ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DEL SOFTWARE

Despliegue de infraestructura y servicios de red en la Residencia Universitaria “Alberto Jiménez Fraud”: Despliegue de infraestructura, implantación de telefonía IP y desarrollo del backend de la aplicación de gestión “Panel del residente”

Network infrastructure and services deployment at the Málaga University’s Dormitory “Alberto Jiménez Fraud”: Network infrastructure deployment, IP telephony implementation and backend development of the management application “Panel del residente”

Realizado por
Antonio Ángel Cruzado Castillo
Tutorizado por
Pedro Merino Gomez
Almudena Díaz Zayas
Victoriano Francisco Giralt García
Departamento
Tecnología electrónica

UNIVERSIDAD DE MÁLAGA
MÁLAGA, Septiembre 2018

Fecha defensa:
El Secretario del Tribunal

Resumen: El objetivo de este trabajo es el de describir en detalle el proceso que se ha seguido para renovar por completo la infraestructura y servicios de red de la residencia universitaria de la Universidad de Málaga, debido a las distintas carencias que esta presentaba en esta materia. En concreto se ha renovado la infraestructura de red cableada e inalámbrica y se ha desarrollado y puesto en producción un nuevo servicio de telefonía y un software de gestión en forma de aplicación web que permite facilitar las relaciones administrativas entre los residentes y el personal de la residencia. Para ello se ha llevado a cabo en primer lugar un despliegue de red a modo de actividad formativa involucrando a distintos miembros de la comunidad universitaria, principalmente estudiantes de distintos grados. En segundo lugar se ha procedido al desarrollo del software web de gestión empleando tecnologías web recientes, mientras en paralelo se ha estado diseñando e implantando un nuevo servicio de telefonía por internet, lo cual ha implicado la configuración y puesta en marcha de una nueva centralita telefónica. Todo esto ha permitido que la residencia universitaria posea una infraestructura de red de calidad la cual es usada a diario por alrededor de 260 residentes. Del mismo modo el nuevo software de gestión facilita la vida de los residentes permitiendo a estos evitar gestiones presenciales con la administración de la residencia. Por otro lado el nuevo servicio de telefonía garantiza la calidad de las telecomunicaciones por voz de los residentes con el exterior del edificio. Todo esto ha sido posible gracias a la robusta red que ha sido desplegada.

Palabras claves: despliegue de red, software de gestión, residencia universitaria, internet, formación, involucración de estudiantes, comunidad universitaria, universidad de Málaga, telefonía por internet, renovación infraestructura, servicios de red, administración electrónica, centralita telefónica, administración de sistemas

Abstract: The main goal of this work is describing in detail the followed procedure to completely upgrade the network infraestructure and services of the University of Malaga's dorm due to its lack of reliable ones. The wired and wireless network infrastructure has been fully renewed, and both a new telephony system and a management web software have been designed, developed and deployed. The managament web software implements a customer area for the residents to perform administrative tasks though the internet related to their daily life in the dorm. To achieve both the wired and wireless infrastructure renewal, a lot of volunteers (most of them students) coming from the university have taken part in the deployment. Right after finishing the network deployment, both the management web software development and the new telephony system deployment (being necessary to configure and deploy a new private branch exchange) started. All of these actions have improved the everyday communications quality of around all the 260 residents living in the dorm. The new telephony system allows the users to be better in touch with the outside world, allowing them to enjoy high quality voice calls, while the new management software simplifies and helps them solve their everyday dorm related administrative tasks and issues. All of this is possible thanks to the strong, reliable deployed network infrastructure.

Keywords: network deployment, management software, student dorm, internet, learning by doing, student involvement, university community,

Málaga University, voice over ip, infrastructure upgrade, network services, electronic administration, system administration, private branch exchange

1. Introducción

En estos últimos años las TIC han tenido un enorme desarrollo y todo esto ha provocado que su inclusión en la sociedad haya llegado a unos niveles nunca antes vistos. Consecuencia de todo esto es que se puede decir que nos encontramos en la denominada Sociedad de la información, en la que las TIC representan una parte importante de nuestras vidas y ejercen por tanto una gran influencia en la forma en la que llevamos esta.

La mayor parte de la población (sobre todo la más joven) disfruta de acceso a Internet en sus hogares y de media cada individuo posee 3 aparatos con posibilidad de conexión a la red (televisores, consolas, smartphones, portátiles…). En todo este contexto la movilidad adquiere además un indiscutible protagonismo: la posesión de dispositivos portátiles es cada vez más elevada y allá donde estemos fuera de casa también queremos permanecer conectados.

De todo esto se infiere que, entre otros lugares, una residencia de estudiantes constituye sin lugar a dudas un lugar que debe estar a la altura para satisfacer la demanda de TICs que se tiene en estos días, y se tendrá en un futuro.

El presente documento describe cómo se ha llevado a cabo un despliegue completo de infraestructura y servicios de red en la Residencia de estudiantes Alberto Jiménez Fraud de la Universidad de Málaga, en adelante la residencia (si bien es extensible a otros entornos con características y demandas similares) para adecuar estos a las demandas actuales y venideras.

La residencia alberga unos 250 miembros de la UMA (entre los que hay alumnado y profesorado de cualquier procedencia) y posee una instalación de red que les da servicio la cual data del año 2000. A lo largo de los años la instalación ha proveído exitosamente a los usuarios de conexión por cable, pero no ha ocurrido igual con la provisión de conexión inalámbrica.

Como se ha mencionado antes, hoy día la movilidad juega un papel importante en la forma en la que nos conectamos, y si además se traslada todo esto a un entorno universitario la importancia de este factor no es en absoluto despreciable. Un caso típico que justifica esta situación es la de un alumno que posee su portátil y lo usa

en la residencia, biblioteca, clase, zonas de exteriores…

Es por ello que la provisión de una conexión inalámbrica completamente funcional que satisfaga las necesidades del residente es un asunto de alta prioridad. Dado que la residencia es un lugar en el que se vive además de estudiar, entre las necesidades del residente consideramos todas las posibles además de las relacionadas con estrictamente el estudio y trabajo.

1.2. Objetivos del trabajo

En base a lo detallado anteriormente los objetivos del trabajo se centran en dotar a la residencia de una infraestructura de red y servicios que satisfagan todas las necesidades TIC tanto de las personas allí alojadas como de los trabajadores (administración, mantenimiento…).

Es por ello que todos los esfuerzos del presente trabajo se han enfocado principalmente en los siguientes puntos:

- Diseñar, desplegar y mantener una infraestructura de red con la capacidad suficiente para dar cabida a los distintos servicios que se ofrecerán al personal (conexión a internet, telefonía IP y servicios como aplicaciones en red local y expuestas a internet).
- Diseñar, desplegar y mantener un sistema de telefonía IP con capacidad para llamadas locales, externas, buzón de voz y llamada en espera principalmente.
- Diseñar, desplegar y mantener una aplicación (servicio) denominada “Panel del residente” que facilita las relaciones y gestiones administrativas entre los residentes y el personal de la residencia. Desde esta aplicación se podrán realizar acciones como consultar el historial de llamadas del apartamento, ver el consumo eléctrico de este en tiempo real, mandar partes de avería informática y otras muchas acciones las cuales podrán ir añadiéndose a lo largo de la vida útil de la aplicación debido al diseño modular que esta posee,
- Consolidar una infraestructura de virtualización que facilite la puesta en marcha y mantenimiento de todos los servicios disponibles en el edificio, así como aumentar la tolerancia a fallos de estos.

Si bien en el título de este documento se hace referencia únicamente a “Despliegue”, en estos puntos se ha añadido además “diseño y mantenimiento”, puesto que en el caso que nos ocupa son dos tareas previas de las que también ha habido que ocuparse para tener un exitoso despliegue.

2. Estado del arte

En esta sección se comentan los distintos avances tecnológicos disponibles y aceptados para su uso (se encuentran consolidados y disponibles para su puesta en producción) que estén relacionados con los asuntos que se tratan en este documento.

2.1. Estado del arte en tecnologías de red cableada, inalámbricas y telefonía sobre IP

En lo referente a tecnologías de red cableada y en lo que incumbe a este trabajo se puede hacer una separación entre tecnologías y/o estándares de transmisión en el medio físico y protocolos.

Para la transmisión en el medio físico se encuentran principalmente tres tipos [1] de cableado estructurado:

- **Par trenzado**: se encuentra organizado en categorías desde la 1 hasta la 7 (a más categoría más ancho de banda en esencia) y dependiendo de la configuración de su recubrimiento los hay UTP (sin blindado), FTP (con lamina protectora alrededor de los pares) y SFTP (con blindado y lámina protectora) [1]. La interfaz física más usada con este cableado es RJ-45. Hoy día el tipo de cable más popular en un entorno empresarial es el CAT6 UTP, el cual alcanza velocidad gigabit y además el ser UTP no requiere de sistema puesta a tierra [2] (una tarea muy compleja en una instalación de red).
- **Fibra óptica**: la hay monomodo y multimodo [1].
- **Cable coaxial**.

En lo referido al cableado estructurado el estándar más empleado en redes locales es además el 802.3 [1] (basado en ethernet) que define la señalización en el nivel físico y el formato y los flujos de intercambio de las tramas de datos en el nivel de enlace del modelo OSI, además de las características del cableado [4].

El estándar 802.3 ha derivado además en otros estándares que contribuyen a la mejora de velocidad de transmisión e incluso a la inclusión de otras características

interesantes como es PoE (alimentación a través de internet, en cableado estructurado conductor de electricidad). Es el caso del estándar 802.3af, que detalla el cómo inyectar energía eléctrica en una infraestructura LAN [5].

En lo referente a tecnologías de red inalámbrica se puede encontrar una gran variedad especificadas en los distintos estándares 802.11 [6], aunque para el caso que nos ocupa en este proyecto nos centraremos en aquellas relacionadas con WiFi.

Existen diversos tipos de WiFi [7], estando cada uno de ellos basado en un estándar 802.11 diferente. Principalmente son:

- WiFi funcionando en la banda de los 2,4GHz: se encuentran WiFi b/g/n, cada uno de ellos basado en los estándares 802.11 b/g/n, respectivamente.
- WiFi funcionando en los 5GHz: se encuentra el WiFi AC, basado en el estándar 802.11 ac.

De todos los nombrados, a día de hoy los más empleados por el considerable ancho de banda que proporcionan son 802.11n ofreciendo unos 300Mbps y 802.11ac ofreciendo hasta 433Mbps y empleando múltiples antenas hasta 1.3Gbps [7].

El WiFi AC está siendo cada vez más desplegado debido al alto ancho de banda que proporciona y a que no presenta interferencias con otras tecnologías como Bluetooth y microondas [8].

Por último y para finalizar, se van a tratar una serie de tecnologías y protocolos de relevancia empleados hoy en día para lograr seguridad tanto en redes inalámbricas como cableadas.

El estándar de relevancia en este ámbito es el 802.1X, el cual define la forma en la que se encapsulan los mensajes EAP en la capa de enlace [9] del modelo OSI.

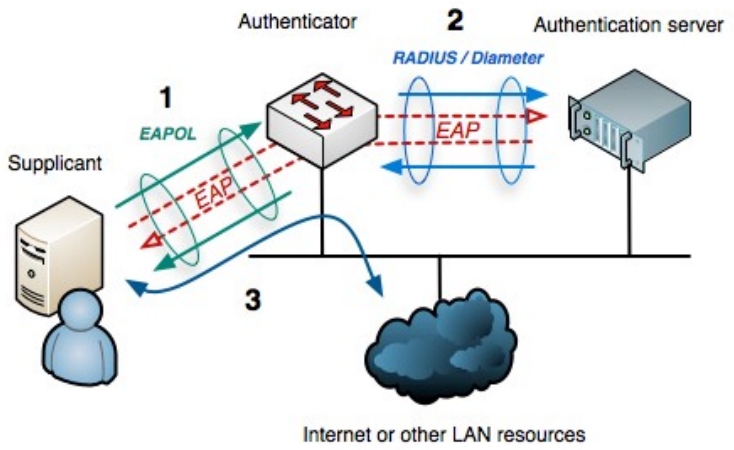


Figura 1: Ejemplo común de infraestructura en la que se emplea EAPOL

EAP es un framework de autenticación el cual cuenta con distintos métodos de autenticación, entre ellos el popular EAP-TTLS, el cual es capaz de encapsular mensajes de protocolos de autenticación antiguos como puede ser PAP en un túnel seguro que emplea seguridad en capa de transporte [10].

Tras esto entra en juego el protocolo RADIUS, el cual ofrece distintas formas de autenticar y es el que en última instancia permite el acceso o no según el éxito de la autenticación [11], con la información que el Supplicant manda al Authenticator.

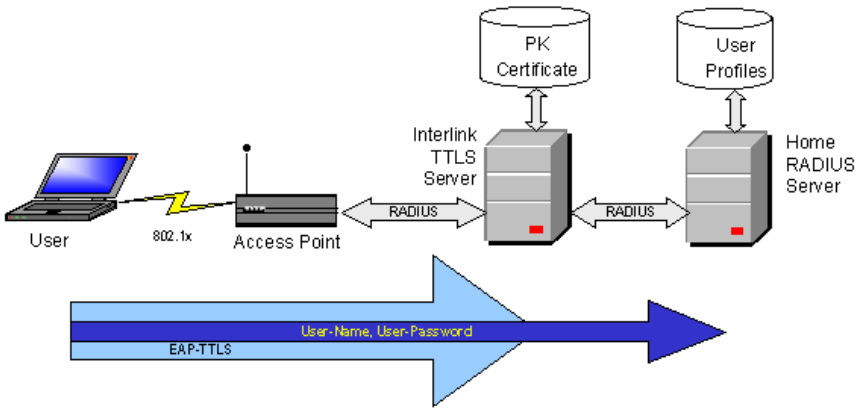


Figura 2: Ejemplo de despliegue típico empleando EAP-TTLS con PAP y RADIUS

Hoy en día el comentado (RADIUS + EAP-TTLS con PAP) es el despliegue de seguridad en términos de autenticación para redes inalámbricas (y cableadas) más empleado, siendo usado en el famoso despliegue EDUROAM, de forma que EAP es encapsulado según 802.1X [13] (EAPOL, EAP over LAN) y empleando el método de

autenticación EAP-TTLS, el cual encapsula un protocolo de autenticación no seguro por sí solo (pero sí encapsulado) como puede ser PAP [10], el cual el miembro de la infraestructura Authenticator pasa a RADIUS en una petición en la que incluye dicha información en la parte del paquete RADIUS AVPs [11] (attribute value pairs).

Por supuesto una implementación de RADIUS también puede conceder acceso o no en base a métodos de autenticación propios de EAP, de forma que EAP se encuentra encapsulado en RADIUS [11] en la parte del paquete AVPs.

En lo referente a telefonía IP, asunto que también se trata en este proyecto, cabe mencionar que a día de hoy su uso está cada vez más extendido. El que hayan desplegadas infraestructuras de red fiables y de alta capacidad hace que se haya llevado a cabo el planteamiento de transmitir la voz por ellas y así obviar la antigua línea telefónica (ya sea en analógica o como RDSI).

A día de hoy cualquier despliegue de telefonía decente en una organización se hace empleando VoIP, y considerando como parte de la infraestructura a una centralita telefónica (PBX) la cual gestiona la lógica de llamadas e incluso ofrece comunicación con el mundo exterior, ya sea mediante VoIP (mediante el uso de un SIP Trunk) o haciendo una “traducción” a la telefonía convencional (normalmente un primario de telefonía).

Es por todo esto que hoy día muchas llamadas de telefonía son completamente IP, teniendo otras que ser “traducidas” y pasar por el circuito “convencional” de telefonía.

En lo referente a tecnologías y herramientas más populares empleadas dentro de VoIP cabe destacar [14]:

- **SIP**: El protocolo más empleado a día de hoy [14] el cual funciona en conjunción con otros protocolos como RTP, ofreciendo todos estos además la opción de contar con seguridad en la capa de transporte [15].
- **Asterisk**: se trata de un PBX modular el cual trabaja con varios protocolos de señalización (no solo SIP, basta con tener un driver de canal que traduzca una señalización específica a la señalización abstracta de Asterisk) y que incorpora interfaces (CGI, REST…) para ejecutar otro software durante las llamadas. Es uno de lo más famosos y ampliamente desplegados por ser de código abierto, maduro y contar con hardware certificado (tarjetas para

Primario...etc.) [16].

2.2. Estado del arte en sistemas de virtualización

Si bien la virtualización ha llevado existiendo mucho tiempo, hoy día ha cobrado una gran popularidad con la aparición de tecnologías y herramientas que facilitan y agilizan esta.

A lo largo de este proyecto la virtualización está presente en todas las fases, desde la de desarrollo hasta la de despliegue, no cabe duda de que su uso ha simplificado de forma sustancial el trabajo.

A día de hoy se encuentran principalmente dos formas de virtualización [17]:

- **Virtualización a nivel de sistema operativo o containerización**: se trata de un tipo de virtualización en la que no se virtualiza hardware ninguno, y en la que además el kernel empleado para la ejecución del software a ser virtualizado es el del sistema operativo anfitrión [17]. De forma que al final este tipo de virtualización llega a ser muy ligera, proporcionando por ende un gran rendimiento, pudiendo verse como un proceso que se ejecuta en un entorno controlado en el sistema operativo anfitrión el cual contiene todas las dependencias que requiere para funcionar y además tiene acceso limitado a los recursos del sistema.
- **Virtualización empleando hipervisor:** es el tipo de virtualización la cual lleva mucho años en uso intensivo (y continúa siendo usada evidentemente) hasta que apareció la virtualización empleando contenedores. En este tipo de virtualización el papel principal lo juega el hipervisor (un software que se ejecuta en el sistema operativo o bien baremetal) el cual se encarga de las gestión y virtualización de recursos hardware para que sean empleados por las aplicaciones huéspedes [17].

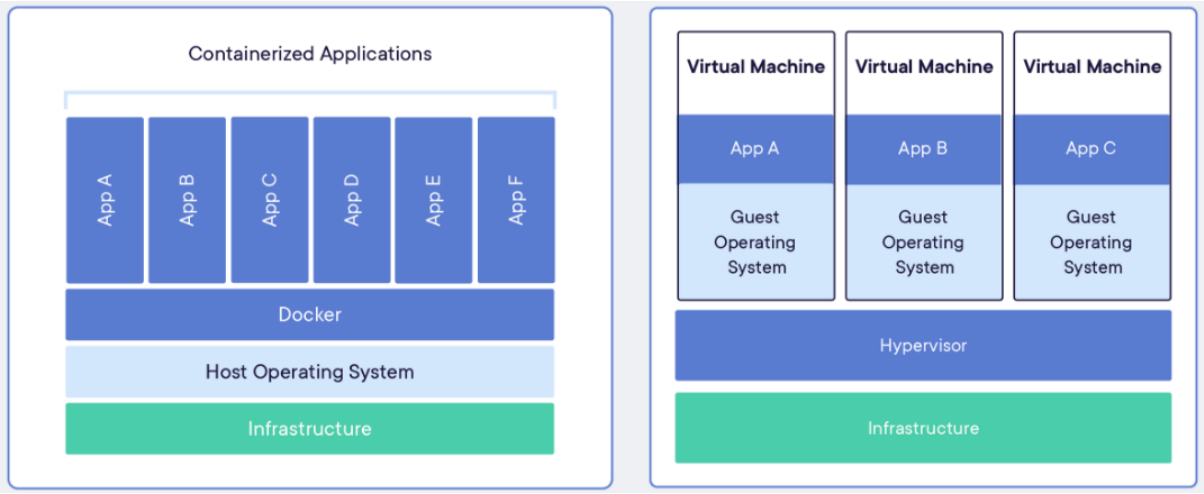


Figura 3: Comparativa entre ambos tipos de virtualización descritos

Si bien a priori la virtualización empleando contenedores (con implementaciones como Docker que delegan en otras a su vez como *libvirt*, LXC...etc [18]) puede parecer la forma definitiva de virtualización, esto no tiene por qué ser así ya que todo depende de las necesidades que se tengan.

Dependiendo de las necesidades a satisfacer emplear un tipo de virtualización u otra será la decisión adecuada.

Por último cabe aclarar que si bien existen distintos tipos de virtualización, cada una en un ámbito, en este apartado se han tratado las principales opciones de virtualización relacionadas con la ejecución y aislamiento de software en un equipo.

Otro tipo de virtualización en el ámbito de las redes que también conviene comentar es la virtualización de redes locales, empleando VLANs (estándar 802.1Q) la cual permite tener varios segmentos de red lógicos en capa 2 sobre un único segmento de red en capa 2 (o dominio de difusión) ya disponible [19].

2.3. Estado del arte en tecnologías de desarrollo de software web

A día de hoy y la hora de desarrollar un nuevo software, entre las decisiones sobre las tecnologías a usar o la plataforma en que implementarlo y ejecutarlo ha aparecido una nueva opción la cual está adquiriendo un peso considerable (si bien ya lo tiene): el empleo de tecnologías web.

