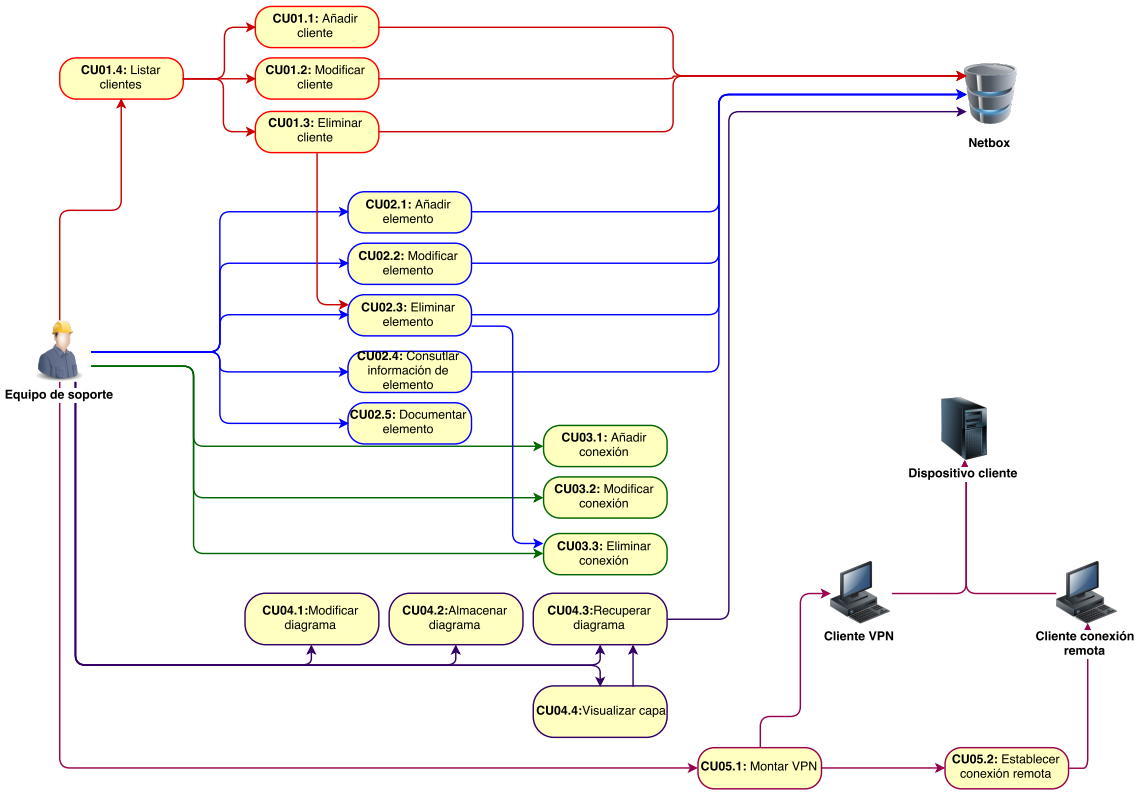

Figura 4.1: Diagrama de casos de uso de la aplicación

4.2. Diseño de la arquitectura del sistema

En este apartado se define la arquitectura del sistema, es decir, se identifican los principales elementos que intervienen en éste y se detalla como se relacionan entre ellos. La aplicación que se ha desarrollado se basa en una arquitectura cliente-servidor. Esta arquitectura es un modelo de computación distribuida en la cual las tareas se reparten entre las máquinas que ofrecen servicios y las máquinas que los solicitan. Esta organización posibilita la centralización de la gestión de la información y la división de deberes o funciones [4]. Así pues, el sistema se podría dividir en 3 capas o partes: el cliente, el servidor web y el servidor de la base de datos. El cliente es donde se presenta la vista de la aplicación, es decir, es la interfaz a través de la cual el usuario interactúa con el sistema. Al tratarse de una aplicación web, el cliente consiste en cualquier navegador, como Chrome, Firefox o Internet Explorer. Cada vez que el usuario realiza una acción sobre la interfaz, el cliente realiza las peticiones correspondientes a la API REST de Netbox. Estas peticiones son enviadas al servidor HTTP Apache, el cual se encarga de servir la API y la interfaz de Netbox. En esta API se encuentra la lógica del *backend*, donde se realizan unas acciones u otras según el tipo de petición. En ese momento, Netbox consulta el modelo que se encuentra en el servidor de la base de datos. Finalmente, se devuelve la respuesta de la acción realizada a la interfaz para que realice los cambios visuales que sean necesarios.

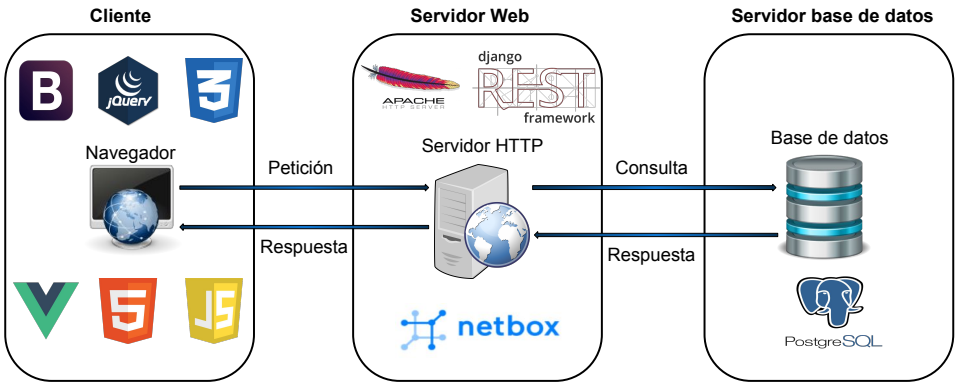

Figura 4.2: Arquitectura del sistema

4.3. Extensión de Netbox

A pesar de que Netbox incluye un modelo de datos muy extenso, éste ha sido diseñado para cubrir propósitos muy generales y así poder llegar a un número más amplio de usuarios. Es por ello que Netbox no cubre todas las necesidades que posee el equipo de 720Tec a la hora de documentar infraestructuras informáticas. Concretamente, es necesario cubrir la siguientes peticiones de funcionalidad propuestas por 720Tec:

1. Poder asociar las VLAN de un cliente a las diferentes interfaces de un dispositivo con el fin de poder representar el tipo de tráfico que circula por cada una de las conexiones.
2. Poder almacenar los diagramas en la base de datos.
3. Poder vincular los diagramas y los dispositivos a diferentes capas con el objetivo de poder gestionar su visualización.

Para poder ampliar el código de Netbox se ha aprovechado la capacidad que proporciona Django para modularizar aplicaciones. Netbox está divido en módulos, cada uno de los cuales incluye su modelo de datos, lógica de control y vistas a retornar. Teniendo esto en cuenta, se ha optado por la opción de diseñar e implementar un nuevo módulo el cual incluye todas las funcionalidades extra que necesita el sistema. Este módulo se ha diseñado de forma que pueda referenciar a los demás módulos incluidos en Netbox, pero no a la inversa. De este modo, no se pierde la capacidad de actualizar el código fuente de Netbox, pues éste no se ha modificado en ningún momento. Simplemente, se ha añadido una nueva funcionalidad de forma modular.

4.3.1. Diseño del módulo propio de 720Tec

Con el objetivo de poder integrar las funcionalidades descritas en el apartado anterior, se ha diseñado 4 nuevos modelos de datos que corresponden a las entidades que se definen a continuación.

InterfaceVlan: Entidad que permite asociar una o varias VLAN a las interfaces de un dispositivo. Para ello, este modelo incluye referencias a los modelos «Interface» y «Vlan» de Netbox. Además, se ha incluido un campo que indica el modo de puerto de la interfaz, el cual puede marcarse como etiquetado o como no etiquetado.

Capa: Esta entidad representa las diferentes capas de visualización que pueden existir en el sistema. Simplemente, se ha incluido un campo que permite definir el nombre de la capa.

CapaDevice: Este modelo permite asociar una capa a un dispositivo para que la visualización de éstos pueda ser gestionada más fácilmente. Para ello se incluye una clave ajena al modelo «Device» de Netbox y otra clave ajena al modelo «Capa».

Diagram: Esta entidad permite almacenar, en formato texto, los diagramas codificados en XML y asociarlos a una determinada capa, cliente y sitio. Para ello se utilizan las claves ajenas a cada una de las correspondientes entidades de Netbox. La posibilidad de asociar un diagrama a un cliente permite la recuperación de diagramas en base a esta entidad. Además, el hecho de poder asociarlo a los diferentes sitios de un cliente permite integrar un sistema de navegación simple de los diferentes diagramas de un mismo cliente.

Tal y como se puede observar en la Figura 4.3, existen diferentes relaciones entre los modelos que se ha diseñado y las entidades contenidas en diferentes módulos de Netbox. Ahora bien, estas relaciones son unidireccionales, es decir, existen referencias del módulo de 720Tec hacia los otros módulos, pero no a la inversa. De este modo se consigue evitar problemas en la base de datos si se desea realizar una actualización del código fuente de Netbox.

4.3.2. Diseño de la API

Una vez terminado el diseño de los nuevos modelos de datos, también ha sido necesario diseñar una API REST que permita manipular estas entidades desde la interfaz.

Esta API REST integra las operaciones típicas de la especificación HTTP para cada una de las entidades descritas. Estas operaciones corresponden a las funciones básicas que se pueden realizar sobre una base de datos: GET para leer, DELETE para borrar, PUT para modificar y POST para crear.

Respecto a la jerarquía, toda la API REST de Netbox cuelga de la raíz, /api/. A partir de ese punto, tal como se puede observar en la Figura 4.4 la estructura de las URL se divide a nivel de aplicación o módulo: «circuits», «DCIM», «extras», «IPAM», «secrets», «tenancy» y la aplicación «tec720» que se ha añadido. Por lo tanto, la API REST que se ha diseñado cuelga de la raíz del nuevo módulo que se ha incorporado, /api/tec720/. De este modo, se puede acceder a cada una de las entidades añadidas a través de las siguientes URL:

- /api/tec720/diagram/
- /api/tec720/interfacevlan/
- /api/tec720/capa/

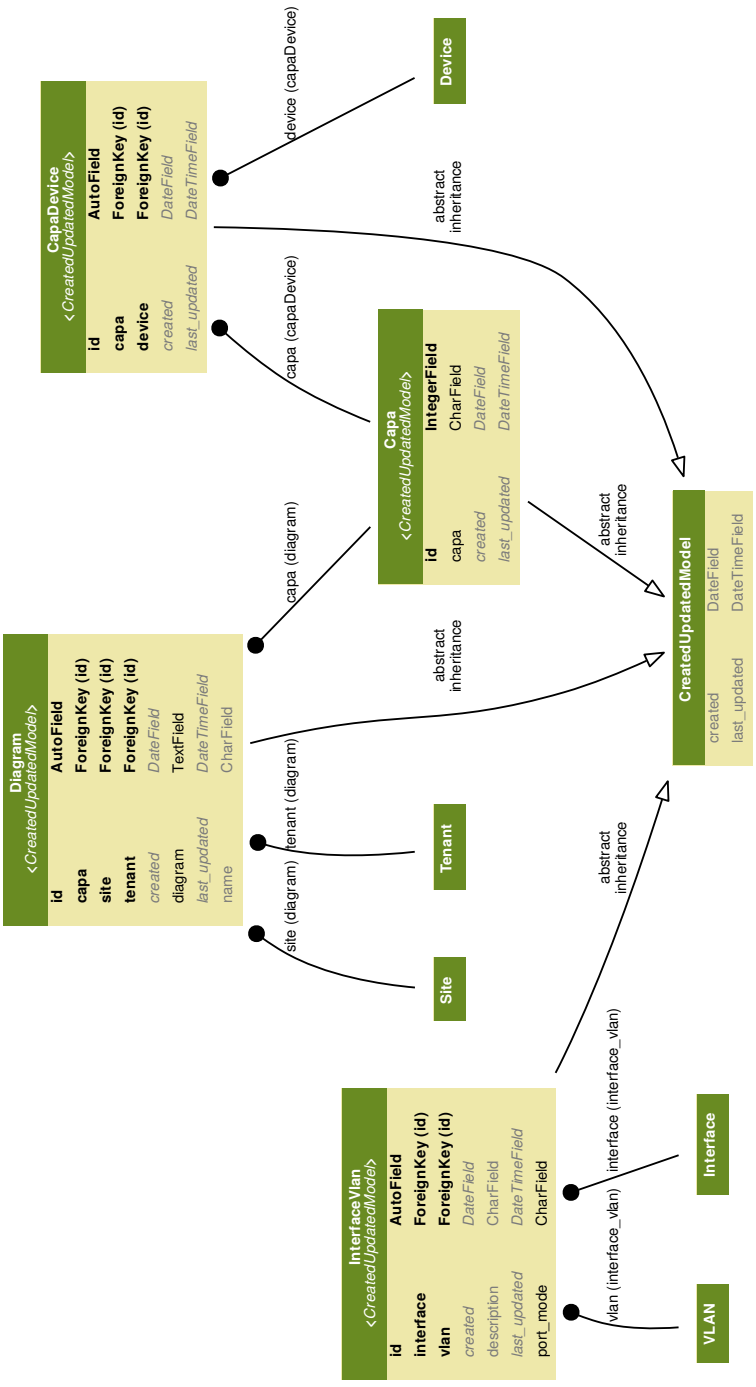

Figura 4.3: Diagrama de los modelos de datos del módulo de «tec720»