En este proyecto se detalla el desarrollo de una aplicación web la cual permite regular las distintas relaciones de carácter administrativo que surgen entre los miembros de la residencia (ya sean residentes, personal administrativo y otro tipo de personal).

Puesto que hoy día es común que una aplicación esté comunicada con el exterior, el decantarse a desarrollar esta empleando tecnologías web es una decisión cada vez más habitual, dada la característica inherente a las aplicaciones web de estar siempre conectadas funcionando en red (generalmente).

Hablar sobre aplicaciones web conlleva mencionar a HTTP, el protocolo imperante en este tipo de aplicaciones [20]. Puesto que HTTP es un protocolo cliente-servidor [21] nos encontramos precisamente con que una aplicación web está distribuida entre el cliente y el servidor dadas las características de esta arquitectura.

Es por ello que cada componente de la aplicación recibe un nombre, siendo estos [22]:

- **Frontend:** parte que se ejecuta en el cliente
- **Backend:** parte que se ejecuta en el servidor.

2.3.1 Frontend

El frontend como bien se ha comentado es la parte de la aplicación web que se ejecuta en el cliente.

Existen diferentes tipos de clientes, desde aquellos que se ejecutan en el navegador web hasta aquellos que se ejecutan en otro tipo tipo de plataforma (como aplicación móvil para Android, iOS...etc. o aplicaciones de escritorio).

Lo más común es que todas estas aplicaciones se comuniquen con la parte backend empleando el protocolo HTTP.

Como se ha comentado la parte frontend de una aplicación web se puede ejecutar tanto en el navegador web como en otro tipo de plataformas.

En otro tipo de plataformas que no sea el navegador web nos encontramos con una gran cantidad de tecnologías disponibles para desarrollar la aplicación web. Esta puede ser hecha en casi cualquier lenguaje de programación (cualquiera que incorpore un cliente HTTP por ejemplo).

Cuando se trata del navegador web el abanico sigue sin resultar pequeño. En primer lugar cabe aclarar que comúnmente el lenguaje de programación empleado en los navegadores web es JavaScript, el cual es acompañado de otros lenguajes y tecnologías como son CSS y HTML que en conjunto permiten elaborar la aplicación web (página web).

El que JavaScript sea la tecnología ampliamente aceptada para programar aplicaciones en el navegador web conlleva que en torno a esta aparezcan multitud de frameworks implementados en JavaScript [24] que hacen más fácil el desarrollo de una aplicación web en el frontend.

Figura 4: En la actualidad existen una gran variedad de Frameworks Javascript

Entre estos frameworks se pueden destacar aquellos mantenidos por grandes empresas como Google y Facebook, como pueden ser AngularJS y React [25].

En torno a todas estas tecnologías surgen a su vez otras cuantas más como pueden ser preprocesadores (como Sass [26] para CSS y lenguajes alternativos que se traducen a Javascript como son CoffeeScript [27] y TypeScript [28]).

2.3.2. Backend

En lo referido al backend existen numerosas tecnologías disponibles. Basta con que una plataforma o lenguaje de programación proporcione la capacidad de crear un servidor (normalmente HTTP) para que este pueda ser empleado para implementar un backend.

Además en torno a la plataforma o lenguaje de programación se da la misma situación que la descrita para el frontend: proliferan infinidad de frameworks y librerías que hacen el desarrollo más fácil.

Por ejemplo para backend es muy habitual el emplear de las siguientes tecnologías, junto a sus correspondientes frameworks:

- **Python**: Flask, Django.
- **PHP**: Laravel, Symfony.
- **Nodejs**: Express.

Un backend además requiere de otros servicios en el sistema en el que se ejecuta, siendo el más común de estos el de un almacenamiento que proporcione

persistencia a los datos el cual acaba siendo en la mayoría de los casos una base de datos (la cual puede ser de distintos tipos [29], como relacional, no relacional…).

En lo referido a base de datos nos encontramos principalmente con bases SQL y no-SQL, ejemplo de estas pueden ser:

- **SQL**: PostgreSQL, MySQL, MariaDB.
- **No-SQL**: MongoDB, CouchDB, Redis.

3. Diseño, configuración y despliegue de la infraestructura de red

Este capítulo describe íntegramente cómo se han llevado a cabo el diseño, despliegue y configuración (por este estricto orden además) completos de una infraestructura de red en la Residencia de Estudiantes Alberto Jiménez Fraud.

3.1. Contexto

La residencia de estudiantes de la universidad de Málaga ha carecido desde sus inicios de una red inalámbrica rápida y robusta. Por otro lado la red cableada con la que ya contaba antes de llevar a cabo este proyecto presentaba una configuración la cual hacía que esta no ofreciera unas prestaciones demasiado elevadas.

Figura 5: Foto aérea de la Residencia, situada en el barrio de La Barriguilla

Fue a principios de 2017 cuando fue viable materializar la idea de renovar esta infraestructura, iniciándose así el proyecto el cual se encuentra aquí descrito.

No obstante el inicio de todo esto se remonta a 2014, año en el que mi compañero y amigo Melchor comenzamos nuestros estudios en Ingeniería del Software en la Universidad de Málaga y optamos por alojarnos en la mencionada residencia.

Al poco tiempo de entrar tuvimos la oportunidad de que se nos concediera a ambos una beca UMA (beca para colaboradores residencia universitaria) la cual hemos mantenido anualmente durante 4 años. El objetivo de la beca era el de mantener la infraestructura de red de la residencia universitaria. En esencia nuestra labor allí era la de administradores de sistemas.

La mencionada infraestructura de red con la que nos encontramos era poco robusta y presentaba fallos o caídas puntuales. No obstante nuestro empeño en mantenerla funcionando hizo que esta presentara el mayor nivel de usabilidad posible durante todo el tiempo que permaneció operativa.

Tal como se ha comentado, fue en 2017 cuando después de todo el trabajo y experiencia acumulados tuvimos la oportunidad (después de nuestras reiteradas solicitudes) de renovar y mejorar la red del edificio. El mantener reuniones periódicas y un permanente contacto con el director de las TIC de la Universidad de Málaga (Victoriano Giralt) permitió hacer viable el proyecto, permitiendo materializar este en un tiempo récord. Más adelante pudimos hacer de este parte de nuestro trabajo de fin de grado (el presente, el cual incorpora parte de infraestructura y servicios) contando además con la tutorización y enorme apoyo de nuestra también profesora Almudena Díaz.

En los siguientes apartados se describirán las diferentes situaciones por las que ha ido pasando el proyecto, junto a todos los detalles oportunos.

3.2. Estado inicial de la instalación

Antes de llevar a cabo el proyecto, la instalación de infraestructura de red disponible tenía las siguientes características:

- **100 puntos de red cableados** con cable de red **categoría 5e** (uno en cada apartamento).
- La interconexión de los puntos de red para tener una red local se lleva a cabo en 3 armarios rack distribuidos a lo largo del edificio. (más adelante se muestran unos planos). Dichos armarios poseían el equipamiento adecuado (switches gestionables en L3, SAIs...etc.).
- Cobertura WiFi de baja calidad en todo el recinto.
- Contaba con una conexión de 50Mb simétricos mediante fibra óptica (más adelante acabó siendo de 200Mb simétricos).
- Puerta de enlace con una distribución linux (Zentyal en aquél momento) que realizaba funciones de:
    - Firewall
    - Router (de ahí que contase con al menos dos tarjetas de red)
    - Calidad del servicio (*Qos*)
    - Servidor DHCP
    - Servidor DNS

La residencia se encuentra esencialmente dividida en 3 zonas, comúnmente nombradas como:

- Zona de los chicos: comprende los bloques 1,2,3 y 4 (apartamentos 1-48).
- Zona de las chicas: comprende los bloques 5,6,7 y 8 (apartamentos 48-90).
- Zona de recepción: se encuentra a la entrada del recinto.

En cada una de estas zonas se encuentra un armario rack (3 en total, tal como se ha mencionado arriba).

A continuación se muestra un esquema de red orientado cada uno a un armario distinto.

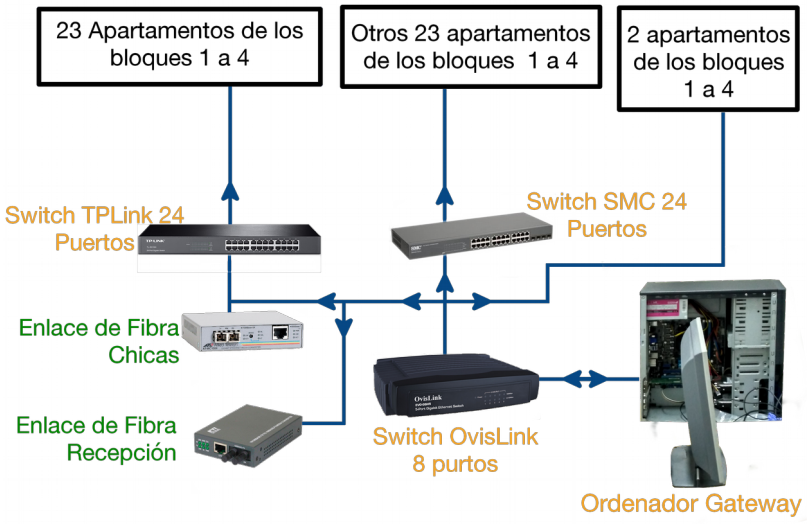


Figura 6: Armario de la zona de los chicos

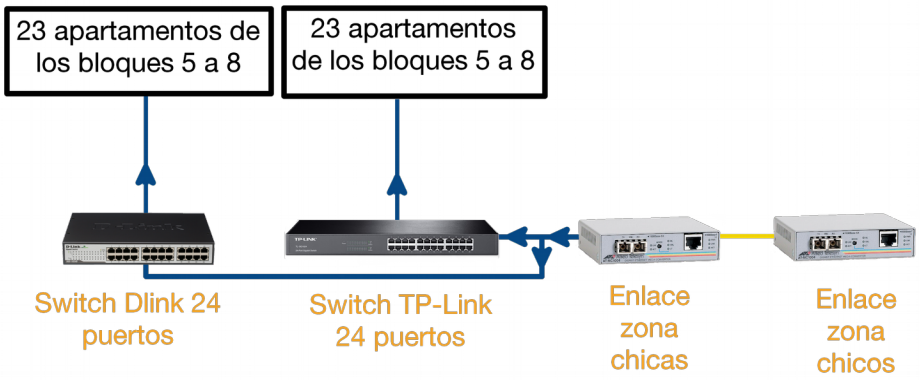


Figura 7: Armario de la zona de las chicas

Figura 8: Armario de recepción

Tal como puede observarse los distintos armarios se encuentran conectados entre estos mediante enlaces de fibra óptica multimodo.

En el anexo se adjuntan los planos de la red de datos cableada de la residencia para cada planta del edificio (figuras 10, 11, 12 y 13).

Cada punto de red cableada aparece aparece presentado en un recuadro negro en el plano, el cual sigue la nomenclatura ***{1,2,3}{A...}{1-10...}***. Dicha nomenclatura presenta 3 grupos, cada uno de los cuales representa, respectivamente:

- Número de rack al que se encuentra conectado el punto de red.
- Panel de parcheo dentro del rack (A, B, C…).
- Puerto del panel de parcheo al que se encuentra conectado el punto de red.

Como se ha comentado la cobertura WiFi en el recinto era de baja calidad debido a que esta se proporcionaba empleando un escaso número de routers domésticos, aún estando estos distribuidos de la forma más estratégica posible por el recinto.

Como primera aproximación para resolver este problema se decidió implantar el modelo "*bring your own device*" con el equipamiento de red, de forma que los propios usuarios se traen sus dispositivos [31] (puntos de acceso, routers...etc.) para lograr conexión inalámbrica. Con el incremento de la cantidad de estos dispositivos y la mala configuración que estos presentaban la estabilidad de la red cableada se vio afectada, haciendo que hubiese que retirar esta política.

La segunda aproximación fue la de instalar unos cuantos puntos de acceso *Unifi* en localizaciones estratégicas del recinto con objeto de ofrecer la mejor cobertura WiFi posible. Sin embargo debido a que se nos suministraron pocos de estos, no pudimos obtener el mejor resultado posible, aunque sin embargo la instalación de estos aparatos supuso una importante mejora en términos de conexión inalámbrica con respecto a la anterior.

A continuación se muestra un *heatmap* de intensidad de señal del WiFi que se provee en la residencia tras este enfoque:

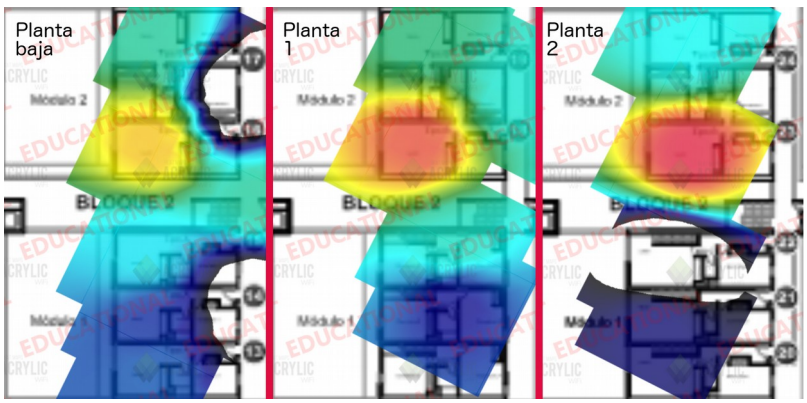


Figura 9: Diagrama de calor del bloque 2 para el WiFi de la residencia, la mayor intensidad de

señal aparece en rojo (aproximadamente unos -60dBm) y la menor en azul oscuro.

Como puede observarse en el diagrama aún con este nuevo enfoque la intensidad de calidad de la señal no es la suficiente para proporcionar una **conexión inalámbrica de calidad**.

3.3. Requisitos

Considerando todo lo anterior, y teniendo ya en el año 2017 la oportunidad de plantear un proyecto para la renovación de la infraestructura de red de la residencia con énfasis en la red inalámbrica, se fijaron una serie de requisitos que debía cumplir la nueva infraestructura de red de la residencia:

- Desplegar una red inalámbrica robusta y de calidad que proporcione cobertura a la totalidad del recinto.
- Proporcionar servicio de conexión inalámbrica empleando WiFi de 2.4GHz y 5GHz.
- En el punto con peor conexión inalámbrica posible, que el ancho de banda de esta no sea inferior a 30Mbps.
- Usar siempre que sea posible tecnología PoE para la alimentación eléctrica del hardware de red (puntos de acceso y teléfonos).
- Proporcionar conexión de red cableada con tolerancia a situaciones como *rogue dhcp.*
- Otorgar a la red de tolerancia a fallos eléctricos mediante el empleo de SAIs.
- Implantar una calidad del servicio que permita distribuir adecuadamente los recursos de red entre los distintos usuarios y dispositivos, primando la regla de que el uso que se hará de la instalación será muy variado y exigente (la emplean usuarios que llegan a vivir durante todo el año en la residencia).
- Disponer en la instalación de hardware que como mínimo sea *Gigabit*.
- Reforzar la velocidad, robustez y calidad de la infraestructura cableada ya existente.
- Implantar y diseñar un modelo que facilite el mantenimiento y supervisión de la red una vez que esta se encuentre en funcionamiento, con el objetivo de asegurar que esta opere en lo mejores niveles de calidad, y contando siempre con la mejor configuración y características posibles.

- Proveer al usuario que lo desee conexión por cable bajo un estricto control del uso de esta para evitar que su uso inadecuado perjudique a la red y al resto de usuarios.
- Desplegar **eduroam**: eduroam es un servicio de roaming que permite el acceso a internet y que ha sido desarrollado para el uso de la comunidad internacional de educación e investigación [32]. Empleando las credenciales de la institución académica a la que esté adscrito el usuario se le permite a esté emplear esta red en cualquier lugar el mundo en el que esté desplegada.

3.4. Diseño

Una vez establecidos y consensuados una serie de requisitos que debe cumplir la nueva infraestructura (e incluso algún que otro servicio funcionando sobre esta) se procede a la toma de decisiones de diseño.

Las decisiones de diseño se clasifican en esencia en aquellas relacionadas con la red inalámbrica y aquellas relacionadas con la red cableada. Como se extrae de los requisitos, la red inalámbrica debe diseñarse para que una vez desplegada y puesta en funcionamiento soporte una demanda media/alta por parte de los usuarios.

En primer lugar se ha procedido con la red inalámbrica, siendo las decisiones tomadas sobre el diseño de esta las que condicionen las tomadas sobre la red cableada.

El haberse decantado por este enfoque de en primer lugar diseñar la red inalámbrica y después la cableada hace que, evidentemente, sea la red inalámbrica la que condicione a la cableada. Se ha considerado que mencionado enfoque es más conveniente puesto que la red inalámbrica ofrece unas prestaciones (en términos de ancho de banda y disponibilidad de puntos de red, uno por apartamento) elevadas de forma que esta puede dar servicio a una red inalámbrica que exija una demanda media-alta.

3.4.1. Diseño de la red inalámbrica

El correcto diseño de una red inalámbrica para cumplir los requisitos establecidos ha pasado por una serie de etapas las cuales se tratan por orden de ejecución en los siguientes puntos.

Cabe destacar el carácter empírico de esta fase, caracterizada por, una vez que se tenían bien claras una serie de consideraciones, la puesta en marcha de un despliegue real a pequeña escala para medir y evaluar su rendimiento, haciendo esto que se puedan tomar decisiones fundamentadas sobre la ubicación de los puntos e incluso sobre lo conveniente que es contar con un hardware de red u otro (fiabilidad, facilidad de instalación y configuración, mantenimiento… y coste) de cara al despliegue final.

El haberse decidido por este enfoque se debe a que, a pesar de contar con resultados de estudios y simulaciones, en la realidad el entorno inalámbrico es bastante “hostil”, al presentarse numerosas interferencias de otra gran cantidad de equipos WiFi (router WiFi domésticos) instalados en el entorno residencial próximo a la residencia.

3.4.1.1. Valoración de opciones y propuestas

El primer lugar se hace necesario estudiar y comprender el entorno en el que se va a llevar a cabo el despliegue. Es por ello que se solicitaron a la dirección del centro los planos del recinto para poder realizar unas mejores valoraciones sobre la magnitud del despliegue.

La residencia universitaria se encuentra, en esencia compuesta por:

- Edificio de cafetería, situado en la parte inferior derecha de la figura 14, frente al bloque 1.
- Edificio de recepción y biblioteca, situado en la parte superior derecha de dicha figura, frente al bloque 8.
- Ocho bloques de apartamentos, numerados también en la figura. La residencia cuenta con 90 apartamentos.

Figura 14: Vista aérea del recinto

En cuanto a los bloques se ha de decir que estos son exactamente iguales entre ellos (idéntica estructura y distribución), a excepción del bloque 5. El cual es en efecto idéntico al resto de los bloques pero sólo incorpora una mitad.

En la residencia existen 3 tipos de apartamentos (uno de ellos de dos plantas, el tipo C) los cuales se encuentran configurados de una manera determinada en un bloque.

En la figura 15 puede observarse la planta de estos apartamentos y dónde se encuentran en un bloque, el cual consta de 3 plantas.

Conocida toda esta información era necesaria ir planeando una ubicación para los puntos de acceso, así como su cantidad. Tras estudiar la configuración del recinto y de los apartamentos, y la ubicación de los puntos de red (tal como se muestra en las figuras 10, 11, 12 y 13), deducimos una serie de conclusiones relacionadas con la ubicación de los puntos de acceso, y su número:

- Como la toma de red se encuentra en el salón, el punto de acceso debe ir en el salón para hacer más fácil la instalación. En el baño es inútil que se encuentre (por la presencia de azulejos, los cuales por su material dificultan la transmisión de las ondas electromagnéticas) y en los dormitorios no por la preocupación que esto puede producir en los residentes debido a la popular creencia de que el WiFi es nocivo para la salud, aunque todo apunta a que no [33]. En cualquier caso llegar hasta el dormitorio supone también una

dificultad técnica.

- En los apartamentos tipo C de dos plantas no hay toma de red en la planta de abajo, así que por dificultad técnica esta planta no es candidata a tener un punto de acceso.
- Considerando las ubicaciones de las antenas, por facilidad de instalación estas van a ser alimentadas empleando PoE.
- Los puntos deben ser de doble banda.
- Dado que una antena omnidireccional en una sola planta dio un buen resultado en esta, se va a proponer el enfoque de una antena por planta (salvo en el caso del dúplex, para le que se ingenia una forma de proporcionarle una adecuada cobertura a la planta que cuenta sin antena), y se suprime un enfoque inicial de una antena (punto de acceso por apartamento) ya que no es viable (coste, dificultad de instalación).

Tras estas conclusiones, se extraen de nuevo algunas más sobre las características del punto de acceso y su número:

- Dado que un bloque tiene 3 plantas y dos módulos, se pretende emplear una antena por planta, y aplicar este enfoque a cada módulo del bloque (consta de 2). De forma que en un bloque se pretende instalar 6 antenas. Considerando que un bloque consta de 12 apartamentos, habría una antena instalada por cada dos apartamentos. En el caso de la planta primera para los apartamentos de los dúplex, el punto de acceso correspondiente a esa planta es propuesto a ser instalado en la planta baja (contando esta con el suyo más el que iría en la planta primera).
- En lo que se refiere al edificio de cafetería se conserva el enfoque de tener un punto de acceso en el centro de esta, y en la planta superior.
- En lo que se refiere a la sala de estudios/biblioteca y recepción, se considera trasladar la ubicación del punto de acceso ya existente al interior de la biblioteca, y reemplazar este punto por uno de mejores características.
- Debido a la gran cantidad de antenas que van a existir, se hace necesario contar con un sistema que permita llevar a cabo una administración centralizada de estas.

3.4.1.2. Propuesta inicial

Una vez tenidas en cuenta las consideraciones expuestas con anterioridad, y tras haber mantenido además una serie de reuniones con expertos en el sector de las telecomunicaciones, se presenta la siguiente propuesta inicial:

- Por propuesta de distintos profesionales, y por experiencia propia al haberlo usado anteriormente, hacer el despliegue empleando el sistema Unifi [34]. Esto contribuye a facilitar enormemente el despliegue, configuración y mantenimiento.
- Se van a emplear puntos que soporten PoE (tanto activo como pasivo) y emiten WiFi en doble banda.
- Dada la alta densidad de puntos (esencial para otorgar una señal WiFi de calidad, sobre todo en 5GHz por sus característica de tener poco rango debido a su alta frecuencia [35]) se van a emplear puntos que aporten una potencia media.

Evaluando todos los puntos anteriores se encontró con un modelo que cumplía todas las características, el Unifi AP-AC Lite, del cual ya se contaba con buenas referencias.

Figura 16: Unifi AP-AC Lite

Tras todas las reuniones y haber considerado las distintas restricciones y requisitos, se hizo un planteamiento inicial de cómo podrían ir ubicados los puntos de acceso en un bloque, el cual se plasma en la siguiente figura 17.

Los puntos de acceso de la sala de estudios y edificio de cafetería/sala de reuniones/salón de actos presentan las ubicaciones descritas en los puntos anteriores.

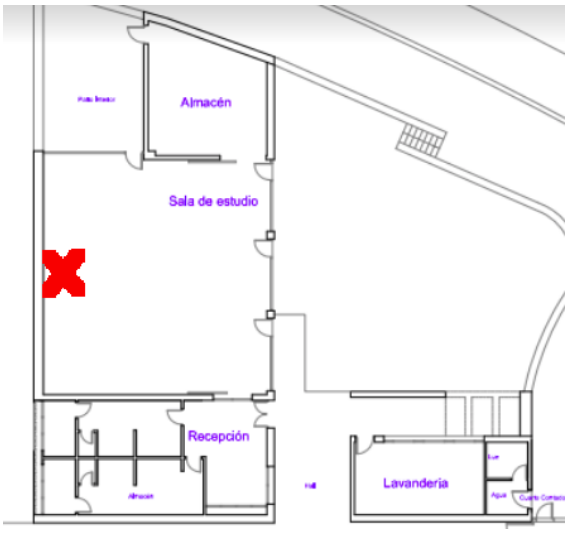


Figura 18: Ubicación del punto de acceso de la sala de estudios, anclado al muro

Del mismo modo para el edificio de cafetería:

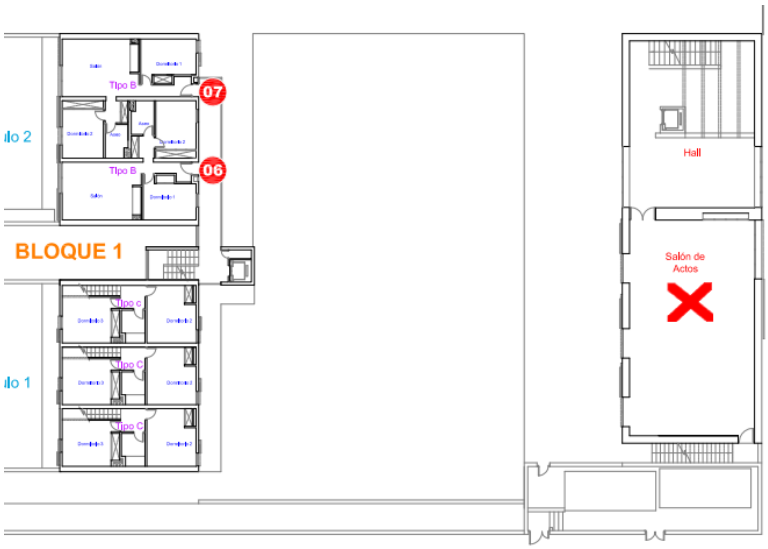


Figura 19: ubicación del punto de acceso del salón de actos/sala de reuniones

Teniendo todo esto se procedió a la compra de 3 puntos Unifi AP-AC Lite, con la intención de hacer los despliegues propuestos y medir y analizar su rendimiento. Puesto que en un bloque se encuentran 6 puntos de acceso y se disponían de sólo 3 (prestados),se montaron primero para una mitad (módulo) de un bloque y luego para la otra.

3.4.1.3. Análisis de rendimiento

Una vez montados los puntos se procedió a hacer una análisis cobertura y rendimiento empleando una equipo el cual contaba con una antena WiFi de doble banda con capacidad de funcionar en modo promiscuo. Con el software adecuado

se pudieron elaborar unos heatmaps de calidad de los cuales se extrajeron una serie de conclusiones muy valiosas. Estos pueden verse en la figura 20.

Como puede verse en los heatmaps el resultado aproximado es de bastante aceptación. Se hicieron además en todo momento tests de descarga y subida para constatar la calidad de la conexión. Como la cantidad y disposición de los puntos de acceso parecía adecuada, se procedió además a:

- **Adquirir** otros tres puntos (además de los que ya se tenían) e instalarlos en sus adecuadas ubicaciones, teniendo así el despliegue del bloque completo.
- **Proponer** a los residentes viviendo en ese bloque (un total de 36, en 12 apartamentos) que usasen la conexión inalámbrica durante una semana. Tras una semana de uso se procedió a hacerles individualmente una encuesta a cada persona para conocer su opinión sobre la conexión.

El resultado de las encuestas fue también bastante satisfactorio, por lo que se decidió emplear este diseño inalámbrico como definitivo.

3.4.1.4. Diseño final

El diseño definitivo que se decidió emplear consiste en el propuesto en el punto anterior, y conservando además las ubicaciones propuestas para el punto de acceso de la sala de estudios y de cafetería.

3.4.2. Diseño de la red cableada

Como se ha comentado anteriormente, el diseño de la red cableada quedaba supeditado al de la red inalámbrica, de forma que esta es diseñada de acuerdo a los requisitos que necesita satisfacer la inalámbrica.

A la hora de su diseño se han tenido en cuenta de nuevo una serie de consideraciones, resultando en las siguientes decisiones:

- Se conserva el cable Cat5-e ya existente.
- Anteriormente se mencionaba que una ventaja en la instalación era instalar el punto de acceso cerca del punto de red del apartamento, con la intención de conectar el punto de acceso a dicha toma. Sin embargo como se tiene previsto hacer un despliegue de telefonía IP y al mismo tiempo permitir que un residente pueda conectarse por cable, se ha decidido desplegar un cable

adicional hacia el apartamento que tendrá recorrido idéntico al ya existente en el apartamento, más el tramo que va desde la toma de red (donde ya acaba el tramo de tubo corrugado) hacia el punto de acceso, tramo por el cual el cable discurrirá por una canaleta por las zonas que según la situación particular del apartamento se estimen más oportunas en el momento de la instalación.

- Dado que el modelo de punto de acceso elegido es PoE y necesita un adecuado ancho de banda se ha decidido desplegar un cable Cat6 UTP (unifilar) hacia cada apartamento en el que vaya instalado un punto de acceso. Dicha categoría de cable ofrece una muy buenas prestaciones para la situación en la que sea emplearse. El haber optado por UTP se debe a que el cable no discurre por lugares en los que sea propenso a interferencias.
- Puesto que los switches con los que se cuenta no son PoE estos deben ser renovados por otros que sean PoE, además se amplía la cantidad de estos de cara a un futuro despliegue VoIP.
- Todo esto lleva a que es necesario cambiar los armarios rack con los que se cuenta por su tamaño insuficiente para albergar el nuevo equipamiento. Esta actuación debe hacerse en ambos racks presentes en el sótano.

En lo que se refiere a elección de equipamiento para esta etapa destacan los switches, y por temas de compatibilidad se ha optado por switches Unifi. En concreto por el modelo PoE de 750W y 48 puertos, el Ubiquiti US-48-750W.

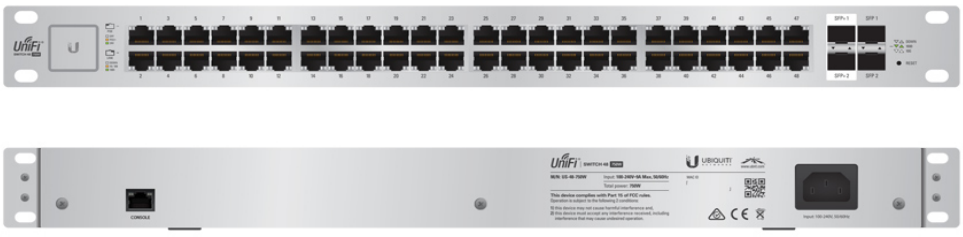


Figura 21: Switch PoE Ubiquiti US-48-750W

3.5. Despliegue

En este apartado se tratan todos los detalles relativos al proceso de despliegue.

Antes, durante y después del proceso se han generado una serie de documentos (hojas de cálculo e informes) los cuales se encuentran adjuntos a este documento para su consulta en la carpeta “despliegue”.

En los siguientes apartados se desarrollan las fases por las que ha transcurrido este.

3.5.1. Lista de materiales

La lista definitiva de material y equipamiento para llevar a cabo el despliegue es la siguiente:

- 47 puntos de acceso Unifi AP-AC Lite.
- 2 armarios rack de 19" (uno para zona chicos y otro zona chicas).
- En cuanto a la altura de los armarios rack debe ser la necesaria para albergar los switches, los paneles de parcheo y los SAI, y en el caso del rack de los chicos debe haber altura además para contener el o los servidores (router, centralita telefónica…). En cuanto a la profundidad de los armarios esta debe ser la mayor posible.
- 4 Switches Ubiquiti US-48-750W**.**
- Para el rack de recepción un switch PoE como el Ubiquiti Edgemax Switch-48-750W**.**
- 4 paneles de parcheo de 48 conexiones cada uno.
- 1 SAI Eaton 5SC-3000VA (para la zona chicos que soportan más carga) y otro Eaton 5SC-2000VA (para la zona de las chicas que no dispone de servidores).
- Para las terminaciones de los cables de red 300 conectores RJ45 para categoría 6 unifilar.
- Tras hacer los cálculos oportunos tomando medidas sobre los planos (aplicando la escala) y mediante revisión in situ de la instalación, serán necesarios 10Km de cable CAT6 UTP.

- 300 metros de canaletas superficiales.
- Mínimo 2 guías para meter cable.
- Crimpadora e impactadora.
- 150 latiguillos, de no más de 1 metro..
- Dado que se van a renovar las rosetas de red, 90 rosetas de red CAT6 completas (con embellecedor incluído).

Cabe mencionar que el precio de este material se desconoce al ser adquirido y gestionado por la UMA. Por otro lado a la hora de solicitar el material se ha hecho de la forma más ajustada posible, con intención de que al finalizar la instalación el exceso de este (como puede ser de cable) sea mínimo.

3.5.2. Plazo y forma de ejecución

El despliegue se lleva a cabo durante el mes de julio de 2017, por 8 voluntarios en total. Los voluntarios disponen de ese mes de comida y alojamiento cubiertos por la empresa concesionaria de la residencia y por la UMA.

3.5.3. Lista de tareas a llevar a cabo en el despliegue, lugar y herramientas necesarias

Las tareas que se llevarán por tanto a cabo durante el despliegue son:

1. **Instalación de antenas Wi-Fi**: se instalarán 6 antenas Wi-Fi por bloque de la residencia (3 en cada módulo). De esas 6, 4 irán montadas en pared atornilladas a pladur y 2 de ellas pegadas al techo de hormigón con el adhesivo adecuado. Lugar: apartamentos de la residencia.
2. **Montaje de canaletas**: por fines estéticos y de protección del cableado se instalarán aproximadamente 6 metros de canaleta autoadhesiva en cada uno de los apartamentos donde se instale una antena Wi-Fi. Están discurrirán tanto por el techo como por la pared y deberán ser cortadas a la medida adecuada.
   Lugar: apartamentos de la residencia.
3. **Instalación de rosetas de red:** dado que las rosetas de red de los apartamentos se van a quedar anticuadas cada roseta que se encuentra en

cada apartamento va a ser renovada por otra. Esto supone instalar una roseta de red por cada apartamento de la residencia (será necesario atornillar y desatornillar e impactar cables, y si es necesario cubrir o reafirmar con masilla algunas cajas empotrables).

Lugar: apartamentos de la residencia.

4. **Despliegue de cableado en los apartamentos adecuados:** Al cableado existente se añadirá cableado adicional para soportar la instalación de los aparatos descrita en el punto 1. El cable deberá ser pasado por los distintos lugares por los que ya discurre el actualmente instalado, de forma que se tendrán recorridos verticales y horizontales por tubo corrugado y recorridos horizontales por bandejas fijadas al techo.

   Lugar: apartamentos de la residencia, trasteros del sótano, pasillos del sótano, túneles laterales del sótano, registros de los apartamentos.

5. **Desmontaje de los armarios rack:** los armarios rack actuales que lo requieren (los que se encuentra en la zona de los chicos y de las chicas) y su equipamiento van a ser desmantelados con objeto de poder instalar los nuevos junto al nuevo equipamiento.

   Lugar: trasteros del sótano.

6. **Instalación del equipamiento en los armarios rack:** en los armarios rack nuevos se instalará el nuevo equipamiento que comprende paneles de parcheo, regletas, SAI’s, switches, servidores...

   Lugar: trasteros del sótano.

Para llevar todo a esto a cabo se han requerido herramientas básicas tales como:

- Destornillador
- Taladro
- Martillo
- Crimpadora
- Impactadora
- Alicate
- Multímetro
- Pelacables
- LAN tester
- Linternas
- Sierras manuales (para cortar las canaletas).

3.5.4. Desarrollo del despliegue

Como se ha comentado el despliegue se desarrolló durante el mes de julio de 2017 por un equipo compuesto por un total de 8 voluntarios. Se trabajaron todos los días del mes tanto por la mañana como por la tarde, estando organizado el trabajo de los voluntarios en distintos turnos (de forma que un voluntario en un día o trabajaba por la mañana o por la tarde). Cabe destacar que Melchor y Antonio trabajaban siempre tanto de mañana como de tarde. Los detalles de la planificación horaria y en general la gestión de recursos humanos se encuentran adjuntos a este documento.

Del párrafo anterior se desprende que el trabajo durante el mes de julio fue intensivo, esto se debe principalmente a:

- Se quería evitar que este ocupase el mes de agosto para respetar el periodo vacacional.
- Era necesario tener el despliegue hecho lo antes posible para que diese tiempo a hacer pruebas y configuraciones para dejar la instalación lista de cara al inicio del curso en septiembre.

Cabe destacar el **carácter didáctico de la actividad** puesto que los residentes voluntarios involucrados en el despliegue, los cuales no necesariamente se encontraban cursando estudios relacionados con las telecomunicaciones o

informática, **recibieron además una formación** consistente en:

- **Conocimientos básicos sobre redes**: Qué es una red, o internet, y cómo funcionan estos. Se trataron los aspectos básicos que permiten comprender cómo funcionan estos.
- **Uso de herramientas**: Se dieron a conocer distintas herramientas y se explicó su uso.
- **Técnicas de instalación y despliegue**: Se ofreció formación relacionada con tender cable, usar guías, terminar conexiones (crimpar), instalar canaletas en superficie...etc.
- Pautas de **seguridad** en el trabajo: A todos los voluntarios se les dió una formación básica en materia de seguridad a la hora de trabajar. Durante su trabajo fueron además supervisados con objeto de que trabajasen en condiciones de seguridad.

Al respecto del último punto cabe destacar que además:

- Los voluntarios fueron equipados con EPIs, constando de:
    - Casco
    - Gafas
    - Mono de trabajo
    - Guantes
    - Botas de seguridad
    - Máscara para trabajar en ambientes con partículas en suspensión.
- Los voluntarios estaban asegurados durante todo el mes.

En las figuras 22, 23, 24, 25, 26, 27, 28 y 29 se puede observar cómo se fue desarrollando el despliegue y el empleo de EPIs en todo momento para llevar a cabo este.

Acabo julio el despliegue fue acabo en su mayoría, quedando pendientes algunos detalles, los cuales se finalizarían a inicios de septiembre.

3.5.5. Configuración mínima para el funcionamiento

Una vez acabado completamente el despliegue se procedió a aplicar una configuración básica al equipamiento de red instalado. Para ello se instaló en el servidor el software controlador de Unifi, el cual permite gestionar y configurar todo el equipamiento de Unifi.

Tras una primera configuración básica y siguiendo una serie de pautas para hacerla más óptima se logró que todos los aparatos instalados, entre los que se encuentran 5 switches y 47 puntos de acceso, funcionasen de forma muy aceptable.

En las figuras 30, 31, 32 y 33 se aprecia la ubicación de los puntos de acceso en el recinto, los cuales dan cobertura a la totalidad de este. En la figura 34 se muestra el resultado de instalación de cable.

3.5.5.1. Pruebas y medida del rendimiento

Para comprobar que la instalación estaba debidamente realizada y configurada se procedió a poner esta a prueba de la mejor forma posible: nos esperamos unos días al inicio del curso a mediados o finales de septiembre a que llegaran todos los residentes, a los que les suministramos acceso a una red de pruebas.

Durante las semanas siguientes estuvimos haciendo tests de velocidad por el recinto, monitoreando el uso de la red y entrevistandonos con varios residentes al mismo tiempo que nosotros mismos también usábamos la red.

La conclusión de todo esto fue que la instalación estaba en un estado decente para ser utilizada y exprimida, y que a partir de ese momento nos dedicaríamos a ir haciendo una serie de mejoras a esta.

3.5.6. Mejora en la configuración y otras consideraciones/ampliaciones futuras

Las mejoras que se han considerado hacerle a la instalación de red han sido las siguientes:

- Sustituir las bombillas incandescentes próximas a los puntos de acceso por LED (para evitar el sobrecalentamiento de estos).
- Trasladar el servidor de un recinto de comunicaciones adicional al armario rack de los chicos.
- Implantar un sistema de refrigeración adecuado (empleando splits probablemente) para el equipamiento albergado en el rack de la zona de los chicos (switches y servidores).
- Aumentar el ancho de banda contratado para llegar a 1Gb simétrico (la red tiene capacidad para soportar estas velocidades).
- Virtualizar los servicios de red, con objeto de facilitar el mantenimiento y uptime. Esto se trata en profundidad en el trabajo desarrollado por Melchor.
- Implantar un sistema de **telefonía IP**, esto se trata en este mismo documento.
- El punto anterior lleva además a que se hace necesario pasar de un primario de telefonía a un SIP Trunk.
- Ofrecer un servicio conocido como “Panel del residente” el cual le permite al residente gestionar su relación con la residencia en unas excelentes condiciones. Esto se trata en este mismo documento.
- Implantar la red **eduroam**: emplea autenticación WPA 2 empresarial con el método de autenticación EAP-TTLS, el cual es un túnel de un protocolo PAP. Tener eduroam desplegado va a ser muy importante debido a que será el nuevo Wi-Fi usado en toda la UMA, y por ende tras esta implementación en la residencia.
- Poner en funcionamiento varias VLAN que permitan segregar el tráfico en red local, aumentando la seguridad de la red y facilitando la implantación de una calidad del servicio (QoS) para la telefonía IP.

3.5.7. Conclusiones

A modo de conclusión ha de decirse que se ha llevado a cabo un enorme trabajo para contar con la infraestructura actual.

El resultado final ha sido bastante bueno y cada día se sigue investigando la forma de mejorarla aún más.

La experiencia vivida por todas las personas involucradas en esto ha sido única y

excelente, así como lo ha sido la basta cantidad de conocimiento obtenido a raíz de esto.

Otros residentes sucederán a Melchor y a Antonio en lo que se refiere a desempeñar la función de becarios administradores de red, los cuales recibirán la adecuada formación para seguir desempeñando adecuadamente sus funciones.

4. Despliegue de telefonía IP

Contando ya la residencia con una infraestructura de red capaz de dar soporte a numerosos servicios, se le ha decidido dar cabida a la telefonía IP. Es sin duda uno de los pasos esenciales en el continuo trabajo por la modernización de las TIC de la residencia.