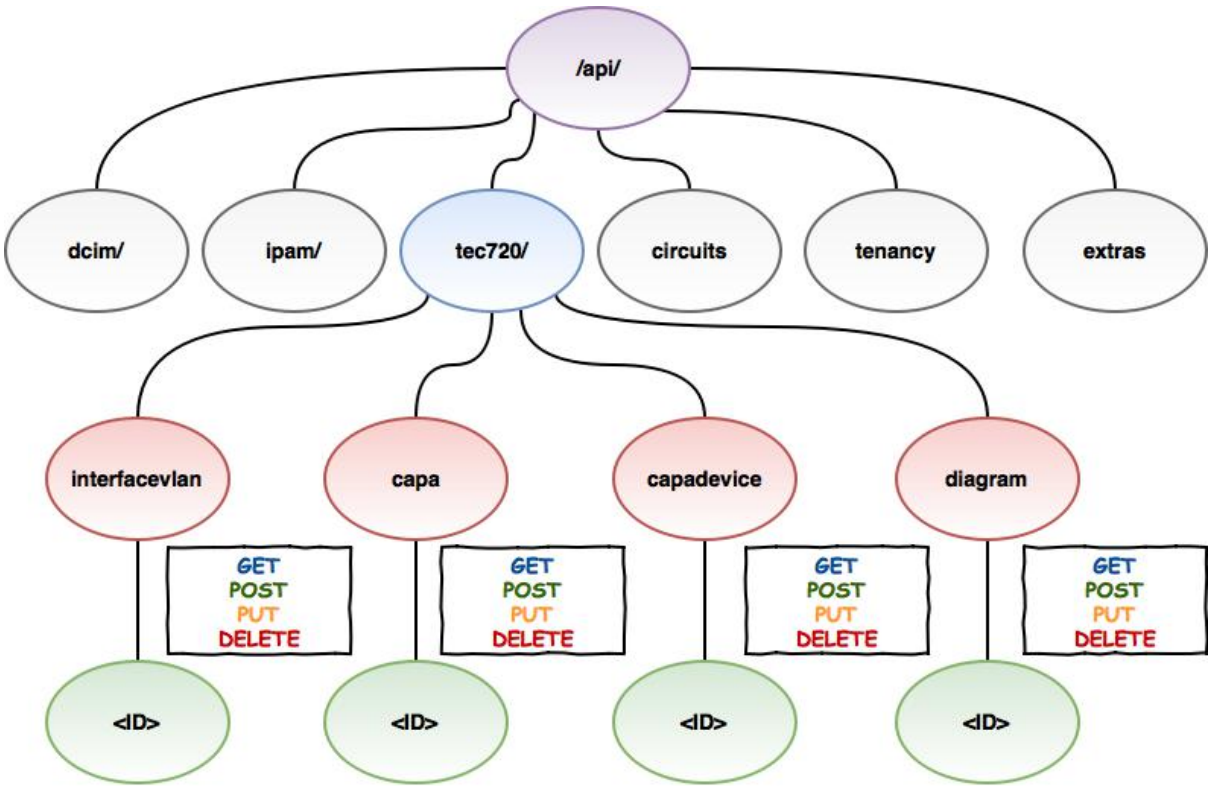

Figura 4.4: Jerarquía de la API REST que se ha implementado

- /api/tec720/capadevice/

Cabe tener en cuenta que a cada modelo se le ha asociado dos URL: una de listar varios elementos de la entidad y otra para realizar operaciones sobre un único elemento. Esta segunda se utiliza para realizar alguna operación CRUD[1] sobre un elemento. Es por ello que esta URL necesita el identificador que corresponde la clave primaria del elemento, llegando así a los nodos hoja de la jerarquía. Por ejemplo, las URL asociadas al modelo «capa» son:

- /api/tec720/capa/ - Lista todas las capas.
- /api/tec720/capa/1/ - Realiza una operación CRUD sobre la capa con identificador 1.

4.4. Diseño de la interfaz

Para finalizar la etapa de diseño se ha definido, a grandes rasgos, cómo debe ser la interfaz de usuario. Para ello se han establecido una serie de directrices a seguir durante la implementación de la aplicación y se ha elaborado un prototipo general de cómo debería visualizarse la interfaz.

Gracias a este proceso se ha tenido en cuenta, durante todo la implementación, cómo debería

[1] Acrónimo de *Create*, *Read*, *Update* and *Delete* que se utiliza para referenciar las operaciones típicas que se realizan sobre una base de datos.

ser el diseño final de la interfaz de usuario. Aún así, cabe recalcar que los prototipos que se muestran en este apartado se diseñaron al inicio del proyecto por lo que el resultado final ha variado en base a las peticiones y demandas semanales del equipo de 720Tec.

A grandes rasgos, las directrices que se han seguido para la implementación de la interfaz de usuario han sido las siguientes:

- El diseño debe ir orientado más a la usabilidad que a una visualización agradable.
- La interfaz debe permitir visualizar la información más importante en un primer nivel, sin necesidad de interactuar con los elementos.
- La interfaz debe ser amigable e intuitiva para el equipo de soporte.
- La paleta de colores no debe distar mucho de los colores que conforman el logotipo de la empresa y éste debe incluirse en alguna parte de la interfaz.
- Se debe maximizar el espacio del diagrama, pues éste constituye el núcleo de la aplicación.

Siguiendo estas indicaciones se ha desarrollado el siguiente prototipo, el cual sigue el diseño típico de las aplicaciones web compuesto por una cabecera, barras laterales, un cuerpo central y un pie de página.

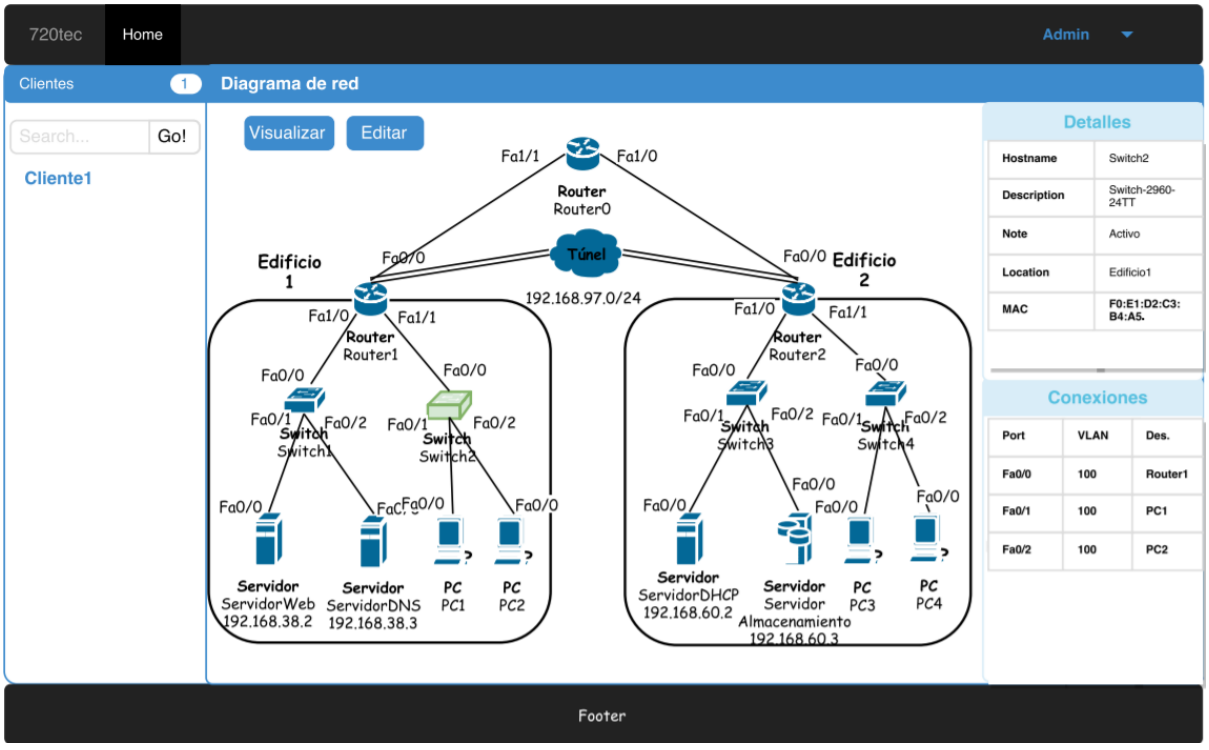

Figura 4.5: Prototipo de la ventana principal de la aplicación

Tal como se observa en la Figura 4.5, la interfaz se basa en un diseño simple pero eficaz. Los menús de navegación se utilizan para moverse entre las diferentes entidades principales del modelo, como pueden los clientes. El diagrama ocupa la mayor parte de la interfaz y desde

esta zona se realizarán casi todas las acciones que pueda efectuar el usuario. En el diagrama se muestra la información más importante de la red, como puede ser nombre de dispositivos, direcciones IP o nombres de interfaces de conexión. Cuando se quiere consultar información más detallada de alguno de los elementos, el sistema lanza una nueva ventana con un diseño parecido al prototipo de la Figura 4.6.

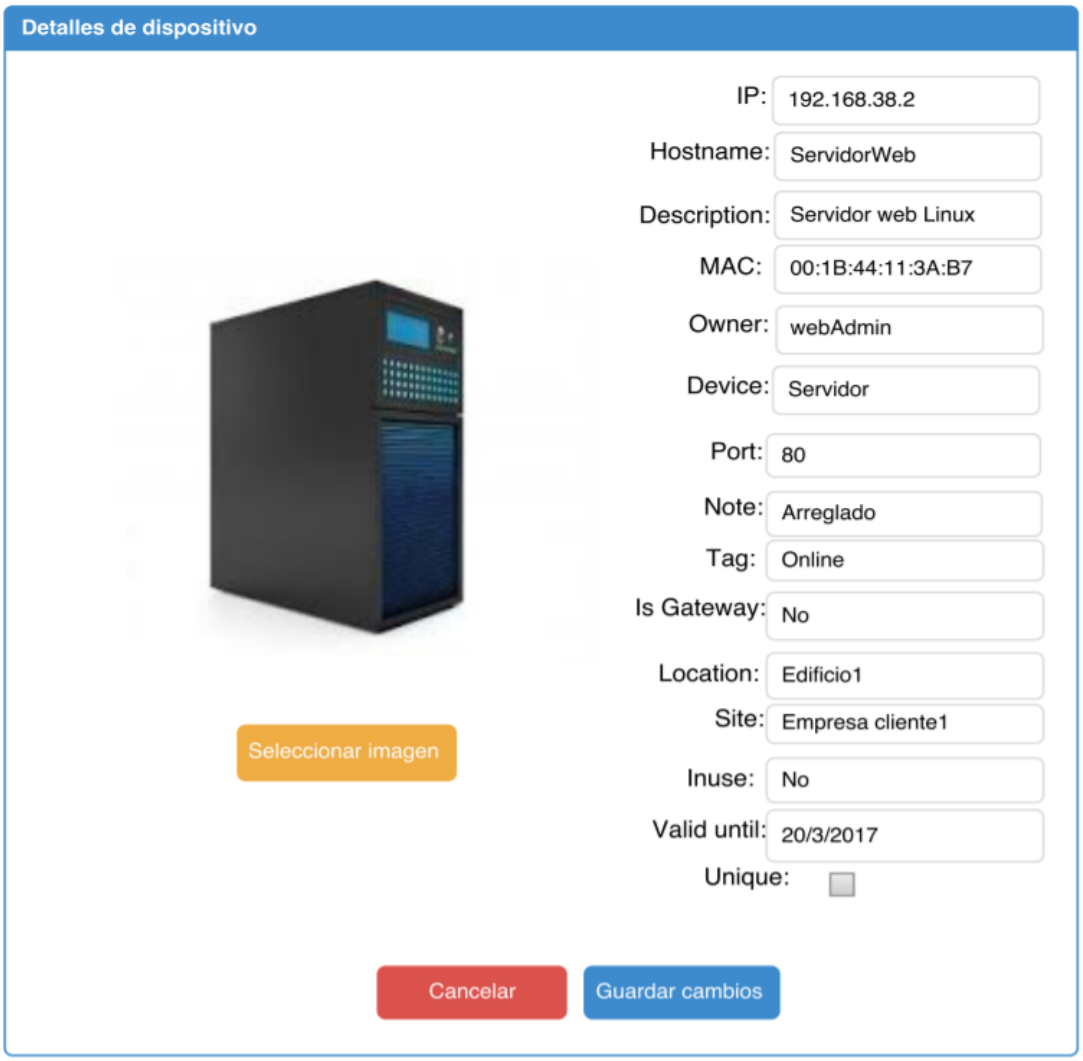

Figura 4.6: Prototipo de la ventana de propiedades de un dispositivo

Finalmente, el resto de ventanas y formularios deben seguir un diseño parecido al planteado en Netbox. De este modo, los usuarios se encontrarán frente a un diseño que ya les resulta familiar, reduciendo la curva de aprendizaje sobre el uso de la aplicación.

Capítulo 5

Detalles de implementación

Contenidos

En el presente capítulo se definen los detalles de implementación de los aspectos más importantes que se han desarrollado durante la fase de implementación. Para ello, se plantean los problemas más importantes que han surgido y se detalla la forma en la cual se han resuelto.

5.1. Arquitectura SPA

Dentro del desarrollo web existe una tendencia creciente denominada «Single Page APP», más conocido como SPA.

Esta tendencia surge con la motivación de acercar más el funcionamiento de una aplicación web al modo en el cual funciona una aplicación de escritorio. Tradicionalmente, toda la lógica de negocio de una aplicación web se realizaba, de forma exclusiva, en la parte del servidor. De esta forma, el navegador o cliente web quedaba libre de carga y el servidor poseía el control total de la lógica de la aplicación. El problema es que cada vez que un usuario realiza una acción debe realizar una petición al servidor y esperar su respuesta para poder mostrar su resultado. Esto supone una gran pérdida de velocidad y rendimiento en la aplicación, pues la velocidad de respuesta ante cada acción del usuario depende de la latencia de red [12].

Con la evolución de la capacidad de cómputo y de la tecnología web asíncrona, se produce un cambio de paradigma. Una aplicación SPA busca aligerar la carga del servidor y realizar el

mínimo de llamadas a éste con el objetivo de mejorar el rendimiento general de la aplicación y la experiencia del usuario. Este hecho supone tener que implementar parte de la lógica de negocio en la interfaz, por lo que en este aspecto, el lado del cliente se vuelve más independiente del servidor, logrando una experiencia parecida al funcionamiento de una aplicación de escritorio.

Así pues, debido a la naturaleza del proyecto, en el cual el usuario está interactuando continuamente con los diagramas y el modelo de datos, se ha optado por implementar una arquitectura SPA. De este modo, solo se realiza una carga de la página. Cada vez que el usuario realiza una acción, la interfaz efectúa llamadas asíncronas (solo cuando sea necesario) y aplicará los cambios correspondientes en la vista mediante JavaScript.

En la aplicación que se ha desarrollado, el modelo de datos de un cliente se carga únicamente una vez. Es decir, cuando el usuario selecciona un cliente, la interfaz recupera todos sus datos mediante llamadas asíncronas al servidor. Esto supone una dura carga inicial que se ve reflejada en un tiempo de espera relativamente largo cuando se selecciona un cliente. Sin embargo, todas las demás operaciones se ejecutan a gran velocidad, mejorando notablemente la experiencia del usuario. La razón por la cual se ha decidido implementar la lógica de esta forma se debe que el equipo de soporte de la empresa suele pasar mucho tiempo interactuando con el modelo de un mismo cliente. Cuando un cliente llama para solucionar una incidencia, el tiempo dedicado a ésta suele superar los 20 minutos. Esto supone que el usuario suele estar todo este tiempo manejando y consultando información de ese cliente para poder solucionar el problema. Por lo tanto, es mucho más eficiente recuperar todos los datos cada vez que se tenga que dar soporte a un cliente y asegurar un buen rendimiento de la aplicación el resto del tiempo.

5.2. Coherencia y asociación de los datos

Probablemente, el problema que ha supuesto un mayor desafío para la materialización del proyecto ha consistido en mantener coherentes los datos del modelo local con los datos del servidor y asociar correctamente cada uno de los modelos a su elemento gráfico correspondiente.

Por una parte, el problema de tener que mantener coherente el modelo local con el modelo residente en la base de datos proviene de las propias características de la arquitectura SPA. A pesar de las muchas ventajas que ofrece esta arquitectura, el hecho de integrar la lógica de negocio en la interfaz también supone la obligación de mantener un modelo de datos local coherente con el modelo remoto. Por lo tanto, cada vez que el usuario realiza una acción que desencadene un cambio en el modelo, éste tiene que ser cambiado en ambos sitios: en el servidor y en el cliente. Netbox posee un modelo de datos muy extenso, cerca de 70 entidades, lo que ha supuesto tener que mantener un modelo local muy grande. Además, la API de Netbox incrusta los datos referenciados. Por ejemplo, cuando se recuperan los datos de una interfaz mediante la API, la respuesta también incluye los datos del dispositivo asociado a la interfaz devuelta. Por lo tanto, hay que tener sumo cuidado a la hora de consultar y modificar datos del modelo local, pues un mismo dato puede aparecer en varios sitios.

Por otra parte, el problema de asociar correctamente el modelo local con los elementos gráficos proviene de la integración de la aplicación con mxGraph. Es importante diferenciar los elementos gráficos, propios del diagrama, con los elementos del modelo de negocio. Tal y como se

ha comentado anteriormente, los vértices y las aristas del diagrama se asocian a alguna entidad del modelo de negocio. Ahora bien, estos elementos son objetos propios de mxGraph con su propio identificador y propieades. En consecuencia, ha sido necesario establecer una forma de relacionar cada uno de los elementos gráficos con la entidad que representa. Cabe recordar, que el diagrama se almacena en forma de plantilla, sin datos del modelo de negocio. En consecuencia, es necesario que ambos elementos queden relacionados, si no, sería imposible relacionar esta plantilla con el modelo de datos.

La solución al primer problema ha consistido en implementar un modelo local que almacene todos los datos del cliente que se encuentran en Netbox y que éste cambie acorde a la base de datos. Además, el modelo contiene datos adicionales que ayudan a facilitar la implementación del diagrama, por ejemplo, se han implementado dos diccionarios que permiten consultar y modificar rápidamente las aristas y los vértices del diagrama. Para lograrlo, se ha implementado toda la lógica de negocio necesaria para mantener coherente el modelo de la interfaz con el modelo de la base de datos. Cada vez que el usuario modifica el modelo, la interfaz realiza una llamada asíncrona para modificar los datos que contiene Netbox. Si la petición se realiza con éxito, se ejecuta el código correspondiente para cambiar el modelo local.

La solución al segundo problema ha consistido en lo siguiente. Aprovechando que ya se tiene un modelo local implementado se ha relacionado cada objeto de este modelo con su correspondiente elemento gráfico, tal y como se puede apreciar en la Figura 5.1. Para ello, se han aprovechado dos funcionalidades que proporciona mxGraph: la posibilidad de cambiar el identificador del elemento gráfico y la posibilidad de asignarle un objeto o valor. De este modo, la interfaz se ha implementado de forma que cuando se dibuja un diagrama a partir de los datos del modelo, primero se dibuja el elemento gráfico, después se le asigna el valor del modelo y finalmente se le cambia el identificador en base a la siguiente convención: el identificador debe ser una cadena de texto compuesta por el tipo de entidad (dispositivo, red, interfaz, etc.), el carácter almohadilla (como separador de campos) y el número identificador del objeto. Con ello se asegura que el elemento gráfico quede asociado al objeto del modelo que representa. De esta forma, al recuperar un diagrama se puede saber con qué objetos se deben rellenar sus elementos gráficos, pues a través del identificador se puede conocer el tipo de entidad del objeto y qué número identificativo posee.

5.3. Recogida de datos y gestión de las promesas

Como se ha comentado anteriormente, la recogida de datos se realiza a través de llamadas a la API REST de Netbox desde la interfaz desarrollada. Cabe tener en cuenta que no todos los datos son exclusivos de un cliente, es decir, existen datos de ámbito global que son compartidos entre todos los clientes introducidos en Netbox. Algunos ejemplos son: los modelos de dispositivo, los proveedores, los tipos de circuito, etc. En consecuencia, no resulta eficiente cargar estos datos cada vez que se realicen llamadas para obtener los datos de un cliente. Por lo tanto se ha optado por repartir el conjunto de llamadas asíncronas para la recogida de datos en dos etapas. En la primera etapa se recogen todos los datos que son compartidos por todos clientes. Estas llamadas se realizan cuando se carga la aplicación en el navegador. Una vez se tienen los datos comunes, el usuario puede elegir un cliente de una lista desplegable que incorpora una sencillo buscador (véase la Figura 5.2). Cuando un cliente es seleccionado se realiza el segundo conjunto

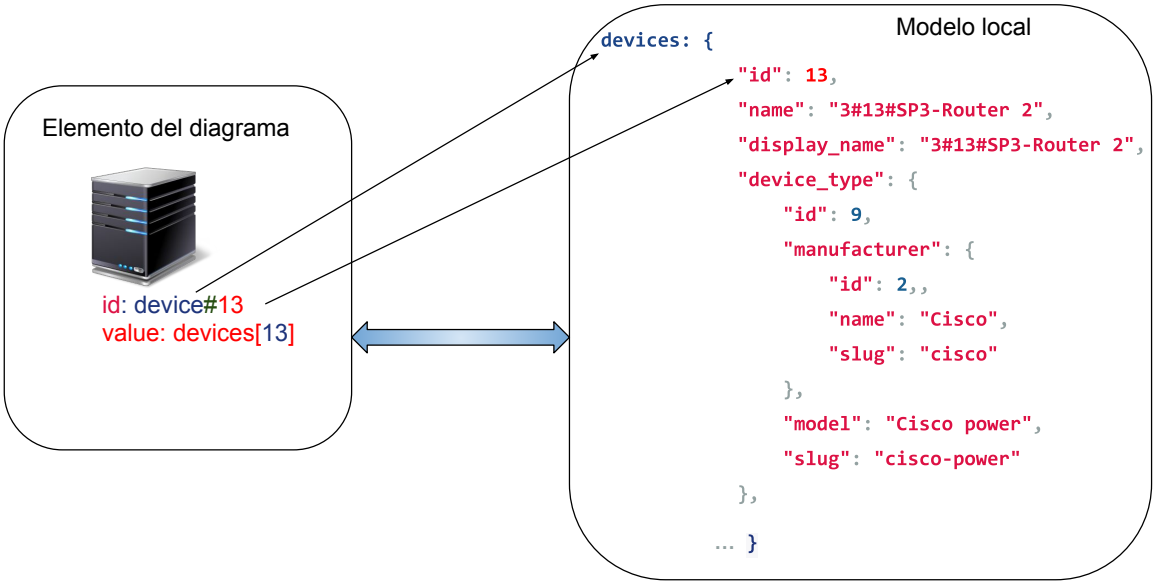

Figura 5.1: Ejemplo de como se asocia el elemento gráfico con el modelo de datos.

de llamadas al servidor, en el cual solo se recuperan los datos asociados al cliente. De este modo se evita reiterar la recogida de datos que ya han sido recogidos anteriormente, aumentando la eficiencia de la aplicación.

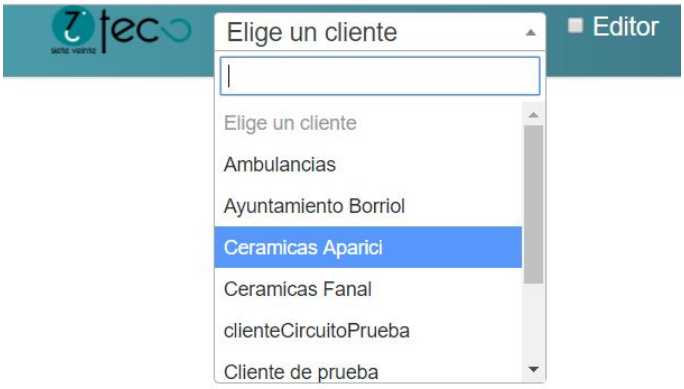

Figura 5.2: Lista de clientes desplegable que incorpora un buscador

Ahora bien, las llamadas asíncronas suponen un problema, pues es imposible saber con seguridad cuándo se van a recibir los datos o si éstos van a estar disponibles. Esto supone un aspecto vital de la aplicación, pues si se intenta dibujar el diagrama cuando todavía faltan datos se puede producir un error que impida su utilización. Además, hay llamadas que se deben realizar una vez se reciben otros datos, es decir, existen dependencias a la hora de recoger la información. Por ejemplo, se necesitan tener todos los circuitos para poder recuperar todos sus extremos y esperar a tener todos estos extremos para que se puedan dibujar en el diagrama. Es por ello que se ha optado por una solución orientada a promesas.

Las promesas son objetos JavaScript que representan un valor que puede estar o no disponible a lo largo del tiempo [3]. Este valor puede ser el resultado de la ejecución de una función

```
//Mapeamos las promesas para que el código se ejecute cuando las tengamos todas
      var promesas = self.circuits.map((circuit)=>{
        return axios.get(self.baseURL+'/api/circuits/circuit-terminations/',{
            params: {circuit_id: circuit.id} })
      });
      //Una vez se han recibido todos los datos los añadimos al modelo local
      Promise.all(promesas).then((values)=>{
        for(var valor of values){
          let terminations = valor.data.results;
          let idCircuit = terminations[0].circuit.id;
          //Creamos una entrada al diccionario que relacione los extremos
          //con los circuitos.
          self.circuitsTerminations[idCircuit] = terminations;
        }
      });
```

Figura 5.3: Ejemplo de control de recogida de datos mediante promesas JavaScript

asíncrona o un error, de modo que una promesa puede estar estar en un estado pendiente, aceptada o rechazada. La ventaja de las promesas es que permiten al programador indicar que lógica ejecutar una vez se recibe la respuesta.

De este modo, las promesas han sido utilizadas cuando se ha necesitado la disponibilidad de cierto conjunto de datos recibidos de forma asíncrona. Para ello, se ha aprovechado el hecho de que Axios funciona mediante promesas, pues cuando éste realiza una llamada, retorna esta clase de objeto. Sabiendo esto, se puede crear una lista de las promesas recibidas por Axios para establecer una función a ejecutar una vez se reciben todos los valores. Con el objetivo de ejemplificar esta lógica, en la Figura 5.3 se muestra una pieza de código JavaScript que se ha utilizado para obtener los extremos de los circuitos.

5.4. Capas de visualización

Como se ha comentado anteriormente, los diferentes diagramas de un cliente se asocian a una capa de un sistema de capas predefinido. Esta funcionalidad permite dividir y clasificar la visualización de una infraestructura informática en diferentes diagramas. De este modo el equipo de soporte puede elegir qué información visualizar según la tarea que tenga que realizar.

En un principio se había pensado en implementar el sistema de capas como un mero sistema de clasificación, en el cual un usuario puede crear diagramas y asociarlos a las diferentes capas del sistema. Esto supone un problema, pues queda a la elección del usuario qué elementos mostrar en cada una de las capas. Si cada usuario piensa de forma diferente, podría haber diagramas, de una misma capa, que no se documenten de la misma forma. Cabe recordar que uno de los objetivos del proyecto consiste en homogeinezar la forma de documentar las infraestructuras de red de los clientes de la empresa. Debido a este problema se ha decidido implementar un sistema de capas que limite la capacidad de elección de los usuarios a través de la automatización de la asociación de los diagramas con las capas.