4.1. Situación actual y justificación del nuevo despliegue

Desde sus inicios la residencia ha contado con un sistema de telefonía compuesto por unos 100 terminales analógicos, presentes en apartamentos (uno por apartamento) y distintas dependencias del edificio como son cafetería, lavandería y oficinas.

Estos teléfonos se encuentran conectados a una centralita telefónica a través de una línea de par de cobre. La centralita a su vez se conecta a la red pública de telefonía mediante un primario RDSI, el cual ofrece una capacidad para 30 conversaciones simultáneas [36].

El diagrama de la instalación de teléfono actual es el siguiente:

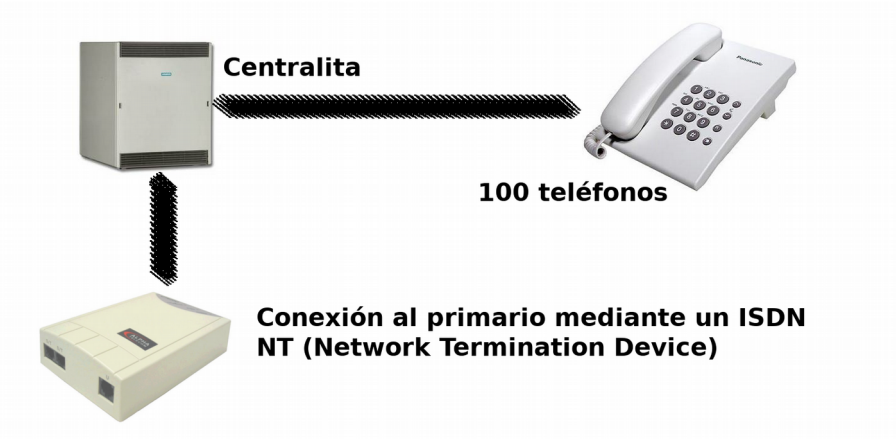


Figura 35: Despliegue actual de telefonía IP

El despliegue de teléfono mostrado ha dejado de ser adecuado para satisfacer las necesidades que se tienen hoy en día debido a una serie de fallos y limitaciones que posee debido a su antigüedad. Esto unido a la intención de la Universidad de

Málaga en emplear telefonía IP en todas su instalaciones hace necesario que en la residencia también se disponga de dicho despliegue.

4.2. Requisitos de la nueva instalación de teléfono

El nuevo despliegue de telefonía IP debe satisfacer los siguientes requisitos:

- El modelo de teléfono IP físico a emplear es el Yealink SIP-T21P E2, al haber sido el suministrado por la universidad. Se puede ver en la figura 36.
- Los teléfonos serán alimentados mediante PoE activo (proporcionado por los switches) siguiendo el estándar 802.3af.
- El códec preferido para las transmisiones de audio será OPUS debido a sus excelentes características.
- El teléfono se encontrará bloqueado para los usuarios de forma que el acceso de estos a sus funciones será mínimo (hacer/recibir llamadas y consultar agenda esencialmente).
- En términos de instalación el teléfono se encontrará anclado al muro.
- El teléfono se encontrará conectado al punto de red del apartamento.
- Si el usuario desea conectarse a la red por cable, lo hará a través del switch que incorpora el teléfono.
- Todos los teléfonos mantendrán una agenda obtenida de un servidor la cual poseerá números de interés los cuales podrán ir siendo agregados con el tiempo.
- Se debe emplear la característica de auto-provisioning de los teléfonos de forma que estos obtienen sus configuraciones (datos de registro, zona horaria, idioma, bloqueo de teclado…) de un servidor de configuración.
- Mediante el empleo de **static leases** en el servidor DHCP, cada teléfono tendrá su propia dirección IP y hostname basado en la MAC del teléfono. Esto es esencial para que el servidor de autodespliegue pueda identificar qué teléfono le está haciendo la petición de configuración en base a su dirección MAC, y así entregarle una dirección MAC personalizada.
- Si en algún momento se quiere aplicar una configuración a todos los teléfonos de la residencia, bastará con reflejarlo así en el servidor de despliegue y esta se irá propagando a todos los teléfonos.
- El software empleado para la PBX será Asterisk.

- Asterisk funcionará en un entorno virtualizado, empleando virtualización mediante contenedores.
- El driver de canal empleado en Asterisk será pjsip.
- Las configuraciones de los distintos módulos de asterisk se encontrarán almacenadas en una base de datos relacional (como MariaDB).
- Habrá servicio de buzón de voz.
- La PBX proporcionará características como transferencia de llamadas, puesta en espera de llamadas...etc.
- Se ofrecerá la opción de que el PBX ejecute aplicaciones implementadas en distintos lenguajes de programación (como Python) gracias a la interfaz que ofrece asterisk basada en websockets (ARI, Asterisk RESTful Interface).
- La gestión de los teléfonos se podrá llevar a cabo desde un software centralizado de gestión de la residencia.
- Desde el panel del residente ya sea un residente o un administrador podrán consultar el registro de llamadas de un teléfono concreto.
- El protocolo empleado de señalización es SIP.
- Siempre que sea posible se debe emplear **seguridad en la capa de transporte para el protocolo empleado de señalización.** Por tanto deseablemente se ha de emplear SIPS (SIP sobre TLS).
- Del mismo modo para la transmisión de sonido se debe usar la versión segura del protocolo RTP: SRTP.
- Existirá una cola de espera para las llamadas entrantes la recepción de la residencia, de forma que a un usuario puesto en esta se le reproducirá una música de fondo.

4.3. Instalación de los teléfonos

En cuanto se recibieron los teléfonos se procedió a la instalación física de estos (realizada esta vez por Melchor y Antonio), instalando uno por apartamento y en las distintas dependencias de la residencia como son oficinas, cafetería y lavandería.

De todas estas ubicaciones la que plantea mayor dificultad eran los apartamentos por una serie de motivos:

1. Encontrar la ubicación correcta del teléfono en el muro: Puesto que cada apartamento tiene una disposición diferente del mobiliario (más las modificaciones que cada residente hace), resultaba difícil encontrar una posición concreta en el muro en la que instalar el teléfono.
2. Puesto que se le quería ofrecer a los residentes conexión por cable era necesario dejar a disposición de estos el switch del teléfono. Dado que el switch del teléfono se encuentra en la parte trasera de este, quedaba inaccesible una vez anclado en la pared.

Como solución al punto 1 se decidió instalar el teléfono inmediatamente encima del punto de red de todos los apartamentos a una altura coherente en base a la estatura media de sus usuarios, y prestando especial atención a los apartamentos dedicados a PMR, en los cuales el teléfono se ha instalado a una altura considerablemente menor.

Además se ha empleado una canaleta de superficie por fines estéticos y para lograr protección del cableado. En la figura 37 se puede encontrar el resultado de la instalación.

En lo que se refiere al punto 2 se ideó un sistema el cual traslada la boca del switch del teléfono desde la parte trasera a la parte delantera, permitiendo al usuario una conexión fácil a este. En la figura 38 se aprecia cómo se ha aplicado este sistema.

4.4. Conceptos

En esta sección se presentan una serie de conceptos esenciales sobre sonido, el protocolo SIP y asterisk.

4.4.1. Conceptos generales sobre audio

El hablar de telefonía significa hablar de audio, de sonido. Al fin al cabo es uno de los elementos principales con los que el software PBX que se ha escogido (asterisk) trabaja, aunque también es capaz de trabajar con video, ofreciendo así la opción a hacer videoconferencias.

Para poder comprender mejor cómo asterisk trabaja y gestiona el audio (codificando, decodificando e incluso transcodificando) es necesario aclarar unos conceptos básicos sobre audio.

4.4.1.1. Conversión analógico digital y viceversa

El sonido es una onda mecánica y por ende con carácter puramente analógico.

Entonces surge la cuestión de cómo un equipo digital trabaja con este. Para lograr esto se incorporan en dichos equipos de grabación y reproducción los denominados DAC (conversor de digital a analógico) y ADC (conversor de analógico a digital), los cuales se encargan de realizar las conversiones pertinentes entre analógico y digital y viceversa.

Cuando por ejemplo se usa un micrófono, el audio analógico es recogido por la tarjeta de sonido y esta lo convierte en una señal digital, en este paso el audio ya está siendo codificado y convertido a digital usando tecnologías como LPCM o PCM [38], en las que el audio no tiene pérdidas. Dicha salida digital es la que se le proporciona como entrada al equipo digital.

Del mismo modo para reproducir sonido se le suministra a la tarjeta de sonido el sonido codificado en PCM o LPCM (PCM lineal) y esta se encarga de generar una señal analógica audible.

4.4.1.2. Contenedores de sonido y codecs

Trabajando con sonido es inevitable que aparezcan estos conceptos, el de contenedor de sonido y el de codec.

En primer lugar un codec (de audio) es un algoritmo que implementa una técnica que permite, dada una entrada de una señal de entrada de sonido, obtener una salida codificada, o bien el proceso inverso [39]. Existen codecs con pérdida (la información de salida requiere de menos espacio para su almacenamiento a cambio de perder calidad) y sin pérdida.

PCM es por ejemplo un códec sin pérdida [38], aunque muy primitivo ya que es el empleado por una tarjeta de sonido.

Existen otros codecs como el conocido MP3, el cual es con pérdida y se encarga de:

- Dado sonido digital codificado en PCM, codificarlo para que ocupe menos espacio.
- Dado sonido digital ya codificado en MP3, decodificarlo en PCM para ser mandado a una tarjeta de sonido.

En segundo lugar existe el concepto de contenedor de audio: Un contenedor de audio define un formato de archivo estructura (un contenedor) que permite almacenar audio codificado junto a información de este (metainformación) [40].

Existen numerosos contenedores (no sólo de audio). Un ejemplo de estos es WAV, el cual permite almacenar audio codificado en distintos formatos junto a información sobre este [41].

Existen otros contenedores como el de MP3 que sólo está hecho para almacenar audio codificado como MP3.

4.4.2. Conceptos generales de SIP

SIP es un protocolo que funciona en la capa de aplicación empleado para crear, modificar y finalizar sesiones de uno o más participantes [42].

SIP por tanto es un protocolo bastante **flexible** con **muchas aplicaciones**, siendo la

principal la de **VoIP**.

Puesto que no es viable resumir todo el protocolo SIP en este documento, sí se van a comentar unos conceptos esenciales sobre este.

4.4.2.1. Métodos SIP

El protocolo SIP recuerda bastante al protocolo HTTP. Al igual que en HTTP, en SIP existen distintos verbos o métodos. Los principales y su uso son:

- **REGISTER**: permite registrar un dispositivo en un AoR específico. El dispositivo suministra su dirección de contacto y el AoR contra el que desea registrarse. En esta misma petición el dispositivo puede adjuntar credenciales con fines de autenticación.
- **INVITE**: permite realizar una llamada. Por ejemplo si un teléfono SIP quiere llamar a otro enviará este mensaje al PBX.

Para información más detallada consultar [42].

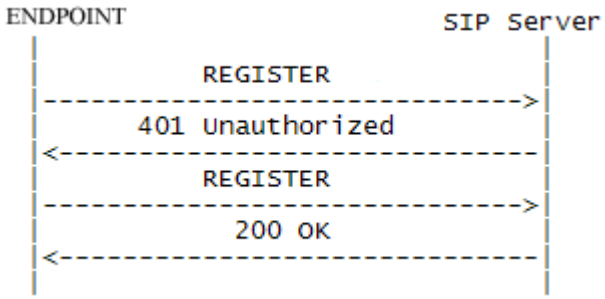


Figura 39: Procedimiento de registro entre un teléfono y un PBX

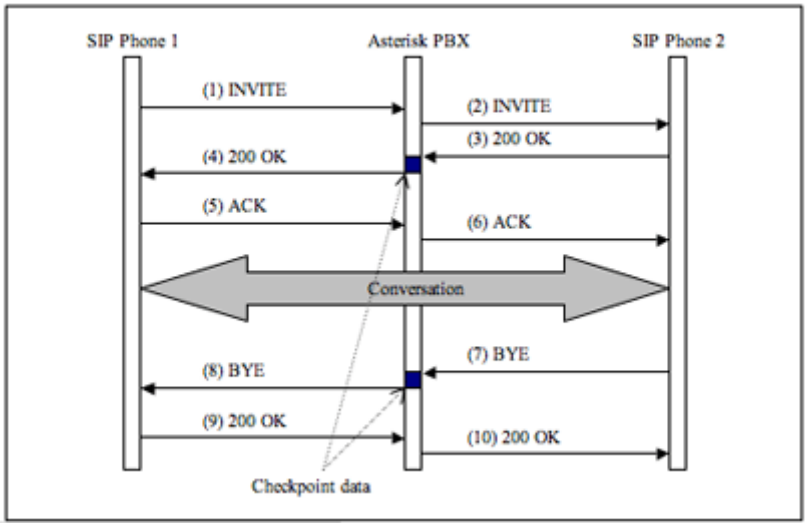


Figura 40: Procedimiento de iniciación de una llamada

4.4.2.2. Direccionamiento en SIP

En SIP existen distintas direcciones [43] entre ellas:

- **Address of Record (AoR)**: Es la dirección más abstracta y de más alto nivel que se le asigna a una entidad en un sistema SIP, del estilo usuario@dominio.com. Como se verá más adelante el AoR es una estructura de datos en asterisk que permite almacenar una lista de contactos.
- **Dirección de contacto:** Puesto que una entidad puede tener varios dispositivos registrados en un mismo AoR, existe la dirección de contacto (contact address), la cual es generalmente de la forma <nombre_aor>@direccionIP_endpoint.

4.4.3. Conceptos sobre asterisk

Asterisk maneja una serie de conceptos, algunos de ellos propios y otros que son análogos a los existentes en SIP. A continuación se enumeran y definen unos cuantos:

- **Endpoint**: Dispositivo final del usuario, como un teléfono físico o software.
- **Protocolo de señalización**: Protocolo empleado para la señalización de las llamadas, como puede ser SIP.
- **Canal**: Conexión establecida entre un endpoint y asterisk para el intercambio de información multimedia.
- **Puente**: Concepto que describe el hecho de que dos o más canales **compartan** información multimedia. Por ejemplo durante una llamada los canales de los endpoints se colocan en un mismo puente.
- **Driver de canal**: Componente software que permite hacer una traducción entre el protocolo de señalización que esté empleando el endpoint y el protocolo propio y abstracto de señalización de asterisk. Este enfoque hace que dos dispositivos con protocolos de señalización distintos puedan comunicarse entre ellos. El driver de canal más común en asterisk es pjsip.
- **Dialplan**: El dialplan es una de las partes más importante de una centralita asterisk. Se trata de un archivo (aunque también puede ir almacenado en una base de datos) el cual describe todo el compartimiento de la centralita. Es pieza clave puesto que, como se vuelve a recalcar, describe cómo funciona la

centralita en su totalidad en términos de definir extensiones y comportamiento de estas. Cuando se llegue a la parte de configuración se hablará en detalle sobre el dialplan diseñado para la residencia.

- **SIP Trunk**: El SIP Trunk permite que el PBX puede hacer y recibir llamadas desde/de internet. De forma típica un SIP Trunk es un servicio que ofrecen los ISPs y que permite que un PBX tenga comunicación con la red telefónica pública.

Para información aún más detallada consultar la documentación oficial de asterisk en [44].

4.4.3.1. Driver de canal PJSIP

Se trata del driver de canal más empleado y a su vez el que se va a emplear en este despliegue. PJSIP maneja una serie de conceptos en sus configuraciones los cuales están en estrecha relación con muchos otros de SIP, entre ellos se encuentran:

- **ENDPOINT**: Término abstracto que hace referencia a un lugar hacia el que se puede hacer o desde el que se puede recibir una llamada. Un ENDPOINT se encuentra asociado a un AoR. El ENDPOINT constituye un perfil que describe la configuración de un teléfono (códec permitidos, contexto en el dialplan…).
- **AUTH**: Permite configurar detalles de autenticación para un ENDPOINT como nombre de usuario y contraseña.
- **AoR**: Estructura de datos que almacena distintas direcciones de contacto para un ENDPOINT.
- **REGISTER**: Describe cómo la centralita se registra con otro proveedor para hacer llamadas, por ejemplo para configurar un SIP Trunk.

4.5. Constitución del entorno de pruebas

Para poder hacer las distintas pruebas y ensayos necesarios, se ha constituido un entorno de pruebas para desarrollar una infraestructura VoIP experimental en miniatura cuya arquitectura se describe en la siguiente figura:

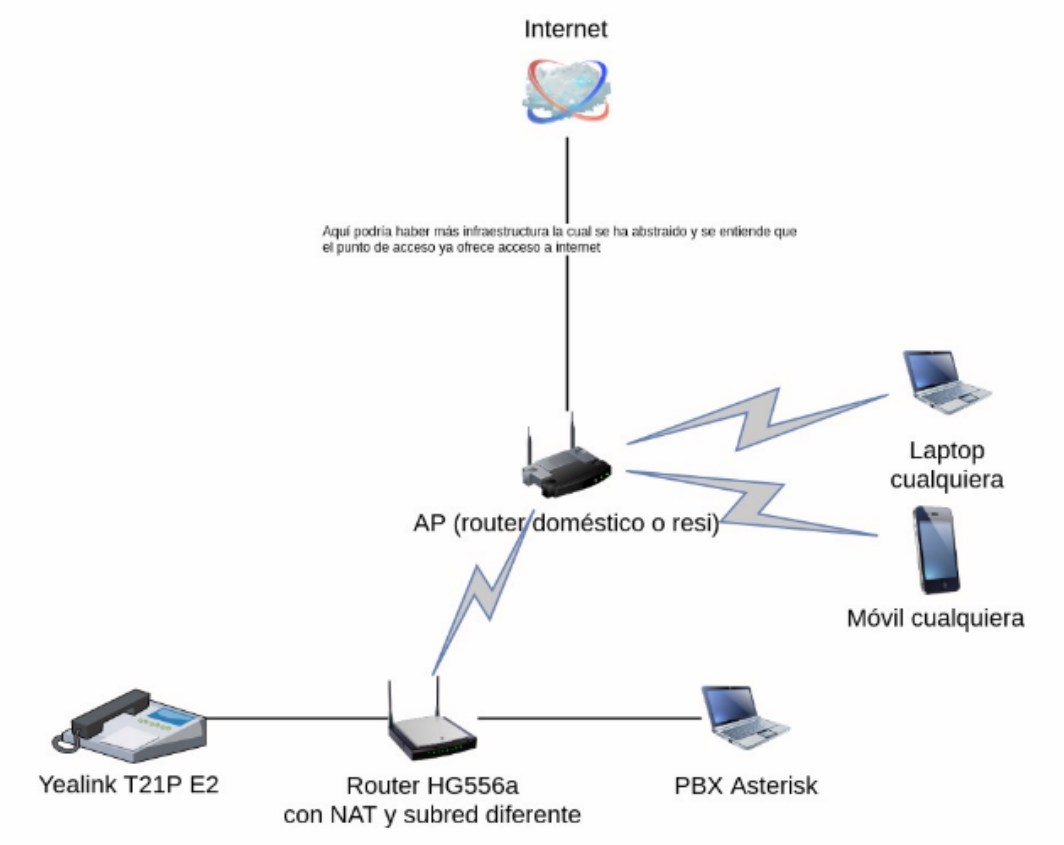


Figura 41: Infraestructura de pruebas para VoIP

Esta infraestructura se ha creado configurando un router adicional (el HG556A) el cual crea una red local para hacer todas las pruebas pertinentes sin afectar al resto de la red. Además proporciona conexión a internet gracias a su característica de cliente inalámbrico.

En dicha red local se encuentran conectados el PBX (asterisk) y el teléfono Yealink empleado para el despliegue. Este planteamiento permite hacer captura de paquetes para entender mejor el funcionamiento del PBX y las comunicaciones que ocurren entre el o los teléfonos y entre estos y la centralita.

Una forma muy cómoda de capturar tráfico con esta infraestructura es ejecutando el siguiente comando de bash sobre un cliente que tenga wireshark instalado:

```
ssh root@192.168.2.1 "tcpdump -i br-lan -U -s0 -w - 'not port 22'" | wireshark -k -i -
```

Este comando ejecuta tcpdump en el router sobre la interfaz adecuada y eliminando el tráfico correspondiente al protocolo ssh, y redirecciona la salida a la entrada de wireshark en el cliente ssh. De esta forma es fácil analizar todo lo que está

ocurriendo en la red.