Este sistema se ha implementado de forma que existen varias capas predefinidas por el

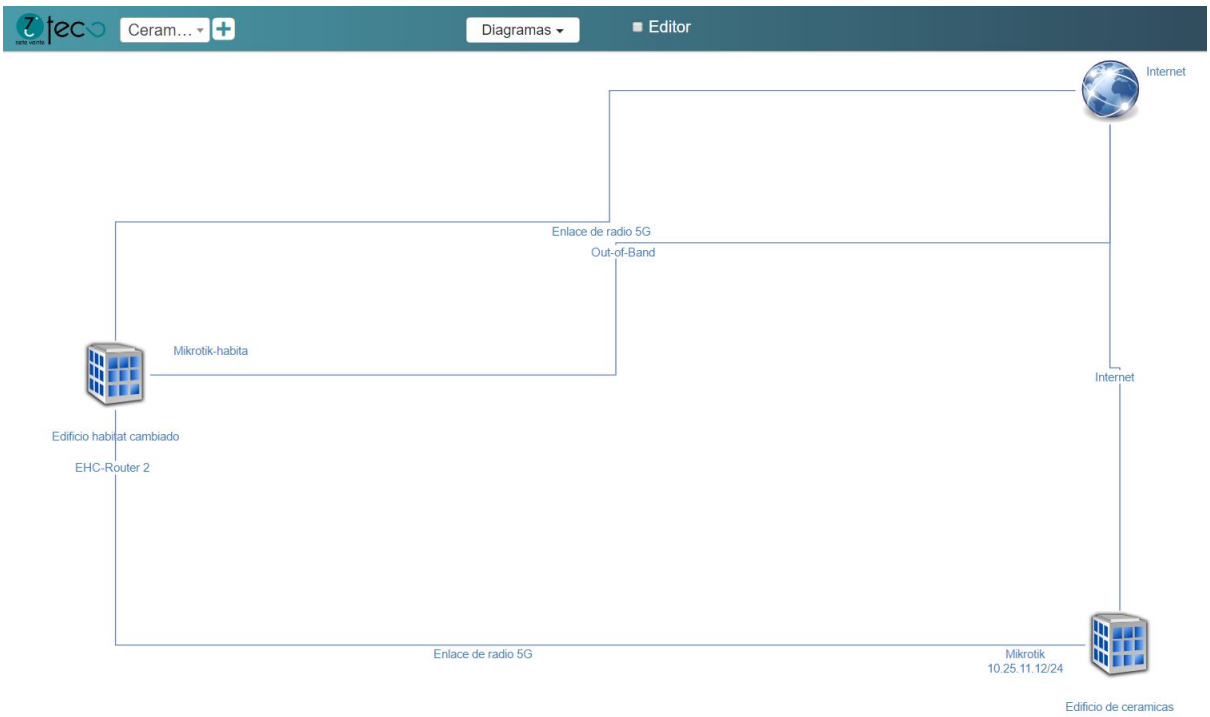

Figura 5.4: Ejemplo de visualización de un diagrama de capa de circuitos

desarrollador. Las capas que se han decidido implementar, para el alcance del proyecto, son 3: una capa de circuitos, una de enrutamiento y una capa de red. Además, el tipo de información que se muestra en cada una de las capas también está predefinido y automatizado por el sistema. Por lo tanto, las capas ya no actúan únicamente como un modo de clasificación o etiquetado, sino que poseen su propio significado y funcionalidad.

5.4.1. Capa de circuitos

La capa de circuitos supone la visualización más global de la infraestructura de red de un cliente. Tal y como se puede observar en la Figura 5.4, en esta capa el usuario puede visualizar rápidamente la organización estructural de un cliente, lo que permite identificar, de un simple vistazo, los canales de comunicación de la organización.

Para ello, se representan los sitios o edificios que contiene un cliente y los circuitos que los conectan. Además, se refleja la salida a internet la cual es representada, dentro del sistema, como un sitio especial compartido por todos los clientes. A la hora de representar los circuitos se muestra información sobre los dispositivos de enrutamiento donde se conectan. Concretamente, esta información corresponde al nombre del dispositivo y a su dirección IP. Además, se muestra el tipo de circuito que conectan los sitios. Así pues, en esta capa quedan reflejados los canales de comunicación más externos de un cliente, el tipo de canal por donde pasan los datos y los dispositivos, con sus correspondientes direcciones IP, que se encargan del enrutamiento.

Con el objetivo de poder mostrar toda esta información ha sido necesario utilizar los modelos de Netbox que se detallan a continuación.

Sites: Son los diferentes sitios o edificios que contiene la organización del cliente.

Circuits: Los diferentes canales de comunicación que conectan sitios.

Circuits types: Representan el tipo de canal de los circuitos.

Circuits terminations: Modelo que relaciona un circuito con los dos sitios que conecta.

Devices: Los diferentes dispositivos que posee un cliente. Para esta capa se ha realizado un filtrado que obtiene únicamente información sobre los dispositivos de enrutamiento asociados a los extremos de los circuitos.

IP addresses: Modelo que almacena las diferentes direcciones IP asignadas a las interfaces de un dispositivo.

La capa de circuitos es la capa principal de cada uno de los clientes, es decir, cuando el usuario selecciona un cliente, se muestra esta capa. Por esta razón, para cada cliente no puede existir más de un diagrama de este nivel, pues el diagrama de capa de circuitos es el punto de partida desde el que se generan los diagramas de las otras capas. Esto se debe a que la aplicación se ha implementado de forma que se genere un diagrama de capa de enrutamiento y un diagrama de capa de red para cada uno de los sitios creados en el diagrama de capa de circuitos. Así pues, el usuario puede tener una visión global de la estructura organizacional de un cliente y, a partir de ahí, ir adentrándose en cada uno de los sitios que posee.

5.4.2. Capa de enrutamiento

La capa de enrutamiento ha sido la capa más compleja de implementar. La característica principal de esta capa es que no permite añadir nuevos elementos gráficos en el diagrama, es decir, es una capa totalmente autocalculada a partir de los datos introducidos en la capa superior e inferior.

En esta capa se refleja todo el tráfico y elementos, a nivel de enrutamiento, existentes en un sitio de un cliente (véase la Figura 5.5). Para ello se dibujan los dispositivos de enrutamiento, las redes a las cuales pertenecen y los sitios. También se dibujan las relaciones que se pueden establecer entre estos elementos, las cuales se detallan a continuación.

Circuitos: Estas relaciones representan los circuitos que conectan con otros sitios. Cabe mencionar que un circuito conecta dos sitios, pero como en esta capa de visualización se representa la información contenida en un sitio, gráficamente se muestra la conexión entre el dispositivo de enrutamiento y el sitio con el cual conecta. En estos tipos de circuitos se muestran las interfaces de los dispositivos de enrutamiento que intervienen en la conexión.

Conexiones físicas entre dispositivos: Estas relaciones representan las conexiones de red existentes entre los dispositivos de enrutamiento dentro del sitio. Así pues, consisten en relaciones que conectan un dispositivo con otro, mostrando información de las interfaces donde se establecen las conexiones.

Relaciones de redes con dispositivos: Estas relaciones representan la pertenencia de los dispositivos de enrutamiento a las redes, según sus direcciones IP. Cabe mencionar que no existe un modelo en Netbox que refleje este tipo de conexiones y que ha sido necesario implementar cierta lógica de filtrado para poder representarlas. Básicamente, se recorre cada una de las interfaces de un dispositivo para ver las direcciones IP que tienen asignadas y se realiza un mapeo entre dispositivos y redes.

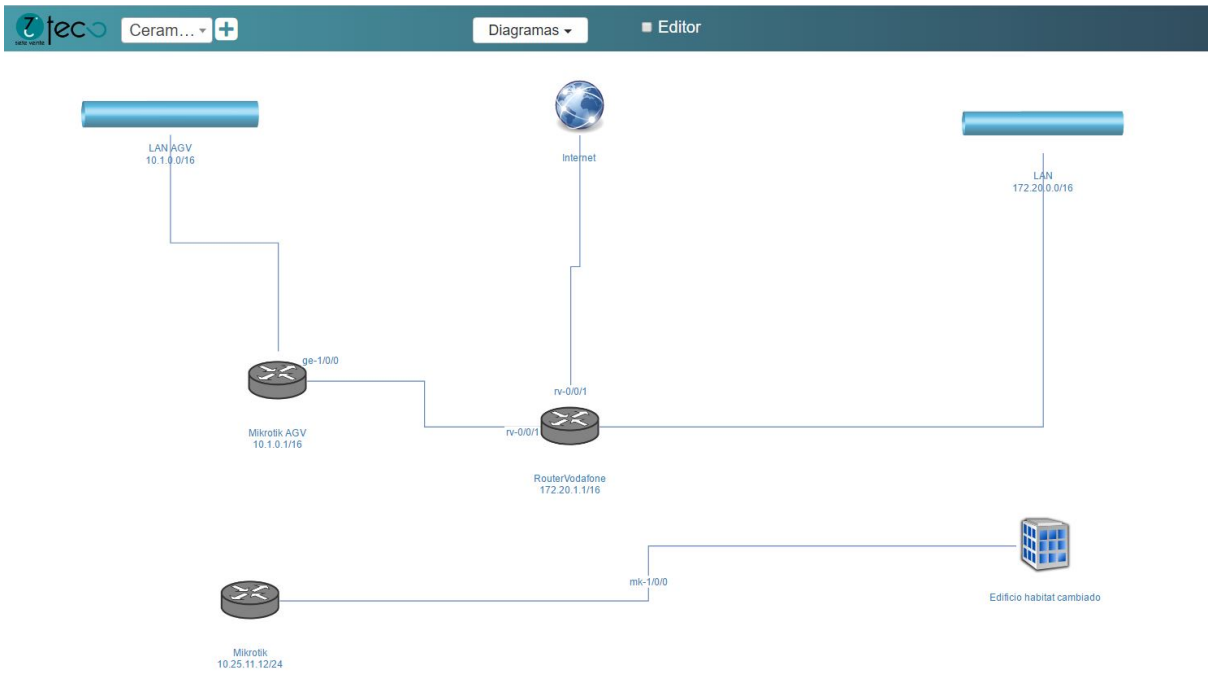

Figura 5.5: Ejemplo de visualización de un diagrama de capa de enrutamiento

Para poder dibujar esta capa se hace uso de los modelos de Netbox que se definen a continuación.

Sites: En esta capa los sitios se filtran en base a los dispositivos de enrutamiento que intervienen en las conexiones.

Circuits: Se necesitan para establecer las conexiones entre sitios y dispositivos.

Circuits terminations: Es donde se encuentra la información que asocia un circuito con el site y los dispositivos.

Devices: En esta capa se realiza un filtrado de los dispositivos de enrutamiento del sitio asociado al diagrama.

IP addresses: Se necesitan para relacionar la interfaz del dispositivo con la dirección IP que tiene asignada.

Interfaces: Modelo que representa las diferentes interfaces o bocas de un dispositivo. Se necesita para relacionar los dispositivos con las redes, a las cuales pertenecen, y con otros dispositivos.

Prefixes: Modelo que representa las redes de un cliente. También se pueden asociar a los sitios, lo que permite realizar el filtrado.

Connections: Modelo que representa las conexiones físicas entre las interfaces de los dispositivos. Este modelo se necesita para dibujar las conexiones existentes entre los dispositivos de enrutamiento.

Finalmente, cabe mencionar la complejidad de esta capa para reflejar los cambios que se realizan en otras. El problema recae en que para poder dibujarla se necesita realizar muchas operaciones de filtrado sobre los modelos. En consecuencia, la lógica implementada para reflejar los cambios que se realizan en otras capas se complica notablemente.

5.4.3. Capa de red

La capa de red corresponde al nivel de representación más básico de la red de un cliente. En esta capa se dibujan todos los dispositivos contenidos en un sitio y las conexiones físicas existentes entre sus interfaces (véase la Figura 5.6). En estas conexiones se muestra información sobre las interfaces de conexión, el tipo de conexión y el tipo de tráfico que pasa por este enlace. Por otra parte, respecto a los dispositivos se muestra información sobre el nombre de dispositivo y su dirección IP principal, en caso de tener una. De este modo, para poder dibujar esta capa se han utilizado los modelos de Netbox que se exponen a continuación.

Devices: En esta capa se realiza un filtrado de los dispositivos del cliente por el sitio asociado a la capa.

IP addresses: Se necesitan para poder mostrar la dirección IP junto al dispositivo.

Interfaces: Se necesitan para mostrar los puntos de conexión de las conexiones entre los dispositivos.

Connections: Necesario para dibujar las conexiones entre los dispositivos.

Vlans: Modelo que representa las VLAN existentes en la red de un cliente. Este modelo es necesario para identificar el tipo de tráfico que circula por cada conexión.

Interface-vlans: Es el modelo que se ha diseñado en el módulo personalizado. Este modelo permite asociar las VLAN con las interfaces de los dispositivos.

Finalmente cabe destacar que las dos interfaces que actúan como punto de conexión pueden poseer diferentes VLAN asociadas. Por lo tanto, para establecer el tráfico que circula entre una conexión se realiza un mapeo que permite identificar las VLAN coincidentes entre las dos interfaces.

5.5. Editor y creación de nuevos elementos gráficos

El editor es un elemento que se ha implementado a través de la clase «mxEditor» que proporciona la biblioteca de mxGraph. Este elemento, tal y como se muestra en la Figura 5.7,

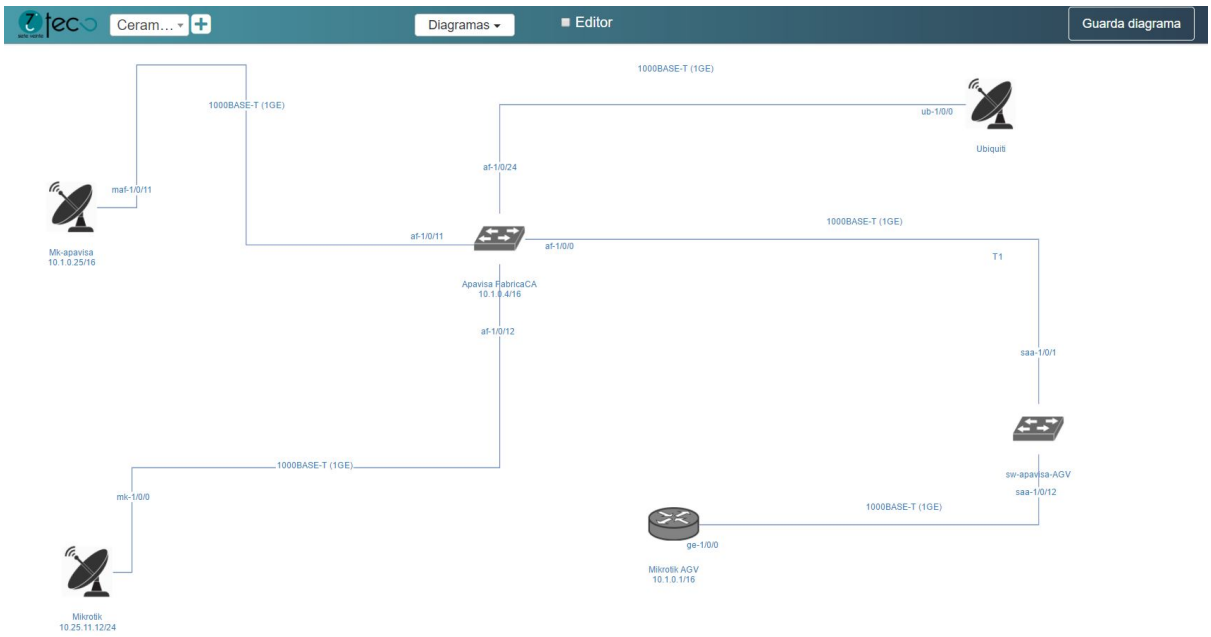

Figura 5.6: Ejemplo de visualización de un diagrama de capa de red

permite definir prototipos de elementos gráficos, a los cuales se puede asociar información, un estilo y eventos. El editor se ha implementado de forma que se ha asociado un evento que se

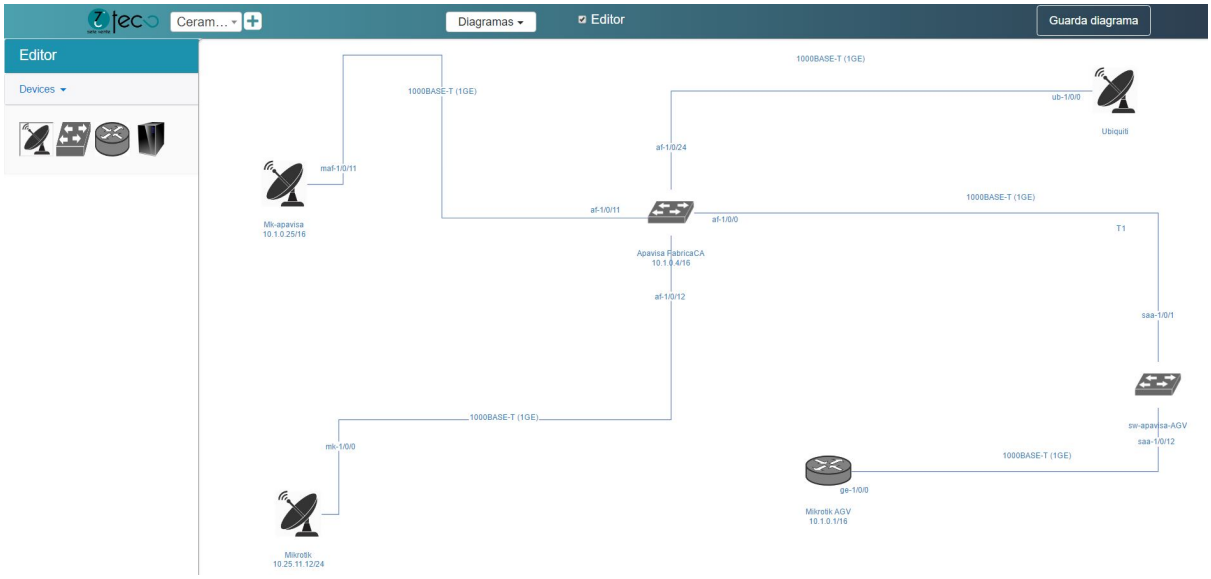

Figura 5.7: Ejemplo de editor asociado a la capa de red

emite cuando se arrastra uno de estos elementos y se suelta dentro del diagrama. Cuando este evento es lanzado se abre un formulario para rellenar la información del nuevo elemento que se desea crear. Una vez rellenados los campos del formulario se realiza la petición a la base de datos para insertar el nuevo elemento y, según la respuesta recibida, se muestra o no en el diagrama.

Como se ha comentado anteriormente, los elementos que se pueden dibujar en un diagrama vienen determinados por la capa de visualización. Por lo tanto, también ha sido necesario limitar qué elementos se pueden generar mediante el editor según la capa en la cual se encuentre el

usuario. La solución ha consistido en implementar dos editores que se muestran u ocultan según la capa. Así pues, en el editor de la capa de circuitos únicamente se pueden crear sitios, mientras que en el editor de la capa de red se pueden crear dispositivos según el rol que desempeñan.

Por otra parte, las conexiones entre elementos se deben crear mediante formularios que se abren a través de una opción de un menú desplegable. De este modo, para generar una conexión es necesario crear los elementos que se desean conectar mediante el editor y después crear la conexión a través de esta opción. De este modo, se posee un mayor control sobre el manejo de conexiones, evitando incoherencias entre los elementos que se muestran en el diagrama y los datos que se encuentran en el modelo.

5.6. Dibujar y rellenar diagramas

Como se ha comentado anteriormente, la información que se muestra en los diagramas en cada una de las capas depende de estas últimas. Ahora bien, hay un primer momento en el cual puede no existir ningún diagrama para un cliente, pero en el cual puede haber datos introducidos desde Netbox. Por lo tanto, con el objetivo de solucionar este problema, ha sido necesario establecer dos procedimientos diferenciados para dibujar el modelo de los clientes.

El primero de estos procedimientos consiste en dibujar un diagrama directamente a partir de los valores contenidos en Netbox. En consecuencia, no existe un diagrama a partir del cual se rellenan sus valores, sino que los elementos gráficos son creados a partir del modelo.

Por otra parte, el segundo procedimiento consiste en justamente lo contrario. Esta vez, se recorre el diagrama, en forma de plantilla, y se rellenan todos sus valores gracias a la vinculación de los identificadores de los elementos gráficos con los identificadores del modelo. Por lo tanto, en este procedimiento no se generan los elementos gráficos, simplemente son rellenados. Sin embargo, supone un grado de dificultad añadida, pues es necesario buscar incoherencias entre los elementos gráficos y el modelo para poder corregir el diagrama en su caso.

De esta forma, es posible combinar estas dos funciones para poder mostrar el diagrama según el contexto. Concretamente, la lógica que se ha implementado consiste en que si la interfaz realiza una petición para recuperar un diagrama de un cliente y éste no existe se utiliza el primer procedimiento descrito. En caso contrario, se utiliza el segundo. Finalmente, cabe destacar que estos procedimientos actúan de diferente forma en base a las capas, pues el modo de rellenar los datos o de dibujar los elementos cambian entre éstas.

5.7. Integración de la interfaz con Netbox

Para el despliegue de la aplicación se ha decidido integrar la interfaz dentro de Netbox, permitiendo proporcionar todo el servicio como un paquete que incluye la versión modificada de Netbox y la interfaz que se ha desarrollado.

Esto proporciona dos grandes ventajas. Por una parte, ya no es necesario tener que ejecutar

dos servicios diferentes (uno para la interfaz y otro para Netbox), sino que toda la aplicación se sirve desde un único servicio. Así pues, se consigue una sensación de que la aplicación que se ha desarrollado forma parte de Netbox. Por otra parte, se puede aprovechar la lógica de autenticación y autorización que ya está implementada en esta tecnología. Que, entre otras opciones, permite configurar una sistema de autenticación basado en un protocolo que permite el acceso a un servicio de directorio activo (LDAP) [6]. De esta forma, los usuarios de la empresa pueden utilizar las cuentas de su directorio activo para acceder a la aplicación, evitando tener que crear las cuentas manualmente.

Todo esto se ha conseguido mediante la integración de la interfaz desarrollada con Netbox. Por una parte, se ha añadido el fichero HTML de la aplicación como una plantilla que se sirve cuando se intenta acceder a cierta URL. Este fichero contiene la clave de autenticación que se debe añadir en las cabeceras de las peticiones para introducir, modificar y borrar datos a través de la API REST. Por otra parte, la biblioteca de mxGraph, el fichero JavaScript que contiene la lógica de la aplicación y los ficheros que le dan estilo se sirven como ficheros estáticos que son llamados desde el fichero HTML. Para ello ha sido necesario definir la nueva URL, la cual cuelga de la aplicación de tec720, e indicar qué fichero servir ante una petición. Todo esto proporciona seguridad a la aplicación, pues para poder acceder a esta página se necesita ser un usuario autenticado en Netbox.

Capítulo 6

Conclusiones

Contenidos

En este capítulo se plantean las conclusiones a las cuales se ha llegado tras la elaboración del proyecto. Además, se plantean que mejoras se podrían llevar a cabo en el futuro para poder mejorar su funcionamiento.

6.1. Reflexiones personales

La materialización de este proyecto ha consistido en el desarrollo de una aplicación que permita, al equipo de soporte de la empresa, documentar y consultar información sobre las infraestructuras informáticas de sus clientes.

Respecto a la consecución de todos los objetivos, éstos no se han podido cumplir de la forma más favorable. Esto se ha debido a las complicaciones y desviaciones que han surgido durante la realización del proyecto, las cuales han supuesto un gran coste temporal. Sin embargo, se han conseguido cumplir los objetivos más importantes, como la visualización basada en un sistema de capas o la centralización de los diagramas, por lo que los objetivos no cumplidos se pueden considerar como mejoras futuras de una aplicación que ya es funcional.

En relación a la metodología de trabajo, opino que ha sido muy acertada. Las reuniones semanales y la utilización de una herramienta de gestión de tareas como MeisterTask permiten realizar una planificación rápida y sencilla a corto plazo. Sin embargo, pienso que una planificación global más detallada hubiera sido de mucha utilidad, pues en muchas ocasiones se ha perdido tiempo corrigiendo y cambiado funcionalidades que, de haber estado definidas desde un principio, se hubieran implementado correctamente a la primera.

Respecto a la aplicación que he desarrollado, considero que no proporciona la suficiente robustez para el cometido que tiene que llevar a cabo. Esto se debe al limitado control que se

posee sobre la base de datos, pues la lógica que se ejecuta en la parte del servidor está limitada por la API de Netbox. En consecuencia, no se pueden utilizar operaciones transaccionales ni llevar un buen control de la concurrencia, suponiendo que a la larga puedan existir incoherencias que conduzcan a errores. Además, mxGraph ha supuesto ser una biblioteca muy difícil de utilizar, pues su gran flexibilidad lo es a costa de una gran dificultad a la hora de utilizarla.

Finalmente, opino que la experiencia personal ha sido realmente buena. Por una parte, he aprendido a utilizar y combinar tecnologías que desconocía totalmente. Además, todos los obstáculos y dificultades que han surgido durante el desarrollo del proyecto me han ayudado a crecer profesionalmente, sobretodo a la hora de plantear y planificar la resolución de problemas. Por otra parte, el entorno de trabajo ha sido realmente favorable, gracias a un equipo de profesionales que en todo momento han estado dispuestos a echar una mano cuando ha sido necesario. Adicionalmente, las reuniones semanales, donde se ha simulado el trato con el cliente, me han parecido una gran experiencia para entender los conflictos de intereses que surgen durante las negociaciones entre desarrollador y cliente.

6.2. Mejoras futuras

Debido al tiempo que se ha tenido para planificar y desarrollar el proyecto y a todas las dificultades y desviaciones que han surgido, no se ha podido desarrollar toda las funcionalidades necesarias para que la aplicación pueda realizar su cometido correctamente. Por esta razón, a continuación se plantean algunas de las posibles mejoras que se podrían llevar a cabo para solucionar este problema.

Gestión de credenciales y conexiones remotas. Como se ha comentado anteriormente, debido a las desviaciones surgidas durante el desarrollo del proyecto, se ha decidido prescindir de la funcionalidad que permite gestionar credenciales y realizar conexiones remotas con algunos de los dispositivos de los clientes. Por lo tanto, este requisito se ha propuesto como una mejora futura.

Historial de documentación. Otra posible mejora podría consistir en el desarrollo de un historial que registre todas las acciones que realizan los usuarios en el sistema.

Implementación de un panel de administración. Una posible mejora futura podría consistir en la implementación de un panel de administración que permita realizar acciones típica de Netbox, como la gestión de los modelos y de los usuarios. De esta forma, la interfaz que se ha desarrollado iría sustituyendo, progresivamente, a toda la aplicación de Netbox.

Desarrollo de una API REST propia. El desarrollo de una API REST propia supondría un mayor control sobre la lógica que reside en la parte del servidor. La idea sería implementar una API que actuara como intermediaria entre la interfaz y la API REST de Netbox, o una API que actuara directamente sobre la base de datos. Esta última podría solventar el problema de

las transacciones, de forma que se aumentaría la tolerancia a fallos. Además, se podría resolver el problema de gestionar diferentes usuarios de forma concurrente cuando estos intentan acceder a los datos de un mismo cliente.

Mejoras generales de la aplicación. Debido al poco tiempo que se ha tenido no se ha podido implementar algunas funcionalidades típicas orientadas a mejorar la experiencia del usuario. Por lo tanto, mejoras como la validación de formularios, un mejor *feedback* para el usuario o un mejor diseño del estilo se podrían llevar a cabo en el futuro.