En la siguiente figura se puede ver un ejemplo de una captura de un registro de un teléfono SIP contra la centralita:

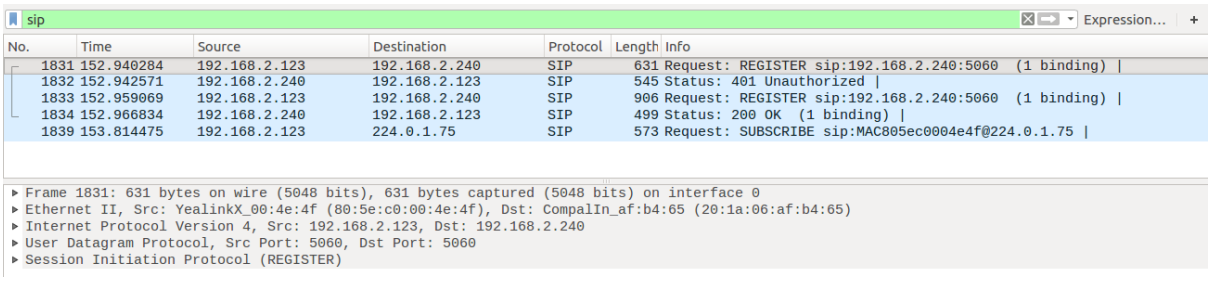


Figura 42: Registro de un teléfono SIP en la centralita

4.6. Constitución de la infraestructura de virtualización

El objetivo que se persigue es el de tener la mayor cantidad posible de servicios en red virtualizados. Es por ello que para el desarrollo del sistema de telefonía IP la virtualización ha sido uno de los aspectos que han estado presentes desde el primer momento.

Como se describe en los requisitos iniciales del proyecto, el objetivo es integrar el sistema de telefonía IP con un sistema de administración (Panel del Residente), ambos evidentemente virtualizados y cooperando, y todo esto funcionando en un entorno de VLANS que permita una correcta segregación del tráfico de red.

En esta sección se presenta el cómo se ha llevado a cabo y planificado la virtualización del sistema de telefonía de forma independiente (sólo el sistema de telefonía IP), pero considerando en su diseño y configuración que este va a ser integrado en una infraestructura de virtualización aún mayor.

Para constituir toda la infraestructura de virtualización mediante contenedores se ha empleado Docker. Los servicios sujetos de virtualización son:

- **Base de datos relacional MySQL**: Es empleada por el software PBX asterisk para persistir configuraciones tales como la información de registro de los usuarios del sistema y el historial de llamadas.
- **Asterisk**: PBX o centralita

4.6.1. Conceptos de Docker

En esta sección se presentan una serie de conceptos esenciales sobre el software Docker, el cual hace posible la virtualización de servicios mediante contenedores.

Con objeto de no extender la sección, se presentan los conceptos de forma resumida, puesto que en [54] se trata este asunto en mayor profundidad.

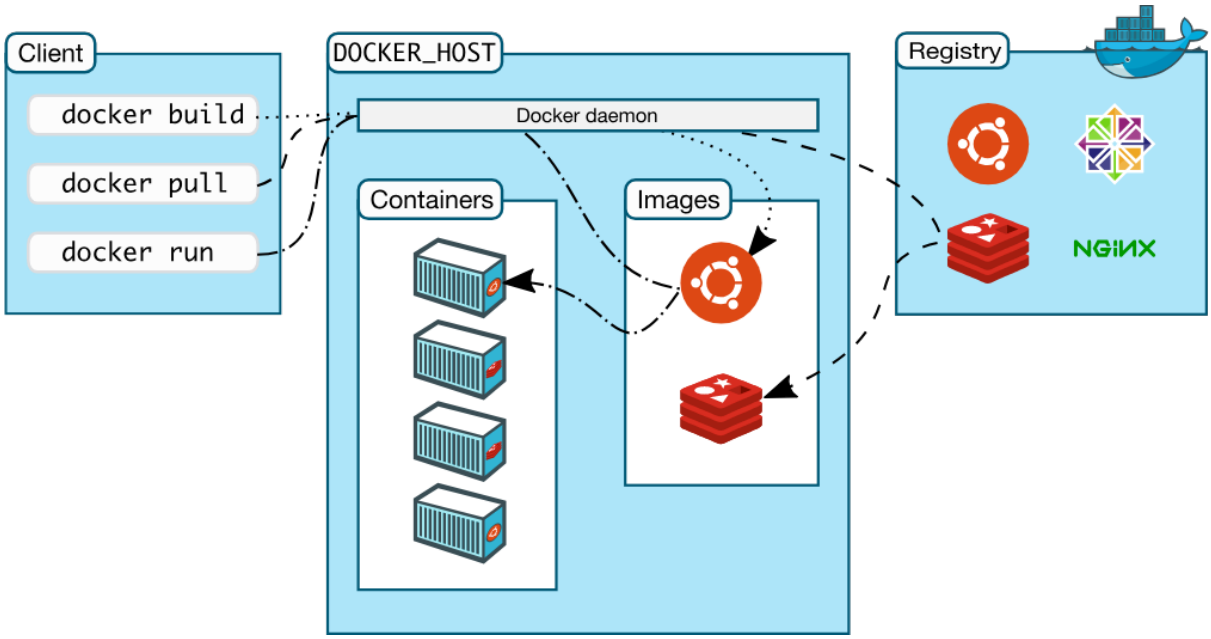


Figura 43: Arquitectura cliente-servidor a grandes rasgos de docker

4.6.1.1. Imagen

Una imagen en docker es un ejecutable el cual es empleado por el motor de docker para poner en marcha un contenedor. Haciendo un símil con la programación orientada a objetos una imagen es una clase y un contenedor es una instancia de dicha clase.

Una imagen de docker encapsula el software y todas sus dependencias para ser ejecutado con independencia al resto del software y librerías existentes en el sistema operativo. Como se aprecia en la figura 43 las imágenes de Ubuntu y Redis pueden ser empleadas por varios contenedores en ejecución.

4.6.1.2. Contenedor

Se trata de una imagen en ejecución, una “instancia” de la imagen. Los

contenedores pueden comunicarse entre ellos, principalmente por red.

4.6.1.3. Red de docker

Red virtual a la cual se pueden asignar contenedores para que mantengan comunicaciones en red entre ellos totalmente aislados de la red del sistema operativo anfitrión. Es posible crear tantas redes de docker como se deseen y además, si se estima conveniente exponer puertos al sistema operativo anfitrión.

4.6.1.4. Dockerfile

Un dockerfile es un archivo que describe, como si de un script se tratase con sintaxis propia de docker, el proceso para la creación de una imagen de docker.

Este sistema aporta una flexibilidad enorme puesto que permite generar (o en terminología de docker construir) imágenes empleando otras como base (como la de Ubuntu, Debian…). Este sistema de generación automatizada de imágenes es el que permite asegurar que la centralita asterisk esté funcionando siempre en la última versión.

4.6.1.5. Orquestación de contenedores

Una buena práctica para docker es la de mantener cada servicio en un contenedor. Puesto que en un entorno real lo más común es que hay servicios que dependen de otros servicios, si se emplea docker se hace necesario el tener que lanzar varios contenedores para constituir el servicio final como cooperación y comunicación de servicios interdependientes.

Llevar esta tarea a mano es tedioso puesto que hay que lanzar y administrar cada contenedor de manera individual, es por ello que existen herramientas de orquestación como docker Compose, docker Swarm o Kubernetes que permiten la gestión múltiple de varios servicios en contenedores, permitiendo incluso la redundancia y facilitando la escalabilidad de servicios mediante el uso de réplicas (varios contenedores ejecutando la misma imagen del servicio).

4.6.2. Imagen de Docker de la centralita

Para facilitar el despliegue y la configuración de la centralita, y con objeto de contar siempre con la versión más actualizada de asterisk, se escribió un Dockerfile el cual permite generar una imagen del PBX perfectamente válida para su ejecución en un contenedor. El dockerfile es un archivo que describe los pasos a seguir para generar una imagen que sea ejecutable en un contenedor. Se adjunta a este documento el dockerfile y todo lo necesario para generar una imagen de docker de la centralita. La última versión de estos ficheros está disponible en el repositorio https://gitlab.com/resiajf/services/asterisk.

El Dockerfile escrito, el cual se adjunta a este documento junto a los archivos de los que depende, se encarga de:

- Descargar la última versión estable de asterisk y compilarla e instalarla.
- Instalar el driver de canal PJSIP y aplicar una serie de configuraciones básicas.
- Instalar el códec OPUS y configurarlo.
- Configurar asterisk para trabajar con una base de datos, almacenando allí las configuraciones de los distintos módulos que emplea el PBX.
- Cargar el dialplan en asterisk al mismo tiempo que otras configuraciones.

4.6.3. Despliegue

El despliegue de la centralita está constituido por la propia centralita que se ejecuta en un contenedor, y el servidor de base de datos del que depende, que se encuentra en otro. En [54] pueden consultarse con exactitud todos los detalles del despliegue de la centralita y sus dependencias empleando virtualización mediante contenedores.

4.7. El dialplan

El dialplan es, por así llamarlo, el “corazón de asterisk”, define toda la lógica de la centralita, esto es, cómo se gestionan las llamadas.

El dialplan generalmente se configura en un archivo llamado “extensions.conf”, en el cual se emplea una sintaxis característica para expresar la configuración, detallada en [44].

El archivo se encuentra dividido o estructurado en lo que se denominan “contextos”. Los contextos son algo así como regiones del archivo que se asocian a distintos usuarios del sistema. Cuando un usuario inicia una llamada (marca un extensión) entra en el contexto que se ha asociado a este.

Del mismo modo dentro del contexto se define un comportamiento para esa extensión, la cual puede ser “capturada” con expresiones regulares. En se momento el comportamiento de la llamada será el resultado de “ejecutar” una por una las líneas definidas para esa extensión, como si de un script se tratase.

Cada contexto define la lógica para gestionar un o más extensiones accesibles al dispositivo que se encuentra asociado a dicho contexto.

Así pues puede existir un contexto para usuarios que no tengan acceso a llamadas a móviles y otro para usuarios que sí lo tengan.

Con objeto de comprender mejor el dialplan, se adjunta un diagrama de flujo del funcionamiento que debe tener la centralita, el cual “implementa” el dialplan.

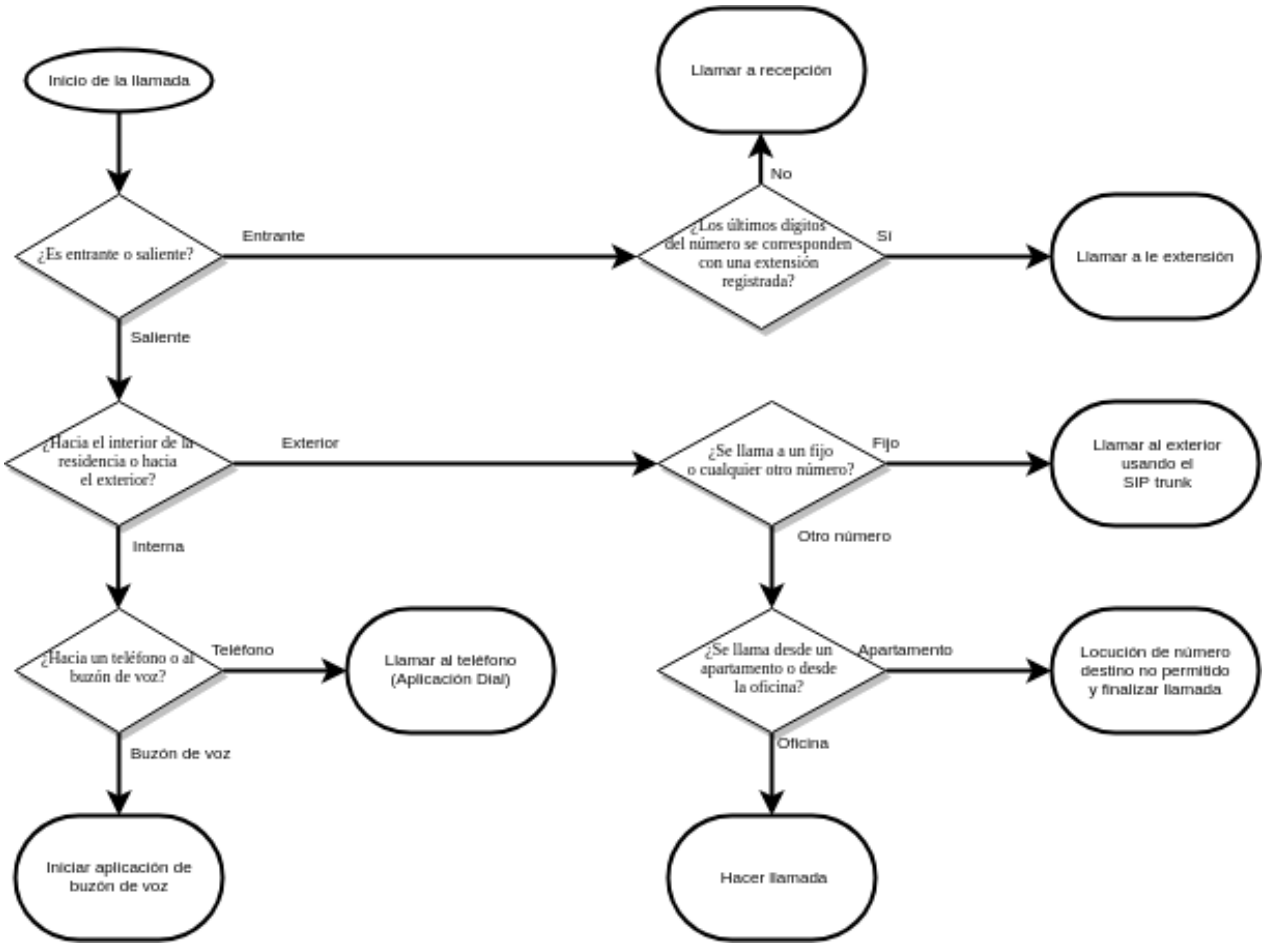


Figura 44: Diagrama de flujo de la centralita

El comportamiento de la centralita se resume en lo siguiente:

- Un residente que llama desde un apartamento sólo puede llamar a otro apartamento o recepción (llamada interna) o al buzón de voz. Además también puede llamar a un teléfono fijo.
- Si el usuario marca un número erróneo se le reproducirá una locución de número erróneo y se le colgará la llamada.
- Un teléfono desde recepción podrá llamar a cualquier fijo y móvil, además del buzón de voz y hacia dentro de la residencia. Si marca un número erróneo se reproducirá una locución indicándolo.
- Un usuario que llama desde fuera (desde la calle) tiene la opción de indicar el número de apartamento al que llama en los dos últimos dígitos del teléfono **9520892XX**, donde XX es un número que va del 00 al 90, siendo 00 un “apartamento” especial que es la recepción de la residencia. Si en los dos

últimos dígitos se indica un número que se sale del rango 00-90, se llamará por defecto a 00.

Adjunto a este documento se encuentra un dialplan de Asterisk comentado que implementa el comportamiento descrito en el diagrama de flujo. El carácter cambiante y evolutivo del sistema en una afán de constante mejora hace que el dialplan adjunto y todos los artefactos relacionados con asterisk adjuntos no se encuentren en su versión más reciente, es por ello que esta puede encontrarse en el repositorio https://gitlab.com/resiajf/services/asterisk.

```
; Definicion de algunas plantillas de las que heredan los contextos del dialplan

[buzon-voz](!) ; Extension de inicio de aplicacion buzon de voz
exten => 120, 1, Voicemail() ; Buzon de voz en la extension 120
same => n, Hangup() ; Colgar llamada al final de la aplicacion


[hacia-dentro](!) ; Llamadas hacia el interior de la residencia, se trata de un numero del 0 al
exten => _X, 1, Dial(PJSIP/${EXTEN}) ; Usar _ para indicar a asterisk que se esta usando un patr
extension marcada, almacenada en ${EXTEN}
same => n, Hangup() ; Colgar llamada

; Para extensiones de dos digitos
exten => _ZX, 1, Dial(PJSIP/${EXTEN}) ; Numero del 10 al 99
same => n, Hangup() ; COlgar

[hacia-fuera-moviles](!) ; Llamadas externas a moviles, solo disponibles desde recepcion

exten => _[67]XXXXXXXX,1,Noop(Llamada a Movil)
same => n,Dial(SIP/TRUNK/${EXTEN},20,rt)
same => n,Hangup() ; COlgar


[hacia-fuera-fijos](!) ; Llamadas externas a fijos, disponiblespara todo el mundo

exten => _[89]ZXXXXXXX,1,Noop(Llamadas a Fijo) ; Imprime en el log de la asterisk
same => n,Dial(SIP/TRUNK/${EXTEN},20,rt) ; Mandar llamada por el TRUNK SIP hacia el exterior
same => n,Hangup() ; COlgar
```

Figura 55: Fragmento documentado y comentado del dialplan

4.8. Servidor de autodespliegue y configuración para los teléfonos

Entre los requisitos del sistema se encuentran el de que los teléfonos conectados a este tengan la capacidad de obtener su configuración de forma automática y centralizada, sin requerir de la actuación manual en ese teléfono.

Un teléfono IP orientado al ámbito empresarial (como los que se han usado) goza de una característica muy interesante y es lo que se denomina autodespliegue. Esta característica consiste en que el teléfono, en cuanto es conectado a la red intenta localizar un “servidor de autodespliegue” con el cual mantiene una comunicación (normalmente por HTTP) de forma que:

- El teléfono presenta su dirección MAC.
- El servidor de autodespliegue en base a la dirección MAC suministrada por el teléfono le entrega una configuración determinada (normalmente información de registro con el PBX).

A continuación se muestra un diagrama de flujo que describe este proceso.

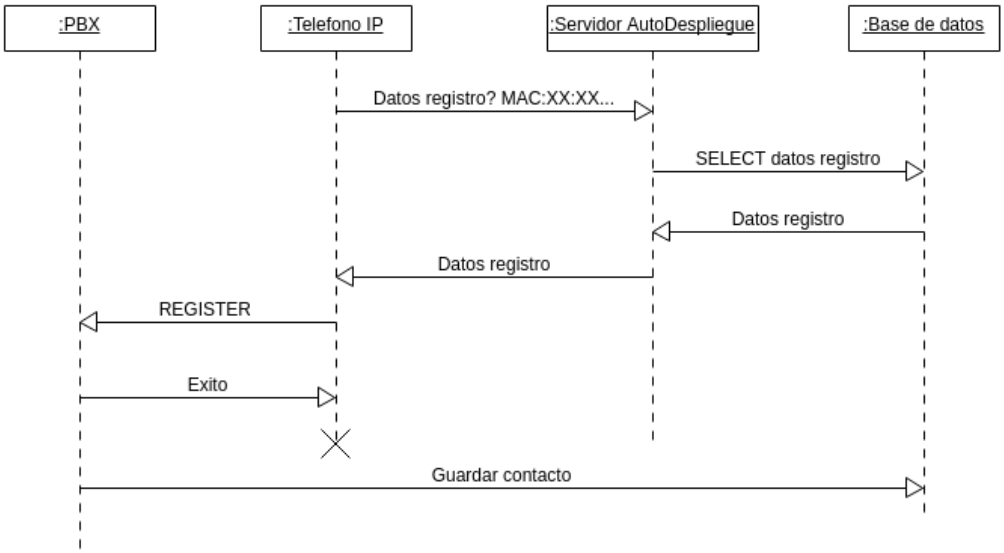


Figura 46: Proceso de autodespliegue y registro con el PBX

Se adjunta a este documento una implementación hecha en Python de un servidor de autodespliegue el cual suministra al teléfono IP sus datos de registro del PBX, obtenidos de la misma base de datos que emplea el PBX para almacenar sus configuraciones. Debido al carácter cambiante y evolutivo de la implementación y del sistema, la última versión del servidor de autodespliegue puede obtenerse en el repositorio https://gitlab.com/resiajf/services/servidor-autodespliegue.

5. Desarrollo del backend de la aplicación de gestión “Panel del Residente”

Contando la residencia con una buena infraestructura de red y un servicio de telefonía IP funcionando sobre esta, se ha estimado además conveniente desarrollar una aplicación de gestión para culminar así la mejora de las TIC de la residencia.

La aplicación de gestión en esencia tendrá un enfoque de panel de usuario en el que los residentes puedan gestionar sus relaciones con la residencia, logrando un efecto final de “área de clientes”.

El nombre escogido para este desarrollo es el de “Panel del Residente” y se ha decido emplear tecnologías web, haciendo implícito en esta aplicación el exponer una arquitectura cliente servidor.

El desarrollo ha pasado por varias fases las cuales se van describiendo en los siguientes puntos. Cada fase ha permitido obtener conclusiones e información valiosa para proseguir a la siguiente fase, aunque de dicha fase se han extraído también conclusiones para mejorar la anterior.

Esto es así debido al enfoque ágil que se ha aplicado al desarrollo, en el que todas las fases están en constante cambio y revisión durante el desarrollo de la aplicación. Un enfoque que difiere mucho del tradicional enfoque en cascada en el que una fase prosigue a la otra sin que resulte fácil retroceder o rectificar detalles en fases anteriores.

Es por ello que el enfoque ágil aplicado, en el que se potencia además la comunicación entre los miembros del equipo, hace que una rectificación en cualquier fase del desarrollo sea fácil de hacer y tenga un impacto mínimo en el desarrollo general del proyecto.