Mejora del sistema de visualización por capas. Habría que extender el sistema de visualización por capas, pues éste es muy limitado. Con el objetivo de hacer que la aplicación sea más escalable, habría que incluir la posibilidad de que el usuario pueda crear sus propias capas y diagramas. Además, habría que cubrir representaciones de problemas típicos a los cuales se enfrenta el equipo de soporte, como puede ser la representación de las rosetas de conexión o el flujo de las copias de seguridad.

De esta forma, considero que con la implementación de todas las mejoras descritas, se podría disponer de una aplicación robusta y funcional preparada para desplegar en producción.

Bibliografía

[1] David Benson. mxgraph user manual javascript client. https://jgraph.github.io/mxgraph/docs/manual.html. [Consultado el 5 de junio de 2017].

[2] Equipo de ingenieros de Digital Ocean. Documentación oficial de netbox. http://netbox.readthedocs.io/en/stable/. [Consultado el 6 de junio de 2017].

[3] Colaboradores de Mozilla Developer Network. Promise. https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Promise. [Consultado el 14 de junio de 2017].

[4] Colaboradores de Wikipedia. Cliente-servidor. https://es.wikipedia.org/w/index.php?title=Cliente-servidor&oldid=99684071. [Consultado el 11 de junio de 2017].

[5] Colaboradores de Wikipedia. Jquery. https://es.wikipedia.org/w/index.php?title=JQuery&oldid=98734549. [Consultado el 5 de junio de 2017].

[6] Colaboradores de Wikipedia. Protocolo ligero de acceso a directorios. https://es.wikipedia.org/w/index.php?title=Protocolo_Ligero_de_Acceso_a_Directorios&oldid=98704302. [Consultado el 20 de junio de 2017].

[7] Saul García M. *Django, la guía definitiva: desarrolla aplicaciones web de una forma rápida y sencilla*. Django Software Corporation, 2015.

[8] Mohammed J. Kabir. *La biblia del servidor Apache*.

[9] Stefan Krause. Js web frameworks benchmark round 6. http://www.stefankrause.net/wp/?p=316. [Consultado el 5 de junio de 2017].

[10] Luis Felipe López Acevedo. *Tutorial de PostgreSQL*, 2016.

[11] JavaScript Rising Stars. 2016 javascript rising stars. https://risingstars2016.js.org/#framework. [Consultado el 5 de junio de 2017].

[12] Javier Tejero. Spa, un paradigma de arquitectura de aplicaciones web en auge. https://cink.es/blog/author/javier-tejero/. [Consultado el 13 de junio de 2017].

[13] Nick Uraltsev. Axios. https://github.com/mzabriskie/axios. [Consultado el 5 de junio de 2017].

[14] Evan You. Documentación oficial de vue.js. https://vuejs.org/v2/guide/. [Consultado el 5 de junio de 2017].

Anexo A

Definición inicial de requisitos

En este apéndice se reproduce el documento con la definición de requisitos de los casos de uso de la Figura 4.1. Cabe tener en cuenta, que la definición de requisitos que se muestra en este capítulo se realizó en base a los casos de uso planteados en la etapa inicial del proyecto. Por lo tanto, las funcionalidades descritas no corresponden totalmente con las funcionalidades que posee la versión final del sistema.

CU01: Gestionar clientes

Este caso de uso comprende las acciones por parte del equipo de soporte sobre la gestión de los clientes, es decir las empresas que contratan los servicios de 720Tec. De esta forma el sistema debe permitir a los usuarios realizar acciones típicas de gestión como añadir, modificar, borrar, consultar o buscar clientes.

CU01.1: Añadir cliente al sistema

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU01.1:** Añadir cliente al sistema.
Área temática	Gestión de clientes.
Suceso empresarial	Un nuevo cliente contrata los servicios de 720Tec respecto al diseño, provisión y mantenimiento de infraestructuras informáticas. Para llevar a cabo la documentación de dicha infraestructura es necesario añadir la cliente en el sistema.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El equipo de soporte introduce un nuevo cliente en el sistema.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Que el cliente a introducir no exista en la base de datos.
Resultado de la terminación	1. El cliente se ha añadido satisfactoriamente. 2. No se ha podido añadir el cliente porque ya existe un cliente con el mismo nombre identificativo. 3. No se ha podido añadir el cliente porque no hay conexión con la base de datos.

1

Condición que puede afectar al resultado de terminación	1. No existe el cliente y hay conexión con la base de datos. 2. El cliente ya existe y hay conexión con la base de datos. 3. El cliente puede o no puede existir, pero no hay conexión con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte elige una opción de “Agregar nuevo cliente”. 2. El cliente web abre un formulario para introducir los datos del cliente. 3. El usuario rellena el formulario y pulsa en un botón de “Agregar nuevo cliente”. 4. El sistema lleva a cabo una validación. a. Los campos del formulario son válidos. b. Los campos del formulario no son válidos. 5. Si los campos son válidos el sistema realiza una llamada a la API para introducir el nuevo cliente. En caso contrario muestra un mensaje de error. 6. Después de la llamada el sistema muestra un mensaje informando al usuario de si se ha introducido el cliente a la base de datos o si no se ha podido realizar esta acción.
Asociaciones de casos de uso	● **CU01.4:** Listar clientes.
Rastreabilidad	Documentación.
Resumen de entradas	Datos relativos al cliente, como nombre de la empresa, campo identificativo, contacto, localización etc.
Resumen de salidas	Mensaje de feedback, lista de nombre de clientes.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 3
Notas del caso de uso	

CU01.2: Modificar cliente del sistema

Campo de caso de uso	Descripción
Nombre del caso de uso	**CU01.2:** Modificar cliente del sistema.
Área temática	Gestión de clientes.
Suceso empresarial	Un cliente realiza algún cambio en su organización, como un traslado, cambio de nombre de la empresa o cambio de contacto. En consecuencia, el personal de soporte, para mantener la integridad de la información respecto al cliente, debe modificar sus datos en el sistema.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la información de una forma íntegra y actualizada respecto a sus clientes.
Visión general del caso de uso	El sistema permite al personal de soporte modificar la información existente asociada a un cliente.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. El cliente a modificar debe existir en el sistema.
Resultado de la terminación	1. El cliente se ha modificado satisfactoriamente. 2. No se ha podido modificar el cliente porque no existe. 3. No se ha podido modificar el cliente porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de	1. Existe el cliente y hay conexión con la base de datos. 2. El cliente no existe y hay conexión con la base de datos. 3. El cliente puede o no existir, pero no hay conexión con la

terminación	base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte elige una opción de “Modificar cliente” en la lista de clientes. 2. El sistema abre un formulario con los datos existentes del cliente y permite al usuario modificar estos datos. 3. El usuario rellena el formulario y pulsa en un botón de “Guardar cambios”. 4. El sistema lleva a cabo una validación. a. Los campos del formulario son válidos. b. Los campos del formulario no son válidos. 5. Si los campos son válidos el sistema realiza una llamada a la API para modificar los datos del cliente. En caso contrario muestra un mensaje de error. 6. Después de la llamada el sistema muestra un mensaje informando al usuario sobre si se ha podido o no modificar los datos del cliente.
Asociaciones de casos de uso	● **CU01.4:** Listar clientes.
Rastreabilidad	Documentación.
Resumen de entradas	Datos relativos al cliente, como nombre de la empresa, campo identificativo, contacto, localización etc.
Resumen de salidas	Mensaje de feedback, lista de nombre de clientes.
Índice de utilización (1/10)	● **Satisfacción:** 8 ● **Importancia:** 8 ● **Frecuencia:** 2
Notas del caso de uso	

CU01.3: Eliminar cliente del sistema

Campo de caso de uso	Descripción
Nombre del caso de uso	**CU01.3:** Eliminar cliente del sistema.
Área temática	Gestión de clientes.
Suceso empresarial	Un cliente decide prescindir de los servicios ofrecidos por 720Tec. En ese caso es necesario eliminar la información existente del cliente.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la información de una forma íntegra y actualizada respecto a sus clientes.
Visión general del caso de uso	El sistema permite al personal de soporte eliminar la información existente asociada a un cliente.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. El cliente a eliminar debe existir en el sistema.
Resultado de la terminación	1. El cliente se ha eliminado satisfactoriamente. 2. No se ha podido eliminar el cliente porque no existe. 3. No se ha podido eliminar el cliente porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Existe el cliente y hay conexión con la base de datos. 2. El cliente no existe y hay conexión con la base de datos. 3. El cliente puede o no existir pero no hay conexión con la base de datos.
Descripción del	1. Uno de los miembros del equipo de soporte elige una

caso de uso	opción de “Eliminar cliente” en la lista de clientes. 2. Al ser una acción importante el sistema muestra un mensaje de verificación de la acción. 3. Si los campos son válidos el sistema realiza una llamada a la API para eliminar el registro del cliente. En caso contrario muestra un mensaje de error. a. El sistema también debe eliminar todos los elementos de red y diagramas asociados al cliente. 4. Después de la llamada, el sistema muestra un mensaje informando al usuario sobre si se ha podido o no eliminar los datos del cliente.
Asociaciones de casos de uso	● **CU01.4:** Listar clientes. ● **CU02.3:** Eliminar elemento.
Rastreabilidad	Documentación.
Resumen de entradas	Identificador del cliente que el sistema debe eliminar.
Resumen de salidas	Mensaje de feedback, lista de nombre de clientes.
Índice de utilización (1/10)	● **Satisfacción:** 8 ● **Importancia:** 10 ● **Frecuencia:** 2
Notas del caso de uso	Al eliminar un cliente es importante eliminar recursivamente todos los elementos, datos y diagramas asociados al mismo, pues si no se podrían generar inconsistencias.

CU01.4: Listar clientes del sistema

Campo de caso de uso	**Descripción**

Nombre del caso de uso	**CU01.4:** Listar clientes del sistema.
Área temática	Gestión de clientes.
Suceso empresarial	El equipo técnico de 720Tec necesita acceder a alguno de sus clientes para realizar alguna tarea relacionada con la documentación o para establecer una conexión remota con algún equipo. Para ello, se mostrará una lista de los clientes existentes en el sistema, de forma que el usuario pueda seleccionar uno dentro del catálogo.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la información de una forma íntegra y actualizada respecto a sus clientes.
Visión general del caso de uso	El sistema permite al personal de soporte visualizar todos los clientes existentes en el sistema, aparte también permite buscar cliente de forma individual a través de un buscador.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Para que un cliente pueda ser listado debe existir en la base de datos.
Resultado de la terminación	1. Se muestran los clientes satisfactoriamente. 2. Los clientes no se pueden mostrar porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay conexión con la base de datos. 2. No hay conexión con la base de datos.
Descripción del caso de uso	1. Un miembro del equipo accede al sistema. 2. El sistema realiza una llamada a la base de datos, obtiene un listado con todos los clientes de la empresa y los muestra en un elemento html.

	3. Aparte del listado de clientes el sistema muestra un buscador donde el usuario puede buscar de forma individualizada. 4. El usuario introduce el nombre del cliente en el buscador y si existe esconde los demás clientes de la lista a excepción del buscado.
Asociaciones de casos de uso	No directamente.
Rastreabilidad	Documentación.
Resumen de entradas	Para el listado completo sólo es necesario acceder al sistema. Para el buscador es necesario proporcionar el nombre de la empresa cliente.
Resumen de salidas	Lista de nombre de clientes.
Índice de utilización (1/10)	● **Satisfacción:** 8 ● **Importancia:** 10 ● **Frecuencia:** 10
Notas del caso de uso	Habría que establecer los términos por los cuales se podría realizar la búsqueda. Incluso se podría categorizar los clientes y mostrar grupos de clientes por categorías.

CU02: Gestionar elementos de red

Este caso de uso comprende las acciones por parte del equipo de soporte sobre la gestión de los componentes de red de las infraestructuras informáticas de sus clientes. De esta forma, el sistema debe permitir a los usuarios crear, modificar y eliminar estos elementos. Aparte debe ser capaz de representarlos de una forma visual como elementos de un diagrama y permitir que los usuarios puedan interactuar con ellos a través de la interfaz gráfica. Este caso de uso también incluye las acciones que el sistema permite realizar sobre los elementos, es decir, consultar entradas de documentación o notas y establecer conexiones con el dispositivo de red seleccionado.

CU02.1: Añadir elemento de red al sistema

Campo de caso de uso	Descripción
Nombre del caso de uso	**CU02.1:** Añadir elemento de red al sistema.
Área temática	Gestión de elementos de red.
Suceso empresarial	Un nuevo cliente contrata los servicios de 720Tec por lo que el equipo de soporte necesita añadir todos los elementos de la infraestructura informática en el sistema. Otro suceso podría ser que un cliente ya existente solicite la inclusión de nuevos equipos en su sistema, por lo que hay que documentarlos correctamente.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite al usuario crear un nuevo elemento de red y representarlo de forma gráfica.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Se debe seleccionar el cliente al cual añadir el elemento de red.
Resultado de la terminación	1. El elemento se ha añadido satisfactoriamente. 2. No se ha podido añadir el elemento porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay conexión con la base de datos. 2. No hay conexión con la base de datos.
Descripción del	1. Uno de los miembros del equipo de soporte selecciona un

caso de uso	cliente. 2. El sistema muestra una barra de herramientas con los diferentes elementos que pueden ser arrastrados al canvas del diagrama. 3. El sistema genera el nuevo elemento, con todos los campos vacíos a excepción del identificador del elemento y lo renderiza en el diagrama, el elemento se asigna a la capa de visualización por defecto.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Ninguno, el número identificativo del elemento se genera automáticamente. Los campos quedan vacíos para que puedan ser modificados posteriormente por los usuarios.
Resumen de salidas	Visualización del diagrama con el nuevo elemento incluído.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 5
Notas del caso de uso	La idea es que el usuario pueda crear un objeto genérico vacío para que pueda ser perfilado más adelante.

CU02.2: Modificar elemento de red al sistema

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU02.2:**Modificar elemento de red al sistema.
Área temática	Gestión de elementos de red.

Suceso empresarial	El equipo de soporte necesita documentar las características de los elementos de la infraestructura informática de un cliente por los sucesos descritos en el caso de uso **"CU02.1:** Añadir elemento de red al sistema".
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite a los usuarios modificar la información asociado los elementos de red.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Debe existir el elemento que se quiere modificar.
Resultado de la terminación	1. El elemento se ha modificado satisfactoriamente. 2. No se ha podido modificar el elemento porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay conexión con la base de datos. 2. No hay conexión con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un elemento y elige una opción "Modificar elemento". 2. El sistema abre una nueva ventana con un formulario que muestra los datos existentes del elemento y que permite modificarlos. 3. El usuario introduce y/o modifica los datos y pulsa un botón de verificación para guardar los cambios. 4. El sistema realiza una validación de los datos introducidos por el usuario. a. Si los datos son válidos se continúa con el siguiente paso .

	b. Si los datos no son válidos se muestra un mensaje de error. 5. El sistema realiza la llamada al servidor para almacenar los cambios realizados y muestra un mensaje conforme los cambios se han podido realizar con éxito.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Información relacionada a un dispositivo de red, tal como puede ser la dirección IP, el nombre del dispositivo, tipo de dispositivo, en que capas se puede visualizar el elementos etc.
Resumen de salidas	Visualización del diagrama con el nuevo elemento incluído.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 6
Notas del caso de uso	El usuario podría acceder a dicha funcionalidad a través de un menú mostrado por pulsar el botón derecho del ratón sobre el elemento.

CU02.3: Eliminar elemento de red del sistema

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU02.3:** Eliminar elemento de red del sistema.

Área temática	Gestión de elementos de red.
Suceso empresarial	Un cliente decide prescindir de los servicios contratados a 720Tec por lo que es necesario eliminar la información referente a su infraestructura informática . Otro suceso podría deberse a un cambio en la infraestructura de un cliente existente que requiera la retirada de ciertos equipos informáticos.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite a los usuarios eliminar elementos de red asociados a un cliente.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Debe existir el elemento que se quiere eliminar.
Resultado de la terminación	1. El elemento se ha eliminado satisfactoriamente. 2. No se ha podido eliminar el elemento porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay conexión con la base de datos. 2. No hay conexión con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un elemento y elige una opción “Eliminar elemento”. 2. El sistema comprueba todas las conexiones existentes del elemento a eliminar. 3. El sistema realiza las llamadas para eliminar todas conexiones del elemento dirigidas a otro elemento de la red. 4. El sistema realiza las llamadas para eliminar el elemento y su representación gráfica en el diagrama.

	5. El sistema muestra el estado de la red después de realizar las operaciones de borrado.
Asociaciones de casos de uso	● **CU01.3:** Eliminar cliente. ● **CU03.3:** Eliminar conexión.
Rastreabilidad	Documentación.
Resumen de entradas	Por parte del usuario ninguna, solo se necesita los identificadores del elemento y sus conexiones para realizar las llamadas al servidor con el fin de poder borrar la información.
Resumen de salidas	Visualización del diagrama con el elemento y sus conexiones borradas.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 4
Notas del caso de uso	El usuario podría acceder a dicha funcionalidad a través de un menú mostrado por pulsar el botón derecho del ratón sobre el elemento. Si otro elemento de la red hace referencia al elemento borrado, por ejemplo si se elimina el DNS de una zona que utilizan varios equipos para la resolución de nombres, habría que automatizar el proceso de desasociación con el fin de facilitar las tareas de los usuarios.

CU02.4: Consultar información de elemento

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU02.4:** Consultar información de elemento.
Área temática	Gestión de elementos de red.

Suceso empresarial	Un miembro del equipo de soporte necesita acceder a la información de algún dispositivo de red de un cliente para poder realizar alguna de sus tareas.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el usuario que necesita obtener los datos de un elemento de red.
Visión general del caso de uso	El sistema permite a los usuarios consultar información referente a un elemento de la red.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Debe existir el elemento que se quiere consultar.
Resultado de la terminación	1. Se muestra la información correctamente. 2. No se ha podido mostrar la información porque el sistema no ha podido obtenerla de la base de datos.
Condición que puede afectar al resultado de terminación	1. Se ha podido cargar la información de la base de datos. 2. No se ha podido cargar la información de la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un elemento y elige una opción “Ver detalles”. 2. El sistema muestra una nueva ventana con todos los detalles del elemento seleccionado.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Por parte del usuario ninguna, solo se necesita los identificadores del elemento para obtener la información referente del elemento.
Resumen de salidas	Visualización, en una ventana, de todos la información referente al elemento, así como una imagen que permita al usuario contextualizar la situación del elemento.

Índice de utilización (1/10)	• **Satisfacción:** 10 • **Importancia:** 7 • **Frecuencia:** 8
Notas del caso de uso	El usuario podría acceder a dicha funcionalidad a través de un menú mostrado por pulsar el botón derecho del ratón sobre el elemento. No hace falta realizar llamadas cada vez que se quiera visualizar los detalles de un elemento, al elegir un cliente puede cargarse en memoria la información de todos los elementos de la red.

CU02.5: Documentar elemento

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU02.5:** Documentar elemento.
Área temática	Gestión de elementos de red.
Suceso empresarial	Sucede un problema en algunos de los equipos de un cliente. El personal de 720Tec realiza alguna acción sobre el equipo para corregir el problema. Tras realizar dicha acción el equipo de soporte quiere registrarla para con el fin de dejar constancia histórica de todo lo realizado sobre cada uno de los equipos de la infraestructura del cliente.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener y registrar toda la información referente a las infraestructuras de los clientes.
Visión general del caso de uso	El sistema permite a los usuarios mantener un registro histórico de las acciones y sucesos que van ocurriendo en cada uno de los elementos de red.

Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Debe existir el elemento que se documentar.
Resultado de la terminación	1. Se ha añadido un nuevo registro satisfactoriamente. 2. No se ha podido agregar el nuevo registro porque no hay conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay conexión con la base de datos. 2. No hay conexión con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo pulsa en “Documentar elemento”. 2. El sistema muestra una nueva ventana con un histórico de las entradas que se han ido añadiendo a lo largo del tiempo por parte del equipo de soporte. Además se muestra un formulario para que el usuario pueda introducir una nueva entrada. 3. El usuario escribe la información y pulsa en “Añadir nueva entrada”. 4. El sistema realiza una llamada al servidor para almacenar el nuevo registro asociado al elemento. 5. El sistema vuelve a mostrar el histórico de entradas, incluye la nueva entrada introducida, o un mensaje de error en caso de que la llamada falle.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Si se desea añadir una nueva entrada de documentación para un elemento es necesario introducir los datos de la misma.

Resumen de salidas	Visualización, en una ventana, de una lista histórica de las entradas de documentación introducidas hasta el momento.
Índice de utilización (1/10)	● **Satisfacción:** 7 ● **Importancia:** 7 ● **Frecuencia:** 8
Notas del caso de uso	El usuario podría acceder a dicha funcionalidad a través de un menú mostrado por pulsar el botón derecho del ratón sobre el elemento. Se podría utilizar un estilo parecido al de MeisterTask, el cual resulta sencillo y eficaz.

CU03: Gestionar conexiones de red

Este caso de uso comprende las acciones dirigidas a relacionar los elemento de red entre sí. De esta forma, el sistema debe permitir a los usuarios crear, modificar y eliminar conexiones entre los diferentes elementos del diagrama, de forma que queden reflejadas en la base de datos. Aparte, debe ser capaz de representarlos de una forma visual como aristas del diagrama y permitir que los usuarios puedan interactuar con ellas a través de la interfaz gráfica.

CU03.1: Añadir conexión

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU03.1:** Añadir conexión.
Área temática	Gestión de conexiones.
Suceso empresarial	Un nuevo cliente contrata los servicios de 720Tec por lo que es necesario representar toda su infraestructura informática con sus correspondientes conexiones. También podría desencadenarse a causa de un aumento del número de equipos de un cliente o un

	cambio en la topología de su red.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite al usuario crear una nueva conexión entre dispositivo de una red de un cliente y las representa de forma gráfica a través de las aristas del diagrama.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Deben existir los dos o más elementos de red que se desean interconectar.
Resultado de la terminación	1. Los elementos se conectan satisfactoriamente. 2. No se han podido conectar debido a un error de validación. 3. No se han podido conectar debido a un error de conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay comunicación con la base de datos y la nueva conexión pasa la validación. 2. Hay comunicación con la base de datos pero la nueva conexión no pasa la validación del sistema. 3. No hay conexión con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un un elemento de la red y pulsa en añadir nueva conexión. 2. El sistema permite arrastrar una arista desde un elemento (vértice del diagrama) a otro. 3. Una vez el usuario indica que dos elementos conectar, el sistema realiza una validación. a. Comprueba si es posible la conexión entre los dos tipos de dispositivos. b. Comprueba que existan interfaces o puntos de conexión libres para poder establecerla. 4. Si el sistema valida la conexión realiza una llamada a la base de datos para añadir la nueva conexión entre los dos

	elementos. En caso contrario, muestra un mensaje de error. 5. El sistema muestra el diagrama renderizando la nueva conexión.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Por parte del usuario ninguna, pero para realizar la llamada se necesitan los identificadores de los dos elementos a conectar.
Resumen de salidas	Visualización del diagrama con la nueva conexión incluída.
Índice de utilización (1/10)	● **Satisfacción:** 7 ● **Importancia:** 8 ● **Frecuencia:** 7
Notas del caso de uso	Habría que mirar de diferenciar entre conexiones físicas y lógicas.

CU03.2: Modificar conexión

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU03.2:** Modificar conexión.
Área temática	Gestión de conexiones.
Suceso empresarial	Uno de los clientes de 720Tec solicitar realizar cambios en su infraestructura de red que implican conexiones diferentes entre los dispositivos.

Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite al usuario modificar los puntos de conexión de una existente en el sistema.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Debe existir la conexión que se desea modificar. 3. Deben existir los dos o más elementos de red que se desean interconectar.
Resultado de la terminación	1. Las conexión se ha modificado correctamente. 2. No se han podido modificar la conexión debido a un fallo en la validación. 3. No se han podido conectar debido a un error de conexión con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay comunicación con la base de datos y la conexión pasa la validación. 2. Hay comunicación con la base de datos pero no pasa la validación del sistema. 3. No hay conexión con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un una conexión de red (arista) y la arrastra para cambiar los puntos de conexión. 2. Una vez el usuario indica que dos elementos conectar, el sistema realiza una validación. a. Comprueba si es posible la conexión entre los dos tipos de dispositivos. b. Comprueba que existan interfaces o puntos de conexión libres para poder establecerla. 3. Si el sistema valida la conexión, realiza una llamada a la base de datos para añadir la nueva conexión entre los dos elementos y borrar la que existía anteriormente. En caso contrario, muestra un mensaje de error y no realiza ningún cambio.

	4. El sistema muestra el diagrama renderizando el cambio en la conexión. En caso de fallar, renderiza la infraestructura tal como estaba anteriormente.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Por parte del usuario ninguna, pero para realizar la llamada se necesita los identificadores de los dos elementos a conectar y la identificación de la conexión a modificar.
Resumen de salidas	Visualización del diagrama con los cambios de conexión incluidos.
Índice de utilización (1/10)	● **Satisfacción:** 7 ● **Importancia:** 6 ● **Frecuencia:** 7
Notas del caso de uso	Al realizar una modificación de una conexión si el sistema no permite llevarla a cabo habría que dejarla tal como estaba anteriormente, es decir, esa conexión no debería perderse.

CU03.3: Eliminar conexión

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU03.3:** Eliminar conexión.
Área temática	Gestión de conexiones.
Suceso empresarial	Uno de los clientes de 720Tec solicitar realizar cambios en su infraestructura de red que implican conexiones diferentes entre los dispositivos. Por otra parte, podría deberse a que un cliente

	decide prescindir de los servicios de 720Tec por lo que es necesario borrar toda la información referente a su infraestructura de red.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite al usuario eliminar conexiones existentes entre elementos del diagrama de red.
Condiciones previas	1. Debe existir conexión entre el cliente y la base de datos. 2. Debe existir la conexión que se desea eliminar. 3. Se debe estar en modo edición de diagrama.
Resultado de la terminación	1. Las conexión se elimina correctamente. 2. No se han podido eliminar la conexión debido a un fallo de comunicación con la base de datos.
Condición que puede afectar al resultado de terminación	1. Hay comunicación con la base de datos. 2. No hay comunicación con la base de datos.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un una conexión de red (arista) y pulsa en “Borrar conexión”. 2. El sistema realiza una llamada a la base de datos para borrar la conexión entre los dos elementos. En caso contrario, muestra un mensaje de error y no realiza ningún cambio. 3. El sistema muestra el diagrama renderizando la infraestructura sin la conexión eliminada. En caso de fallar, renderiza la infraestructura tal como estaba anteriormente.
Asociaciones de casos de uso	● **CU01.3:** Eliminar cliente. ● **CU02.3:** Eliminar elemento.
Rastreabilidad	Documentación.
Resumen de	Por parte del usuario ninguna, pero para realizar la llamada se

entradas	necesita la identificación de la conexión.
Resumen de salidas	Visualización del diagrama sin la conexión eliminada.
Índice de utilización (1/10)	● **Satisfacción:** 7 ● **Importancia:** 6 ● **Frecuencia:** 6
Notas del caso de uso	En caso de almacenar información adicional de la conexión también habría que eliminarla.