5.1. Requisitos

Esta primera fase ha sido la que ha dado comienzo al desarrollo de la aplicación. Tras mantener conversaciones con los residentes y el personal de administración en

la residencia, se han extraído una serie de necesidades las cuales esta aplicación va a permitir suplir.

Se han definido y obtenido una serie de requisitos siguiendo el modelo de user stories que debe cumplir la aplicación final. Puesto que la aplicación está separada en la parte cliente (frontend) y la parte servidor (backend) cada requisito se implementará de una forma específica en una parte dependiendo de la que se trate. La conjunción de las implementaciones en ambas partes es la que permitirá cumplir el requisito en cuestión.

5.1.1. Requisitos funcionales

Los requisitos funcionales definidos son los siguientes, siendo estos clasificados en una serie de módulos en los que se encontrará estructurada la aplicación:

En lo referido a permisos y roles en la aplicación:

- En la aplicación se distinguen varios roles, cada uno de ellos determinará las acciones que el usuario puede llevar a cabo en esta. Los roles de usuario serán:
    - Administrador
    - Residente
    - Personal de la UMA no residente
    - Personal de administración
- Los permisos se estructuran por páginas y módulos: De forma que por ejemplo un usuario tiene un rol X, el cual tiene un permiso concreto sobre la página A del módulo B.
- La visibilidad de los módulos y de los distintos apartados dentro de un módulo estarán supeditados a los permisos que posea el rol asignado al usuario.

En lo referido a la consulta del consumo de internet:

- La aplicación permitirá ver el consumo de internet de cada uno de los dispositivos ligados a un residente. La aplicación mediante los mecanismos oportunos debe registrar automáticamente los dispositivos que emplea el usuario cuando accede a la aplicación, asociándolos a este.
- Los administradores pueden ver todos los dispositivos asociados, para ver

información sobre ellos o realizar acciones sobre ellos.

- Si un usuario reclama que su dispositivo, aún siendo suyo, no lo encuentra en la aplicación, podrá notificarlo a los becarios de informática para proceder a realizar el cambio de “dueño”.
- Los consumos de internet se podrán ver en una serie de rangos, que van desde un día, hasta una semana o un mes.

En lo referido al módulo de consulta de noticias:

- La aplicación debe poseer un módulo que muestre un resumen de las noticias publicadas en el blog ya existente de las noticias. Esto le permitirá al usuario ver el blog de un vistazo sin salir de la aplicación.

En lo referido al módulo de consulta del consumo energético:

- Mostrar el consumo energético en tiempo real del apartamento del residente y ofrecer resúmenes útiles sobre este (en forma de gráfico por ejemplo).
- Un administrador puede ver los datos de consumo energético de todos los residentes.
- Del mismo modo que con el consumo del internet, los consumos eléctricos se podrán consultar en una serie de rangos que van desde un día, semana o mes.

En lo referido al módulo de consulta del historial de llamadas:

- El usuario tendrá la capacidad de ver el historial de llamadas de su apartamento.
- El personal de administración podrá ver el historial de llamadas de cualquier apartamento.
- El historial de llamadas podrá filtrarse en base a unos parámetros de filtrado útiles, como número de teléfono de/desde el que se hace la llamada o fecha y hora.

En lo referente al módulo de incidencias informáticas:

- La notificación y gestión de incidencias informáticas debe hacerse a través de la aplicación. Existirá un formulario el cual pueda rellenar un usuario del sistema para notificar sobre una avería o problema o simplemente hacer una consulta informática.

- Los administradores podrán ir actualizando el estado de una incidencia informática, quedando todo esto registrado en un seguimiento asociado el cual podrá ser consultado por el usuario que ha enviado la incidencia.
- En un momento dado el usuario puede descartar una incidencia que tenga abierta, pudiendo detallar el motivo por el que lo ha hecho (el problema ya ha desparecido por ejemplo).
- Los administradores podrán ver las incidencias activas y resueltas.
- Una incidencia tiene varios estados por los que pasa:
    - Pendiente: Nueva incidencia, esperando a que algún becario la vea y atienda.
    - Esperando: Incidencia lista para ser arreglada, pero por motivos de compatibilidad horario no se puede atender aún.
    - Trabajando: Hay un becario trabajando en ella.
    - Esperando a usuario: Se está esperando una respuesta de usuario por algún motivo.
    - Resuelto: La incidencia se ha resuelto.
    - Descartado: El usuario ha descartado la incidencia.
    - No se arreglará: por algún motivo razonable la incidencia no será arreglada.

Referido a las notificaciones y al módulo de notificaciones:

- La aplicación enviará notificaciones a los usuarios ante distintos eventos que ocurran en el sistema. Esta contará con distintos métodos de notificación:
    - **Correo institucional de la UMA**, método de notificación por defecto.
    - **Bot de telegram**, el bot mandará a los usuarios con los que mantiene una conversación notificaciones oportunas.
    - **Notificaciones push**.
- Cada usuario puede seleccionar por qué medios desea recibir las notificaciones, pudiendo activarlos y desactivarlos. Las notificaciones instantáneas dentro de la aplicación siempre estarán activadas.
- Dentro de la aplicación existirá un centro de notificaciones desde donde se pueden consultar todas las notificaciones (pendiente, vistas…).

En lo referente a la gestión de la sesión y la autenticación en la aplicación:

- Cualquier usuario de la aplicación excepto el personal de administración puede debe identificarse en la aplicación usando sus credenciales de iDUMA. El personal de administración que no se encuentre en el DUMA tendrá sus propias credenciales, estando inscrito en un directorio propio interno a la residencia.

5.1.2. Requisitos técnicos

Del mismo modo, se han definido una serie de requisitos técnicos que debe cumplir la implementación. Dichos requisitos son bastante más concretos:

- La aplicación debe ser modular, permitiendo su extensión fácilmente.
- La aplicación debe poder usarse desde un navegador web en cualquier dispositivo (será por tanto responsive).
- La aplicación se encuentra perfectamente dividida en dos partes (un cliente y un servidor, siguiendo la arquitectura cliente servidor), y se debe garantizar la perfecta interoperabilidad entre ambas mediante el empleo del estilo arquitectónico REST y usando el formato de intercambio de datos JSON. En concreto la aplicación la componen:
    - La parte visual, que comprende la interfaz de usuario y su lógica de control es la parte cliente, denominada frontend.
    - La parte servidor, que comprenden la lógica de negocio y expone la interfaz (RESTFul API) que consumirá el cliente, denominada backend.
- Una notificación ya consultada se borrará mediante una tarea programada al cabo de un tiempo prudente (unos dos o tres meses).
- La sesión establecida en la aplicación será mandada por el cliente al servidor en cada petición en forma de token JWT. La validez del primer token JWT obtenido mediante login con credenciales será de 2 días.
- En una respuesta HTTP del servidor se mandará en la cabecera un campo de extensión cuando el token esté a punto de caducar, sugiriendo al cliente que debería renovarlo. La cabecera se denominará *X-ShouldRenewToken* y tendrá un valor establecido a 1.
- Un token JWT obtenido mediante un login con credenciales tendrá una validez de dos días, y se avisará con una hora de antelación a su caducidad de que debe renovarlo.

- Un token JWT obtenido mediante renovación tendrá una validez de dos horas, y se avisará al usuario de su caducidad con una antelación previa de 30 minutos.
- El cliente podrá renovar el token, obteniendo uno nuevo renovado con una validez menor, en una ruta habilitada al defecto.
- El mecanismo de persistencia del token JWT en el cliente será proporcionado por la API de localstorage/sessionstorage del navegador web.
    - Localstorage permite mantener la sesión incluso si cierra la pestaña o el navegador. Es el método de la API que permite implementar el mecanismo de recuérdame.
    - Sessionstorage mantiene la sesión sólo en la pestaña actual del navegador.
- Si un usuario realiza una petición al servidor para realizar una acción, pero el usuario que la realiza no tiene permisos, debe fallar la petición indicando que no tiene permisos para realizar dicha acción.
- En el frontend, si se intenta acceder a una página para la que no se tienen permisos, aparecerá que la página no existe.

5.1.3. Casos de uso de alto nivel

En base a los requisitos suministrados, se ha elaborado un diagrama UML de casos de uso, el cual se encuentra a continuación. En el diagrama se aprecian con facilidad los distintos actores/roles involucrados en en el sistema.

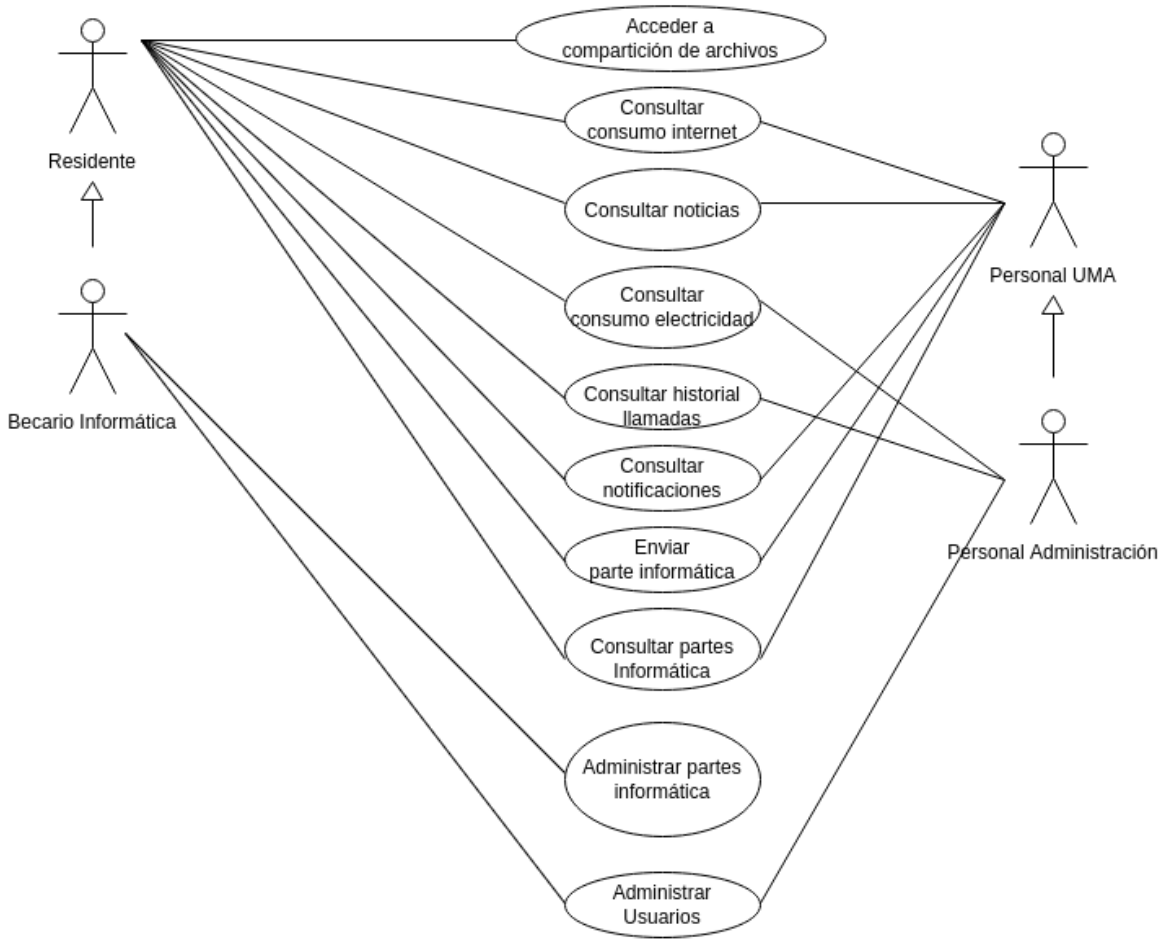


Figura 47: Diagrama de casos de uso

5.2. Diseño

Tras analizar, valorar y comprender todos los requisitos, se ha elaborado un diseño del backend el cual proporciona toda la información necesaria y requerida para llevar a cabo su implementación.

Como se ha comentado la arquitectura que tendrá la aplicación será cliente-servidor y la comunicación entre ambos se llevará a cabo mediante una RESTFul API, haciendo peticiones y recibiendo respuestas empleando el protocolo HTTP a/desde una serie de endpoints disponibles en el backend.

5.2.1. Diagrama de clases

En primer lugar se ha modelado la estructura del sistema estableciendo la entidades existentes existentes en el sistema y las relaciones existentes entre estas. Este modelado estructural se ha concretado en el siguiente diagrama de clases:

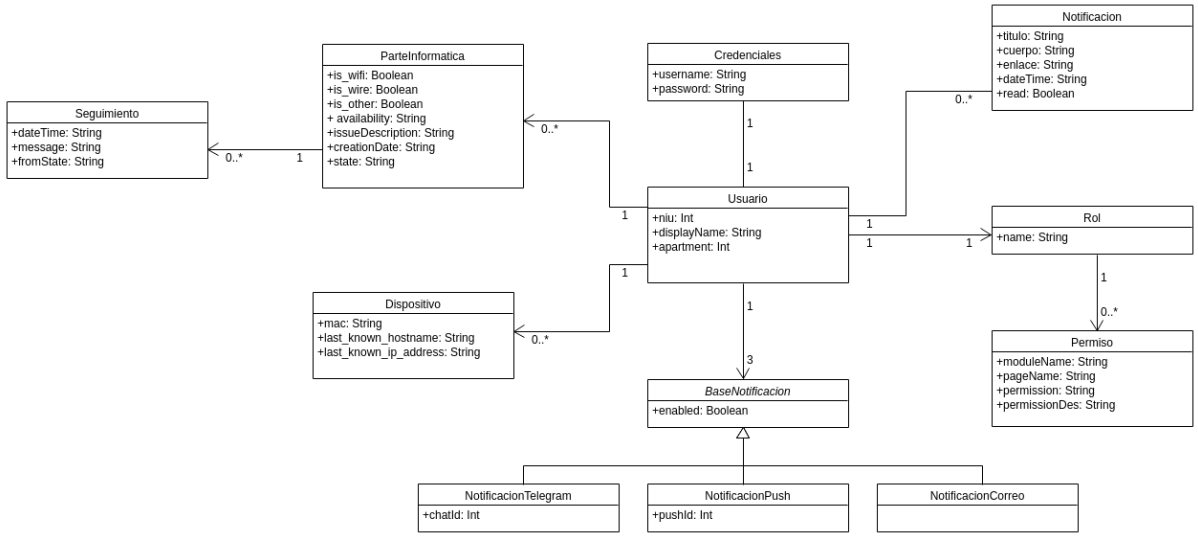


Figura 48: Diagrama UML de clases del sistema

Como se puede observar, en esta fase de diseño se han asociado además una serie de atributos de utilidad a las distintas entidades. A continuación se describen las entidades existentes en el sistema y cómo se relacionan con el resto:

Usuario:

Se trata de un usuario que usa el sistema, entre sus atributos notables se encuentran el niu (Número de identificación universitaria), del que opcionalmente podrá disponer, y el número de apartamento en el que habita, también opcional.

Puesto que todo el sistema gira en torno al usuario, este tendrá bastante relación con el resto de las entidades, tal como se describe en la imágen.

En el diagrama dichas relacionales se encuentran debidamente descritas, aunque cabe destacar la relación entre usuario y BaseNotificacion. En la

descripción de esta entidad detalla la relación.

Un usuario posee un rol el cual tiene una serie de permisos asociados.

BaseNotificacion:
En los requisitos se establecía que un usuario puede configurar diferentes vías de notificación, pudiendo habilitarlas y deshabilitarlas. Esta clase abstracta sirve como base para definir los diferentes tipos de notificaciones que existen en el sistema.

El que el usuario se relacione con 3 ejemplares que hereden de esta clase se debe a que existen tres canales de notificación.

Es por ello que el usuario estará relacionado con una configuración por cada canal de notificación existente (actualmente tres). La configuración principal de un canal de notificación es el de poder habilitarlo y deshabilitarlo.

Notificacion:
Se trata de una notificación del sistema la cual el usuario puede recibir por los distintos canales de configuración existentes. Un usuario puede estar relacionado con varias de estas como se muestra en el diagrama.

La notificación cuenta con una serie de atributos de utilidad como pueden ser título, cuerpo, estado de leída o no, fecha y hora y enlace. El atributo opcional incluye incluye un vínculo el cual el usuario podrá visitar para obtener más información acerca de la notificación.

Permiso:
Se trata de un permiso de la forma en la que se ha descrito en los requisitos. Esta clase describe un permiso que se tiene en una página concreta de un módulo determinado.

Rol:
Describe un rol del sistema el cual tiene una serie de permisos asociados.

Credenciales:
En los requisitos se comenta el caso de que un usuario no esté en el DUMA, careciendo por tanto de credenciales institucionales. Es por ello que esta entidad permite asociar unas credenciales complementarias (nombre de usuario y contraseña) para poder autenticarse en el sistema a un usuario que las requiera por no estar en el DUMA.

ParteInformatica:
Modela un parte de incidencias informáticas, y presenta una serie de atributos útiles que le dan utilidad a este para resolver la incidencia.

Cabe destacar el atributo availability, que describe la disponibilidad del usuario para poder atender su incidencia. Availability tendrá un valor establecido igual a un diccionario de JSON definido de forma determinado.

Seguimiento:
Modela un ítem en el seguimiento de un parte de informática, un parte de informática está asociado a varios ítems de seguimiento.

Cabe destacar el atributo fromState, que indica el estadio que tenía el parte en el momento de actualizar su seguimiento (lo que puede haber hecho que el parte cambie de estado).

Dispositivo:
En los requisitos se establece que un usuario puede tener varios dispositivos asociados para poder así consultar su consumo de internet.

Esta entidad modela un dispositivo, pudiendo un usuario tener asociados varios de estos.

Un dispositivo posee una serie de atributos interesantes que permiten identificarlo en la red, como pueden ser: Dirección IP, dirección MAC y nombre de host.

5.2.2. Diagrama Entidad Relación

A partir del diagrama de clases se ha elaborado, casi de forma directa (por la estrecha relación entre una clase y una entidad) un diagrama entidad relación que modela el cómo se van a persistir y relacionar las entidades en una base de datos relacional. El diagrama se encuentra en la figura 49.

Como se puede observar, este diagrama guarda una estrecha relación con el de clases puesto que ha sido derivado del anterior, aportando además una información esencial para poder persistir las entidades y sus relaciones en la base de datos.

En este nivel se han aportado, con respecto al diagrama de clases, una serie de detalles útiles para la persistencia como son:

- **Claves primarias** para identificar unívocamente a una entidad (aparecen en el diagrama dibujadas con una llave al lado del atributo de la entidad).
- **Opcionalidad en determinados atributos** (aparece representado como un rombo sin colorear al lado del nombre del atributo).
- **Claves foráneas**: Aparecen representadas con un rombo rojo, si además son clave primaria con un punto debajo.
- Entidad adicional “rol_has_permiso” que permite **persistir una relación de muchos a muchos** entre roles y permisos (un permiso lo pueden tener varios roles, y un rol puede tener varios permisos).

5.2.3. Diagrama de componentes

De cara a aportar un mayor detalle a la hora de hacer la implementación, se ha elaborado un diagrama UML de componentes en el que se detallan los distintos componentes que conforman el sistema, del mismo modo que las interfaces que estos ofrecen y requieren.

A la hora de hacer este diagrama se ha presupuesto (siempre condicionando lo menos posible) el empleo de ciertas tecnologías de cara a la implementación, como son:

- Empleo del **framework Connexion**, hecho en Python [46] que ofrece capacidad de validación de parámetros y enrutamiento para una RESTFul

API.

- Empleo de **Websockets** para las notificaciones en tiempo real dentro de la aplicación.
- Empleo de un **gestor de caché** que se apoya en un almacenamiento no persistente, y NoSQL (no se trata de una base de datos relacional).
- Uso de un **ORM**: Se trata de una característica que puede proporcionar algún framework (como el famoso SQLAlchemy [47] para python) para hacer un mapeo objeto relacional en la aplicación. Esta forma de trabajo permite abstraerse del motor de persistencia subyacente a la aplicación y centrarse en el desarrollo al trabajar con elementos del modelo definidos empleando los medios que ofrece el lenguaje de programación que se está usando.

Como se puede observar el backend interactúa con una cantidad considerable de servicios del exterior, lo que hace que requiera emplear una gran cantidad de componentes (ya sea tanto internos como externos).

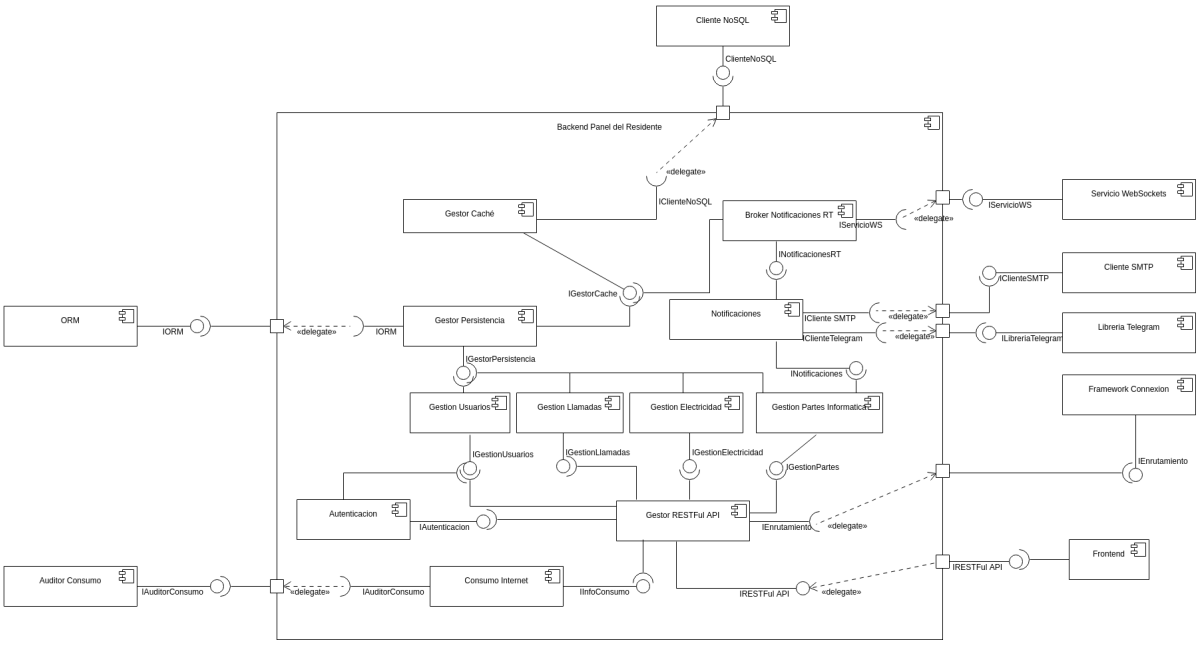


Figura 50: Diagrama de despliegue de la aplicación

A continuación se proporciona una descripción de los componentes observados en el diagrama de componentes del sistema, detallando de qué función tienen estos y las interfaces que ofrecen y requieren.

Backend Panel del Residente

Se trata del componente principal que encapsula a todos los componentes que conforman el backend del panel del residente, implementando cada uno de estos partes concretas de la lógica de negocio del backend.

Este componentes requiere, a través de puertos una serie de componentes del exterior (dependencia) para poder garantizar su correcto funcionamiento y el de los componentes internos que las requieren.

Por su parte este mismo componente expone la interfaz IRESTFUL API, se trata de una interfaz RESTFul (se adhiere a las restricciones del estilo arquitectónico REST y funciona sobre HTTP, haciendo un uso adecuado de sus verbos). Esta interfaz es consumida y requerida por el cliente (la parte frontend de la aplicación), cuyos detalles de diseño e implementación entre otros se detallan en [54].

Auditor Consumo

Componente externo el cual se encarga de recopilar información sobre el consumo del internet en la residencia de forma individualizada por cada dispositivo, mediante una captura de paquetes constante y almacenamiento del tamaño de estos (descartando su payload).

ORM

Componente externo el cual permite llevar a cabo el mapeo objeto relacional, de forma que todos los detalles de las persistencia se abstraen y esta es llevada a cabo mediante el empleo de una serie de clases que se han definido las cuales representan las distintas entidades del modelo.

Cliente NoSQL

Componente externo el cual es un cliente que permite conectarse al almacenamiento volátil empleado para las cachés.

Servicio de WebSockets

Componente externo el cual es un servidor que ofrece una implementación de Websockets, una tecnología que permite establecer un canal TCP de

comunicación bidireccional en tiempo real entre el frontend y el backend. Este componente es empleado para ofrecer el sistema de notificaciones en tiempo real dentro de la apliación.

Cliente SMTP

Componente externo el cual implementa un cliente SMTP para poder hacer envío de correos electrónicos de notificaciones a los usuarios del sistema.

Liberia de Telegram

Componente externo el cual permite mandar mensajes a usuarios que desean ser notificados mediante el canal de notificaciones de Telegram, empleando la API de telegram.

Framework Connexion

Componente externo el cual se trata del Framework Connexion, se trata de un Framework implementado en Pyhon el cual proporciona ya implementado una buena parte del boilerplate (implementaciones repetitivas de proyecto en proyecto) necesario para montar una RESTFul API, proporcionando validación de parámetros, enrutamiendo, manejadores por rutas, gestión de autorización y de autenticación y gestión de errores y excepciones, además emplea como entrada un modelo el cual sigue una especificación ampliamente apoyada, como se detallará más adelante.

Gestor RESTFul API

Componente interno el cual define una serie de manejadores para las rutas, y que por tanto necesita de la interfaz de enrutamiento, y que se encarga de llamar a los componentes necesarios dentro del handler que atiende la petición a una ruta concreta del servidor.

Consumo Internet

Componente interno el cual proporciona información de consumo (extraída del componente auditor de consumo) procesada y válida para ser mandada al frontend.

Autenticación

Componente interno el cual permite autenticar a un usuario, para ello se ampara en el módulo de gestión de usuarios.

Gestion Usuarios

Componente interno el cual se encarga de todo lo relacionado con la gestión de usuarios de la aplicación (alta, baja, modificación). Requiere del gestor de persistencia, dado que el usuario es una entidad que se encuentra persistida.

Gestion de Llamadas

Componente interno el cual se encarga de obtener del almacenamiento persistente información sobre llamadas llevadas a cabo por los teléfonos IP de la residencia, ofreciendo además capacidades de filtrado.

Gestion de electricidad

Componente interno el cual se encarga de obtener información sobre el consumo eléctrico de los apartamentos, requiere del acceso a la persistencia.

Gestion Partes Informatica

Componente interno el cual se encarga de la gestión íntegra de los partes de informática (actualización del seguimiento, creación, modificación) en el sistema. Requiere de la interfaz de persistencia y además de la de notificación, esto es así puesto que todas las actualizaciones por las que pasa el parte de informática le son notificadas al usuario.

Gestor Persistencia

Componente interno el cual abstrae, empleando el ORM el proceso de persistencia. Define y ofrece una serie de clases de modelo y métodos útiles para interactuar indirectamente con el ORM, que a su vez lo hace con el motor de persistencia relacional.

A su vez también permite abstraer el empleo de la caché de la aplicación, requiriendo de la interfaz del gestor de caché.

Notificaciones

Componente interno el cual se encarga de entregar a los usuarios por las distintas vías las notificaciones que necesiten mandar el resto de componentes.

Es por ello que requiere de interfaces que le habiliten para el envío de notificaciones en tiempo real, mediante correo y por Telegram.

Broker Notificaciones RT

Componente interno el cual es un broker de notificaciones en tiempo real mediante websockets. Requiere de la interfaz del gestor de caché para mantener una suscripción en un canal de notificaciones internos a la aplicación que le permita atender aquellas notificaciones internas relacionadas con el envío de notificaciones a un usuario.

Para enviar una notificación a un usuario por Websockets el componente requiere de un servidor de websockets el cual mantiene conexiones abiertas con los distintos clientes que se encuentran escuchando a la espera de notificaciones.

Gestor Caché

Componente interno el cual permite interactuar con el almacenamiento volátil no SQL, permitiendo emplear características que ofrece este como la suscripción y publicación en canales.

5.2.4. Autenticación y autorización en la aplicación

A poco que una aplicación sea mínimamente compleja, requiere de características de autenticación y autorización, en primer lugar, se definen estos conceptos como:

- **Autenticación**: El hecho de poder identificar a un usuario de la aplicación.
- **Autorización**: Estando identificado el usuario, permitirle realizar ciertas acciones en base a los permisos de los que este disponga en el sistema.

Una decisión de diseño que se ha tomado en la aplicación para gestionar la autenticación y autorización de usuarios es el empleo de tokens JWT.

JWT es es un método abierto y estándar (RFC 7519) para representar de forma segura un acuerdo entre ambas partes [49]. La especificación define cómo decodificar, verificar y generar JWT.

De esta forma cuando un usuario hace un login exitoso, el servidor le proporciona un token JWT cifrado, el cual el cliente debe conservar y presentar al servidor en todas las peticiones que haga desde ese momento en adelante, permitiendo este mecanismo lograr los objetivos de autenticación (al presentar un token válido se sabe que pertenece a un usuario del sistema) y autorización (el token permite guardar en su payload información útil que puede emplear el backend al recibir y decodificar el token, como id de usuario y permisos que tiene).

El token JWT es en esencia una cadena de texto de aspecto similar a:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c

A continuación se muestra un diagrama de flujo de un login exitoso empleando usuario y contraseña en el que el usuario tras presentar unas credenciales correctas obtiene un token JWT, el cual presentará en las sucesivas peticiones para tener acceso a los recursos.

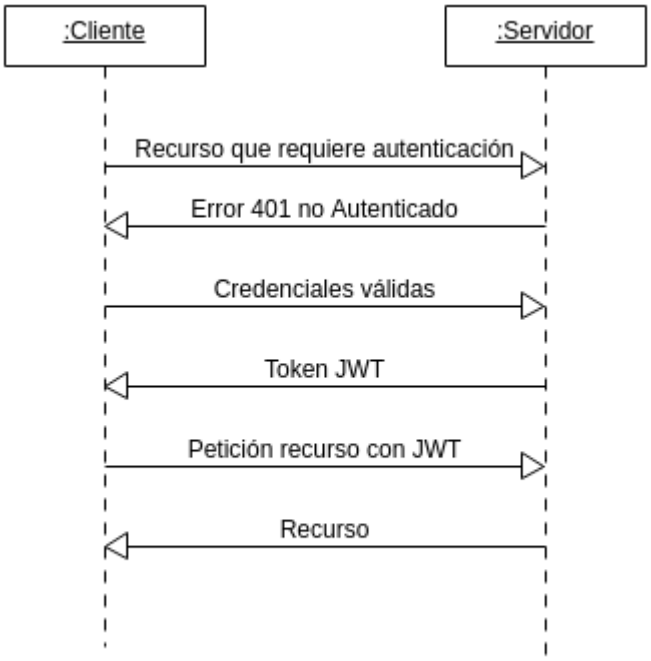


Figura 51: Flujo de solicitud y uso de token JWT

5.2.5. Autenticación en la aplicación mediante iDUMA

Uno de los requisitos de la aplicación es el de permitir a los usuarios identificarse en la aplicación empleando sus credenciales del DUMA, mediante iDUMA.

De esta forma es la UMA la que actúa como IdP (proveedor de identidad), confiando la aplicación del Panel del Residente en que las autenticaciones que hace el IdP de la UMA son fiables.

Para lograr esto se ha registrado la aplicación del “Panel del residente” como un proveedor de servicio en el IdP de la UMA, permitiendo así que esta pueda autenticar sus usuarios mediante el IdP de la UMA empleando SAML 2.0 [50]. SAML 2.0, al igual que JWT permite también autenticar y autorizar al usuario, aunque goza de mayor complejidad que JWT.

Es por ello que en realidad lo que se ha hecho es registrar como proveedor de servicio una aplicación proxy que tiene un único recurso en una URL determinada. Al solicitar el usuario el recurso sin estar autenticado, comenzará el flujo SAML 2.0 tal como se describe en la figura a continuación, concluyendo este en la obtención

del recurso.

El recurso protegido en el proxy obtenido por el usuario no es más que una redirección a una URL del backend con unos datos específicos proporcionados por el IdP de la UMA en el querystring de la URL que le permitirá al backend identificar al usuario (empleando datos como el NIU) y así generar y proporcionarle un token JWT mediante otra redirección a una URL del frontend en la que se adjunta el token JWT generado en el querystring de esta para ser almacenado por el frontend.

Como punto adicional los datos sobre el usuario de la UMA proporcionados por el IdP (los mínimos posibles, como son el NIU y el nombre del usuario) son guardados por el proxy en un almacenamiento no persistente de redis en el servidor, de forma que lo que el proxy realmente le suministra al backend en la redirección en el querystring de la URL no son los datos proporcionados por el IdP sino una clave para que el backend pueda obtenerlos del almacenamiento volátil.

A continuación se muestra un diagrama de secuencia en el que se describe este proceso.

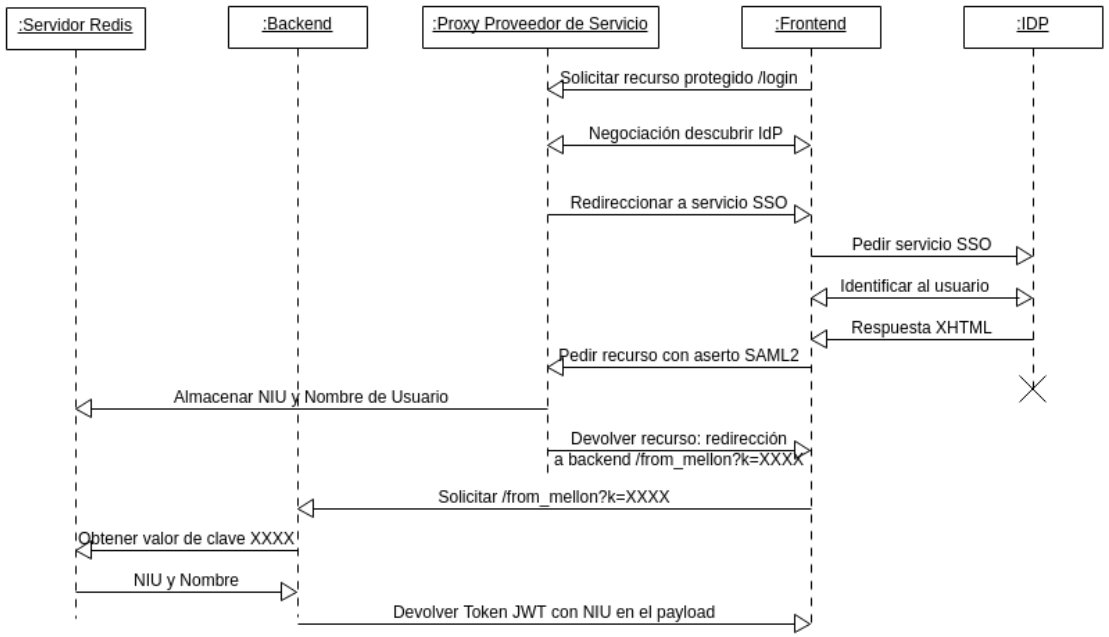


Figura 52: Autenticación empleando sistema híbrido de SAML 2.0 y JWT

El flujo correspondiente exclusivamente a SAML 2.0 se encuentra en el diagrama hasta el momento en el que el cliente hace la petición al Proxy empleando el aserto SAML 2.0 . Los detalles específicos del flujo de SAML 2.0 se pueden consultar en [50].

A partir de ese momento el Proxy tiene identificado al usuario y conoce su NIU y nombre de usuario (proporcionados por la UMA). Acto seguido los guarda en el almacenamiento no persistente y obtiene la clave con la que se han guardado estos datos.

Inmediatamente después se manda al frontend una redirección con la URL del backend /from/mellon?k=XXXX, donde XXXX es la clave de redis. El frontend hace una petición a esta URL del backend y en ese momento el backend, empleando la clave de redis del querystring de la URL, obtiene el NIU y nombre de usuario, teniendo así identificado al usuario y estando en condición de devolverle un token JWT el cual podrá usar el frontend como se describe en la figura 51.

5.2.6. Renovación del token JWT

En los requisitos se hace referencia a un mecanismo el cual permite al cliente renovar el token JWT cuando este esté próximo a su caducidad.

Para ello el cliente hará una petición a un endpoint determinado proporcionando el token JWT y recibiendo uno nuevo, quedando el antiguo token JWT anulado al ser colocado en una lista negra.

A continuación se muestra un diagrama de secuencia en el que se describe con éxito una renovación de token.

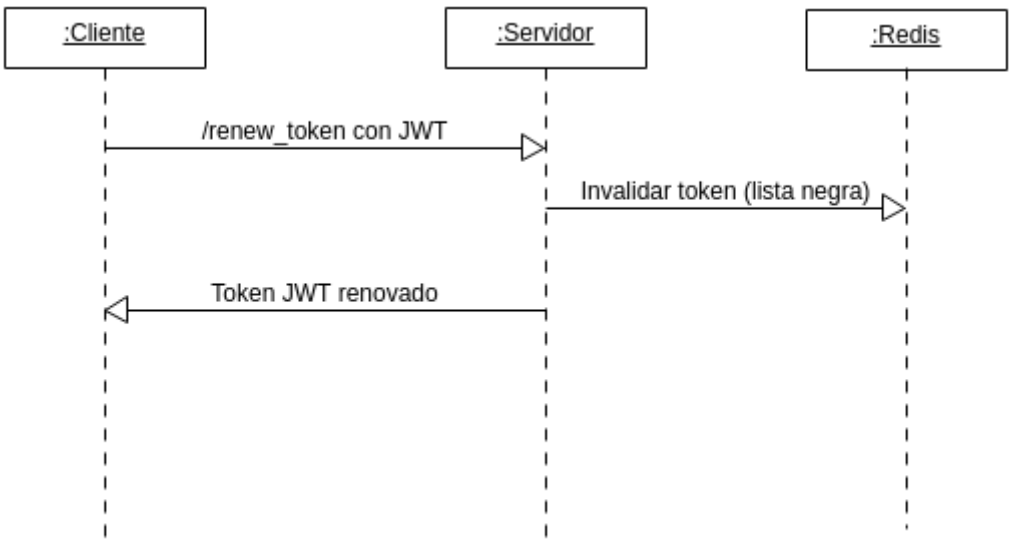


Figura 53: Renovación de un token JWT

5.2.7. Sistema de notificaciones en tiempo real

Para mandar notificaciones en tiempo real al navegador de los usuarios se ha empleado un sistema basado en Websockets el cual cuenta como componentes clave (además de un servidor de Websockets) con el servidor de base de datos Redis, el cual implementa un sistema de persistencia en RAM y/o disco duro [48].

Redis tiene un característica interesante y es la de los canales, en los cuales se pueden publicar mensajes que le llegarán al instante a todas las entidades que se encuentren suscritos a estos.

El sistema que se ha ideado para mandar notificaciones en tiempo real es el siguiente, y cuenta con una serie de componentes:

- **Servidor Redis**: Se encarga de mandar a todas las entidades que se encuentren **suscritas** a un canal los mensajes que se van publicando en este.
- **Servidor de websockets**: Se encarga de mantener conexiones TCP activas bidireccionales con los distintos clientes empleando el protocolo de WebSockets.
- **Broker de notificaciones**: Componente de la aplicación que se encarga de escuchar los mensajes de notificaciones que se publican en el canal de redis entrega estas por websocket a los usuarios que correspondan.

Como se puede observar el empleo de este sistema tiene una serie de ventajas bastante importantes, entre ellas el desacoplamiento de entidades, puesto que por ejemplo cualquier cliente empleado en cualquier aplicación que se desarrolle en el futuro que publique en el canal adecuado estará mandando notificaciones a los usuarios de esta aplicación.

A continuación un diagrama de secuencia en el que se describe una interacción exitosa entre estas entidades.

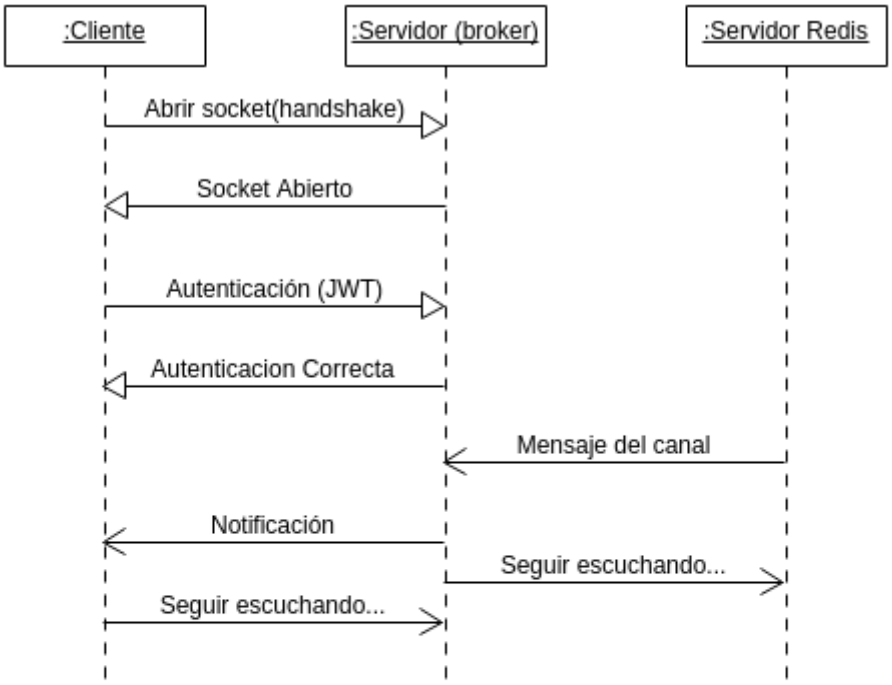


Figura 54: Diagrama de secuencia describiendo la interacción cliente-servidor mediante WebSockets

El servidor se encuentra escuchando mensajes en el canal de Redis, y, tras la autenticación del usuario, recibe por el canal una notificación dirigida a dicho usuario la cual le redirige a este.