CU04: Gestión de visualización de red

Este caso de uso comprende las acciones del sistema relacionadas con la visualización de la infraestructura informática mediante diagramas. El sistema tiene que representar visualmente todos los elementos de red, existentes en la base de datos, a través de un diagrama y permitir a los usuarios interactuar con los elementos gráficos y filtrar la visualización siguiendo un sistema de capas. Los cambios visuales realizados en el diagrama deben almacenarse en el servidor y poderse recuperar posteriormente. Es importante que los diagramas no guarden información sensible sobre los clientes, por lo que se gestionan como si se trataran de plantillas que son rellenadas por el sistema.

CU04.1: Modificar diagrama

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU04.1:** Modificar diagrama.
Área temática	Gestión de visualización de red.
Suceso empresarial	El equipo de soporte necesita construir el diagrama de un cliente para tener correctamente documentada y poseer comprensión sobre su infraestructura informática.

Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite al usuario modificar los elementos gráficos del diagrama para adaptarlo a sus necesidades sin que la información sobre los elementos y conexiones se vea alterada.
Condiciones previas	1. Debe existir conexión con la base de datos. 2. Debe existir un diagrama sobre el cual el usuario pueda trabajar.
Resultado de la terminación	1. Los cambios gráficos se hacen visibles.
Condición que puede afectar al resultado de terminación	1. El cliente funciona o no correctamente.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un un elemento del diagrama y lo arrastra. 2. El sistema permite cambiar de posición el elemento sin que esto afecta a la integridad de los datos. 3. Los cambios se hacen visibles para el usuario.
Asociaciones de casos de uso	
Rastreabilidad	Documentación.
Resumen de entradas	Posiciones y valores que necesite la biblioteca de grafos para dibujar correctamente los cambios.
Resumen de salidas	Visualización del diagrama con las modificaciones realizadas.
Índice de utilización (1/10)	● **Satisfacción:** 8 ● **Importancia:** 7 ● **Frecuencia:** 9

Notas del caso de uso	Los cambios se ejecutan en el cliente y cada periodo de tiempo son enviados al servidor.

CU04.2: Almacenar diagrama

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU04.2:** Almacenar diagrama.
Área temática	Gestión de visualización de red.
Suceso empresarial	Después o mientras el equipo de 720Tec esté realizando cambios en un diagrama, desea guardar los cambios realizados para que puedan ser recuperados posteriormente.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el encargado de mantener la documentación de las infraestructuras de sus clientes.
Visión general del caso de uso	El sistema permite al usuario almacenar los diagramas en el servidor para que puedan ser recuperados posteriormente. Los diagramas se almacenan en forma de plantilla para evitar el robo de datos sensibles. El diagrama se transmite codificado en XML o JSON para simplificar su tratamiento.
Condiciones previas	1. Debe existir conexión con la base de datos. 2. Debe existir un diagrama que almacenar.
Resultado de la terminación	1. El diagrama se almacena en la base datos. 2. El diagrama no se ha podido almacenar, por fallo de conexión.
Condición que puede afectar al	1. Hay conexión con la base de datos. 2. No hay conexión con la base de datos.

resultado de terminación	
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte, después de realizar cambios en el diagrama, pulsa en “Guardar cambios”. 2. El sistema realiza las llamadas necesarias para guardar información sobre los elementos de red en el servidor. 3. El sistema realiza las llamadas necesarias para guardar la información complementaria de los elementos y conexiones en el servidor. 4. El sistema codifica el diagrama en JSON o XML y lo transmite al servidor. El servidor lo almacena de forma que queda asociado al cliente y a la capa de visualización.
Asociaciones de casos de uso	● **CU04.1:** Modificar diagrama diagrama.
Rastreabilidad	Documentación.
Resumen de entradas	Diagrama codificado en XML o JSON.
Resumen de salidas	Mensaje de feedback para el usuario.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 10
Notas del caso de uso	El diagrama debe enviarse al servidor en forma de plantilla para que los datos puedan ser rellenados al recuperarse. Lo que sí que deberá contener es la identificación de cada elemento para que queden relacionados con la información contenida en las bases de datos.

CU04.3: Recuperar diagrama

Campo de caso de uso	Descripción
Nombre del caso de uso	**CU04.3:** Recuperar diagrama.
Área temática	Gestión de visualización de red.
Suceso empresarial	Un usuario del sistema desea consultar información sobre la estructura informática de un cliente, esta información tiene que ser recuperada y presentada para que el usuario pueda visualizarla.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el interesado en visualizar la información de una infraestructura para el desempeño de sus tareas.
Visión general del caso de uso	El sistema se encarga de recuperar la información asociada a un diagrama para renderizarlo en el estado que lo almacenó el usuario.
Condiciones previas	1. Debe existir conexión con la base de datos. 2. Debe existir un diagrama almacenado. 3. Deben existir todos los elementos asociados al diagrama.
Resultado de la terminación	1. El diagrama se visualiza correctamente. 2. El diagrama no se puede visualizar por fallo de conexión. 3. El diagrama no se puede visualizar correctamente por inconsistencia de los datos.
Condición que puede afectar al resultado de	1. Hay conexión con la base de datos y los datos son consistentes. 2. No hay conexión con la base de datos.

terminación	3. Hay conexión con la base de datos, pero las asociaciones de los elementos del diagrama con los datos del servidor no son consistentes.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona un cliente para visualizar su diagrama de red. 2. El sistema realiza una llamada al servidor para recuperar la plantilla del diagrama en formato JSON o XML. 3. El sistema realiza llamadas al servidor para obtener la información asociada a cada uno de los elementos del diagrama, haciendo uso de los identificadores y rellena la plantilla con los datos obtenidos. a. Si existen inconsistencias, identificadores de elementos que no existen, el sistema muestra un mensaje de error y elimina ese elemento de la plantilla. 4. Una vez la plantilla está rellenada con los datos, el sistema la renderiza y la muestra por pantalla.
Asociaciones de casos de uso	● **CU04.4:**Visualizar capa.
Rastreabilidad	Documentación.
Resumen de entradas	Identificador del cliente del cual se quiere recuperar el diagrama.
Resumen de salidas	Visualización del diagrama.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 10
Notas del caso de uso	Si existen inconsistencia se puede borrar el elemento del diagrama de forma que no se vuelva a renderizar, sin corromper la integridad del resto del diagrama.

CU04.4: Visualizar capa

Campo de caso de uso	Descripción
Nombre del caso de uso	**CU04.4:** Visualizar capa.
Área temática	Gestión de visualización de red.
Suceso empresarial	Un usuario del sistema desea consultar información sobre la estructura informática de un cliente, pero no necesita mostrar toda la información, solamente la que está relacionada con algún tipo de funcionalidad.
Actores	El actor que inicia el caso de uso es el equipo de soporte, pues es el interesado en visualizar la información de una infraestructura para el desempeño de sus tareas.
Visión general del caso de uso	El sistema se encarga de presentar la información del diagrama según las necesidades del usuario. Para ello, se utiliza un sistema de capas que actúan como un filtro a la hora de renderizar cada elemento del diagrama.
Condiciones previas	1. Debe existir conexión con la base de datos. 2. Debe existir un diagrama almacenado. 3. Deben existir todos los elementos asociados al diagrama.
Resultado de la terminación	1. El diagrama se visualiza correctamente. 2. El diagrama no se puede visualizar por fallo de conexión. 3. El diagrama no se puede visualizar correctamente por inconsistencia de los datos.
Condición que puede afectar al	1. Hay conexión con la base de datos y los datos son consistentes.

resultado de terminación	2. No hay conexión con la base de datos. 3. Hay conexión pero las asociaciones de los elementos del diagrama con los datos del servidor no son consistentes.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona una capa de visualización. 2. El sistema realiza una llamada al servidor para comprobar la existencia de un diagrama almacenado para la capa seleccionada. 3. En caso de existir, el sistema recupera el diagrama, realiza las llamadas necesarias, rellena la plantilla y la renderiza. 4. En caso de no existir, el sistema utiliza los enlaces de la capa por defecto y renderiza automáticamente todos los elementos asociados a dicha capa.
Asociaciones de casos de uso	● **CU04.3:** Recuperar diagrama.
Rastreabilidad	Documentación.
Resumen de entradas	Identificador del cliente y capa asociados al diagrama que se desea recuperar.
Resumen de salidas	Visualización del diagrama de la capa indicada.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 8
Notas del caso de uso	Sería interesante permitir que el usuario pudiera crear sus propias capas. Una vez se renderiza un diagrama de forma automática para una capa, el usuario puede almacenar los cambios que realice en el diagrama.

CU05: Conectar con la red cliente

Este caso de uso consiste en las acciones que debe realizar el sistema para permitir al usuario conectarse con la red de un cliente y poder establecer conexiones con algunos dispositivos de red.

CU05.1: Montar VPN

Campo de caso de uso	Descripción
Nombre del caso de uso	**CU05.1:** Montar VPN.
Área temática	Conectar con la red cliente.
Suceso empresarial	El equipo de soporte necesita conectarse a algún dispositivo de la red del cliente con el fin de llevar a cabo algún tipo de configuración o comprobación de estado.
Actores	El actor que inicia el caso de uso es el equipo de soporte, encargado de comprobar el estado de las redes en busca de errores y mantener una correcta configuración del sistema.
Visión general del caso de uso	El sistema permite al usuario establecer una conexión virtual, punto a punto, entre la red de la empresa y la red del cliente.
Condiciones previas	1. Debe existir conexión con la base de datos. 2. Debe existir el cliente en el sistema. 3. Debe existir el elemento, en el diagrama, que sirva como punto de conexión. 4. Deben existir las condiciones correctas para establecer la VPN. a. La máquina del usuario tiene que ser capaz de alcanzar la máquina del cliente a través de internet. b. La máquina servidora del cliente debe ofrecer el servicio VPN. c. La máquina del usuario debe tener instalado el cliente VPN y el perfil necesario para establecer la conexión. d. El servidor debe poseer las credenciales del cliente para poder establecer la conexión.

Resultado de la terminación	1. Se establece la conexión VPN. 2. No se establece la conexión VPN.
Condición que puede afectar al resultado de terminación	1. Hay algún error en los perfiles. 2. Hay algún error en las credenciales. 3. No se puede comunicar con el servidor. 4. No hay acceso a internet. 5. No hay un cliente VPN en la máquina local. 6. La máquina no ofrece el servicio VPN o no es accesible.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona el elemento correspondiente y pulsa en “Establecer VPN”. 2. El sistema busca el perfil VPN del cliente. a. El sistema muestra un mensaje de error si no se encuentra el perfil. 3. El sistema realiza una llamada segura al servidor con el fin de obtener las credenciales del cliente. a. El sistema muestra un mensaje de error si no se pueden obtener las credenciales. 4. El sistema ejecuta el cliente VPN, en la máquina local, con la configuración del perfil y las credenciales obtenidas. a. El sistema muestra un mensaje de error si no se puede ejecutar el cliente VPN. 5. Se establece la conexión punto a punto.
Asociaciones de casos de uso	● **CU05.2:** Establecer conexión remota.
Rastreabilidad	Documentación.
Resumen de entradas	Identificador del cliente y máquina servidora.
Resumen de salidas	Cliente VPN con la conexión establecida.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 10

Notas del caso de uso	En caso de no poder implementar la funcionalidad de forma que la conexión VPN se establezca automáticamente, habría que implementarla de forma que la información de las credenciales se redirija hacia el cliente VPN, a través del portapapeles, y ocultando la información sensible.

CU05.2: Establecer conexión remota

Campo de caso de uso	**Descripción**
Nombre del caso de uso	**CU05.2:** Establecer conexión remota.
Área temática	Conectar con la red cliente.
Suceso empresarial	El equipo de soporte necesita conectarse a algún dispositivo de la red del cliente con el fin de llevar a cabo algún tipo de configuración o comprobación de estado.
Actores	El actor que inicia el caso de uso es el equipo de soporte, encargado de comprobar el estado de las redes en busca de errores y mantener una correcta configuración del sistema.
Visión general del caso de uso	El sistema permite al usuario establecer una conexión remota con algún dispositivo de la red del cliente.
Condiciones previas	1. Debe existir una VPN establecida entre la red de la empresa y la del cliente, esto incluye las condiciones descritas en el caso de uso: **CU05.1:** Montar VPN. 2. La máquina sobre la cual se quiere establecer conexión remota debe ofrecer el servicio: ssh, escritorio remoto, etc. 3. La máquina local debe tener instalado el cliente que permita establecer la conexión remota.
Resultado de la	1. Se establece la conexión remota.

terminación	2. No se establece la conexión remota.
Condición que puede afectar al resultado de terminación	1. No se ha establecido la conexión VPN. 2. No existen las credenciales del cliente. 3. El cliente para establecer la conexión remota no existe en la máquina local.
Descripción del caso de uso	1. Uno de los miembros del equipo de soporte selecciona el elemento correspondiente y pulsa en “Establecer conexión remota”. 2. El sistema analiza el tipo de dispositivo y ofrece posibilidades según el mismo: ssh para una máquina Linux, escritorio remoto para Windows, etc. 3. El sistema realiza una llamada segura al servidor con el fin de obtener las credenciales del cliente. a. El sistema muestra un mensaje de error si no se pueden obtener las credenciales. 4. El sistema ejecuta el cliente de conexión remota, en la máquina local, con las credenciales obtenidas. a. El sistema muestra un mensaje de error si no se puede ejecutar el cliente de conexión remota. 5. Se establece la conexión y el usuario obtiene el control de la máquina.
Asociaciones de casos de uso	● **CU05.1:**Montar VPN.
Rastreabilidad	Documentación.
Resumen de entradas	Identificador del cliente y máquina servidora.
Resumen de salidas	Conexión remota establecida.
Índice de utilización (1/10)	● **Satisfacción:** 10 ● **Importancia:** 10 ● **Frecuencia:** 10
Notas del caso de	En el momento de establecer la conexión remota, el sistema

uso	podría comprobar si es capaz de comunicarse con la máquina, en caso contrario, ya no ofrecería la posibilidad de establecer la conexión.