5.2.8. Activación de notificaciones en telegram

En los requisitos se menciona la opción de recibir notificaciones de la aplicación por Telegram (aplicación de mensajería instantánea que ofrece una API para que los desarrolles usen su plataforma). Para ello se ha implementado en el backend una lógica que responde al diagrama que se muestra a continuación.

Además se ha desarrollado un “bot” de telegram que tenga capacidad para responder a los mensajes del usuario.

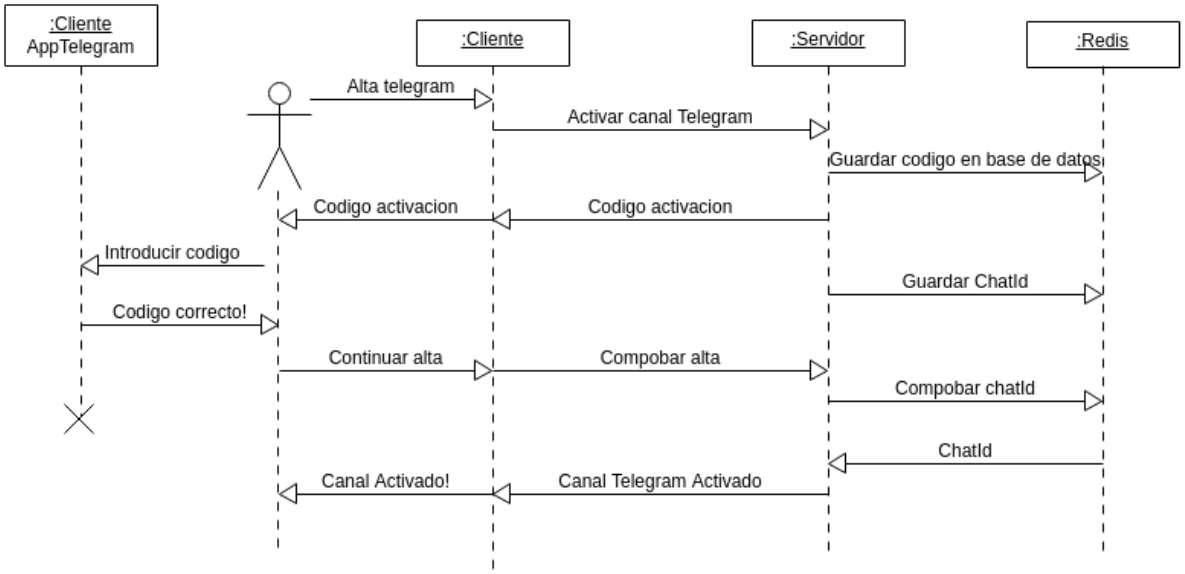


Figura 55: Alta exitosa de las notificaciones mediante Telegram

En el diagrama de observan las 5 entidades que intervienen (contando al usuario). En esencia el proceso mostrado se da en una serie de fases:

- El usuario solicita dar de alta el servicio, para lo que se le ofrece un código numérico que debe proporcionar al bot. El código numérico se guarda en el almacenamiento volátil.
- El usuario le proporciona el código al bot de telegram y este establece en el almacenamiento volátil unos parámetros que le permitan al backend mandarle mensajes al usuario. El bot le sugiere al usuario que confirme en el frontend que ha suministrado el código.
- El usuario confirma en el frontend que ha suministrado el código y a continuación el backend finaliza el proceso de activación notificando al usuario por telegram.

Cuando todo el proceso descrito se culmina se es notificado al usuario por Telegram de la correcta activación tal como se muestra en la siguiente figura:

Figura 56: Parte del proceso de activación y recepción de notificaciones variadas

5.3. Especificación de la RESTFul API

En las consideraciones de diseño anteriores se hace referencia a la presencia de una RESTFul API ofrecida por el backend y consumida por el frontend y de una serie de endpoints con los que la comunicación ocurre mediante el empleo del protocolo HTTP.

El haber optado por este enfoque se debe a que hoy día es la forma más habitual de diseñar una API para una aplicación cliente-servidor distribuida. De forma que un servicio web (ofrecido por el backend) que se adhiere a las restricciones arquitectónicas de REST, se denomina RESTFul [51].

Una API RESTFul puede ofrecerse de muchas formas, pero la más común es hacerlo basándose en el protocolo HTTP, ya que este tiene una serie de características REST inherentes.

Otro tipo de APIs existentes en servicios web son aquellas basadas en SOAP, las cuales aún cuentan con un gran uso pero se encuentran en decadencia para dar paso a REST.

Una API a su vez necesita ser definida de manera formal, para que un programa pueda “entender” los servicios que ofrece y cómo llamarlos y emplearlos. En SOAP esto se logra mediante el famoso WDSL, pero en REST hasta hace no mucho no había una manera formal de lograrlo.

Esto cambió con la aparición de Swagger 2.0/ OAS 2.0, se trata de una iniciativa apoyada por gigantes como Google y la Linux Foundation entre otros que especifica de forma detallada el cómo definir una HTTP RESTFul API mediante el empleo de un fichero YAML o JSON [52]. OAS 2.0 permite definir en detalle una API de forma similar a como se hace en un WSDL para SOAP.

Es por ello que la HTTP RESTFul API ofrecida por el backend de la aplicación “Panel del Residente” se ha definido empleando esta especificación, la cual se adjunta a este documento. Junto a esta especificación se adjunta un archivo autocontenido el cual es posible abrir en un navegador web para examinar todos los servicios que ofrece la API, cómo llamarlos (y los requisitos que hacen falta para ellos) y la respuesta que se espera de estos.

La representación visual de la especificación tiene un aspecto similar a este:

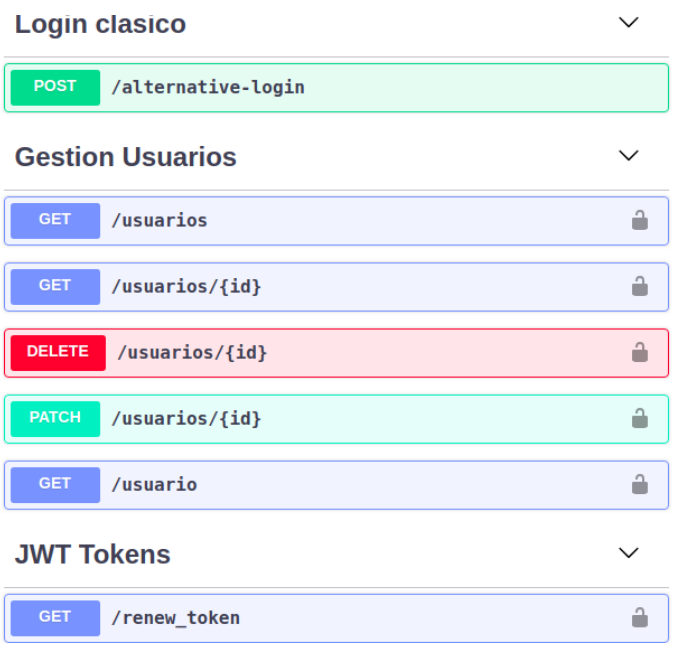


Figura 57: Representación visual de la especificación

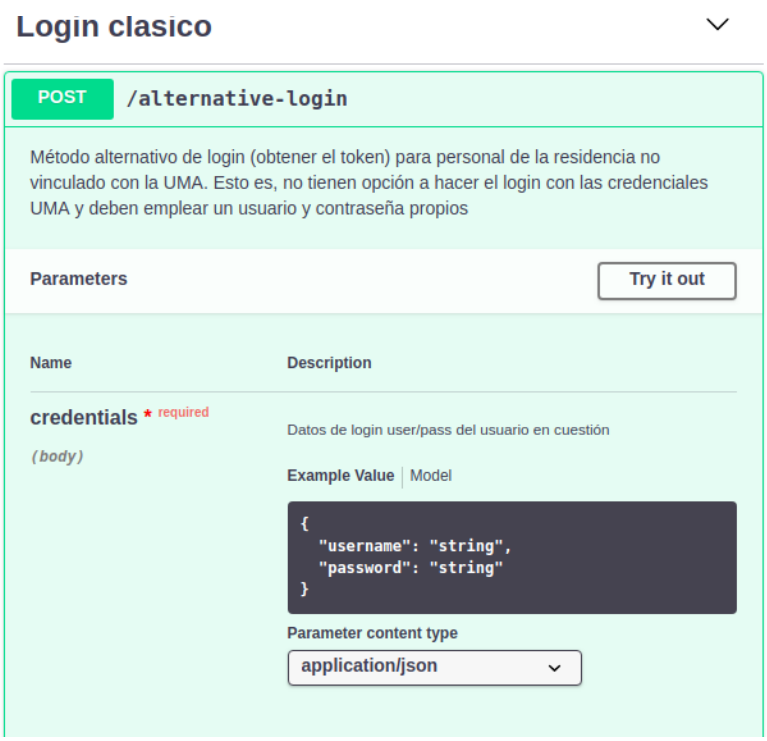


Figura 58: Representación visual de los detalles de un endpoint

Mientras que la representación textual un aspecto similar a este:

```yaml
/alternative-login:
  post:
    tags:
      - Login clasico
    description: |
      Método alternativo de login (obtener el token) para personal de la
    operationId: auth.classic_login
    parameters:
      - in: body
        name: credentials
        description: Datos de login user/pass del usuario en cuestión
        required: true
        schema:
          type: object
          required:
            - username
            - password
          properties:
            username:
              type: string
            password:
              type: string
    security: []
    responses:
      200:
        description: OK
        schema:
          type: object
          properties:
            token:
              type: string
              description: Token JWT válido para usar la API
      401:
```

Figura 59: Extracto de la especificación de la API

5.3.1. Gestión de errores en la API

Como es normal un uso erróneo de la API provocará situaciones de error, al igual que un funcionamiento anómalo de esta. Estos errores provocarán situaciones excepcionales las cuales se comunicarán al cliente devolviéndole una estructura de datos (objeto JSON) la cual cuenta con cuatro campos bien definidos junto a un código de estado HTTP. Los campos de la estructura con la descripción del error son:

- Detail: Descripción textual del error que se ha producido
- Status: Código de estado HTTP devuelto en la petición
- Title: Título del error que se ha producido
- Type: Código interno del error que se ha producido

Se adjunta a este documento una lista de códigos de errores que pueden producirse en el backend (además de los habituales definidos en la especificación por motivos de autenticación/autorización...etc.).

5.3.2. Gestión de la autorización

Tal como se observa en la especificación cada método de cada endpoint tiene unos “scopes”. Un scope indica el ámbito en el que puede usarse un método de un endpoint. De esa forma un endpoint con un scope admin podrá ser usado solo por un admin y un endpoint con un scope usuario podrá ser usado solo por un usuario.

Cuando se genera un token JWT para el usuario, se guarda en el payload de este los scopes para los que tiene acceso, de forma que por ejemplo en el token JWT del admin se guardan todos los scopes de la API, entendiéndose que este tiene acceso total a esta.

5.4. Implementación

La implementación del backend se ha llevado a cabo siguiendo las pautas de diseño definidas anteriormente y empleando el lenguaje de programación Python.

Aunque en los diseños se aprecie orientación a clases, por la forma en la que trabajan distintos frameworks empleados, se ha hecho una implementación en su mayoría orientada a procedimientos.

La implementación (con el código fuente debidamente documentado y comentado) y todos los artefactos relacionados se encuentran adjuntos a este documento. Debido al carácter evolutivo y cambiante de la aplicación el código fuente adjunto puede no estar en su versión más actual en el momento de la consulta, es por ello que la última versión puede encontrarse en el repositorio https://gitlab.com/resiajf/services/backend-panel-del-residente. Esta presenta una serie de dependencias externas las cuales pueden verse en el archivo “requirements.txt”. Entre ellas caben destacar, no obstante:

- Connexion: Framework desarrollado por Zalando el cual implementa todo el boilerplate necesario para poner en marcha una RESTFul API [46].
- Cliente de Redis: Cliente que permite conectarse al almacenamiento no persistente.
- ORM SQLAlchemy: Framework que permite llevar a cabo un mapeo objeto-relacional.
- Flask-Sockets: Librería que permite tener un servidor de WebSockets.
- Flask: Librería que se encarga de las labores de enrutamiento, es una dependencia de Connexion.
- PyJWT: Librería Python que facilita la gestión de token JWT (validar, decodificar y generar).

5.5. Testing

Para probar la API se ha empleado la herramienta Postman [53] la cual facilita el poder llevar a cabo pruebas funcionales del sistema sobre la interfaz de la API, de forma que para unas entradas determinadas se comprueba que la salida se corresponde con lo esperado.

5.6. Consideraciones de seguridad

El que una aplicación funcione en red (como es el caso) la hace vulnerable a una cantidad aún mayor de ataques. Es por ello que se han tomado una serie de

consideraciones esenciales para evitar estas situaciones:

- Emplear las últimas versiones disponibles de frameworks y librerías, que incorporan las correcciones de seguridad más actuales.
- Realizar todas las comunicaciones en red empleando seguridad en la capa de transporte mediante SSL/TLS.
- Nunca persistir contraseñas en plano, debe guardarse un hash de estas.
- Validar siempre la entrada de usuario para garantizar que esta no contiene código maligno.

5.5. Despliegue

El despliegue de la aplicación se ha llevado a cabo empleando la tecnología de virtualización mediante contenedores descrita en la sección de VoIP. En [54] se encuentra descrita en gran detalle la técnica empleada para lograr la virtualización.

Por motivos de escalabilidad se despliegan varias réplicas del backend a las cuales se redirige la petición del cliente mediante un proxy, logrando así redundancia. El que el backend se exponga mediante una API HTTP, y que HTTP sea un protocolo sin estado facilita estas tareas.

El optar por tener varias réplicas del backend desplegadas simultáneamente responde a un objetivo principal que se quiere lograr: el de disponibilidad.

Haciendo esto la aplicación es más tolerante a fallos, puesto que si una de las réplicas falla y experimenta una excepción, habrá otra réplica disponible para atender la petición.

6. Conclusiones y trabajos futuros

El haber podido llevar a cabo este trabajo ha sido una experiencia completamente satisfactoria y enriquecedora, la cual se ha ido desarrollado desde hace 4 años hasta el presente.

En todo ese tiempo el conocimiento obtenido ha sido de un valor incalculable (combinado con el del grado a su vez), a la vez que la enorme experiencia por el hecho de estar constantemente enfrentados a la realidad, a problemas y situaciones reales.

Este trabajo constituye una de nuestras mayores y valiosas aportaciones a las TIC de la residencia, entre otras que hemos hecho y no hemos señalado.

No cabe duda que las TIC de una residencia universitaria (y de un recinto universitario en general) están sometidas a una alta demanda por sus usuarios y es por ello que con esto no acaba la mejora de los servicios e infraestructura de la residencia: Hay una serie de trabajos futuros planeados los cuales seguirán a este en su desarrollo.

De hecho esta misma labor es cambiante, evoluciona y se ha pensado, diseñado e implementado para que así sea, y para que otorgue la flexibilidad suficiente para atender las demandas futuras de las tecnologías de la información de la residencia durante un buen tiempo.

Muchas de las mejoras propuestas, y otras que están en marcha no gozan de una velocidad de desarrollo lo suficientemente rápida debido a que el avance de estas depende de una cantidad considerable de instituciones y personas, repercutiendo todas estas dependencias político administrativas en la velocidad con la que se ejecutan dichas mejoras.

De cara al futuro se pretenden incorporar las siguientes mejoras y novedades en la instalación:

- Solucionar problemas de cobertura inalámbrica en ciertas zonas del recinto que se ha detectado que recepcionan un ancho de banda ligeramente menor al fijado.

- Finalizar la implementación de Eduroam en la residencia.
- Acabar de implementar y desplegar finalmente el sistema de visualización de consumo eléctrico.
- Integrar en la aplicación del “Panel del Residente” más tareas de administración que se dan en la residencia.
- Instalar un sistema de refrigeración adecuado en los lugares en los que se encuentre equipamiento de red (switches, SAIs, servidores).

Referencias

[1] "Cableado estructurado - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Cableado_estructurado. [Accedido: 2-ago-2018]

[2] "Chapter 4: Cabling". [En línea]. Disponible en: https://fcit.usf.edu/network/chap4/chap4.htm. [Accedido: 2-ago-2018]

[3] "Grounding for Screened and Shielded Network Cabling - Siemon". [En línea]. Disponible en: https://www.siemon.com/us/white_papers/06-07-20-grounding.asp. [Accedido: 2-ago-2018]

[4] "IEEE 802.3 - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/IEEE_802.3. [Accedido: 2-ago-2018]

[5] "Power over Ethernet - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Power_over_Ethernet. [Accedido: 3-ago-2018]

[6] "IEEE 802.11 - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/IEEE_802.11. [Accedido: 3-ago-2018]

[7] "Wireless Standards: 802.11a, 802.11b/g/n and 802.11ac". [En línea]. Disponible en: https://www.lifewire.com/wireless-standards-802-11a-802-11b-g-n-and-802-11ac-816553. [Accedido: 4-ago-2018]

[8] "Why Does Running My Microwave Kill My Wi-Fi Connectivity? ". [En línea]. Disponible en: https://www.howtogeek.com/171869/why-does-running-my-microwave-kill-my-wi-fi-connectivity/. [Accedido: 4-ago-2018]

[9] "IEEE 802.1X - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/IEEE_802.1X. [Accedido:5-ago-2018]

[10] "Extensible Authentication Protocol - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Extensible_Authentication_Protocol. [Accedido: 5-ago-2018]

[11] "RADIUS - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/RADIUS. [Accedido: 5-ago-2018]

[12] "EAP-PEAP and EAP-TTLS Authentication with a RADIUS Server". [En línea]. Disponible en: https://www.interlinknetworks.com/app_notes/eap-peap.htm. [Accedido: 5-ago-2018]

[13] "Inventory of 802.1X-based solutions for inter-NRENs roaming". [En línea]. Disponible en: https://www.terena.org/activities/tf-mobility/deliverables/delD/DelD_v1.2-f.pdf.

[Accedido: 5-ago-2018]

[14] "Voz sobre protocolo de internet - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Voz_sobre_protocolo_de_internet. [Accedido: 6-ago-2018]

[15] "Session Initiation Protocol - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Session_Initiation_Protocol. [Accedido: 6-ago-2018]

[16] "Open Source Communications Software | Asterisk Official Site
". [En línea]. Disponible en: https://www.docker.com/resources/what-container. [Accedido: 6-ago-2018]

[17] "What is a Container | Docker". [En línea]. Disponible en: https://www.docker.com/resources/what-container. [Accedido: 7-ago-2018]

[18] "Docker (software) - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Docker_(software). [Accedido: 7-ago-2018]

[19] "IEEE 802.1: 802.1Q-2014 - Bridges and Bridged Networks". [En línea]. Disponible en: http://www.ieee802.org/1/pages/802.1Q-2014.html. [Accedido: 8-ago-2018]

[20] "Web application - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Web_application. [Accedido: 8-ago-2018]

[21] "RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1". [En línea]. Disponible en: https://tools.ietf.org/html/rfc2616. [Accedido: 9-ago-2018]

[22] "Front and back ends - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Front_and_back_ends. [Accedido: 9-ago-2018]

[23] "Web browser - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Web_browser. [Accedido: 10-ago-2018]

[24] "100+ JavaScript Frameworks for Web Developers". [En línea]. Disponible en: https://cssauthor.com/javascript-frameworks/. [Accedido: 10-ago-2018]

[25] " 9 Popular JavaScript Frameworks Used in 2018 · Raygun Blog". [En línea]. Disponible en: https://raygun.com/blog/popular-javascript-frameworks/. [Accedido: 11-ago-2018]

[26] "Sass: Syntactically Awesome Style Sheets". [En línea]. Disponible en: https://sass-lang.com/. [Accedido: 12-ago-2018]

[27] "CoffeeScript". [En línea]. Disponible en: https://coffeescript.org/. [Accedido: 21-ago-2018]

[28] " TypeScript - JavaScript that scales. ". [En línea]. Disponible en: https://www.typescriptlang.org/. [Accedido: 21-ago-2018]

[29] "Database - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Database. [Accedido: 22-ago-2018]

[30] " BECAS - Beca para colaboradores residencia universitaria - Universidad de Málaga". [En línea]. Disponible en: https://www.uma.es/becas/info/100481/beca-para-colaboradores-residencia-universitaria/. [Accedido: 22-ago-2018]

[31] "Bring your own device - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Bring_your_own_device. [Accedido: 23-ago-2018]

[32] "What is eduroam? – eduroam". [En línea]. Disponible en: https://www.eduroam.org/what-is-eduroam/. [Accedido: 23-ago-2018]

[33] "A Radiation Oncologist Says Everything You Need To Hear About WiFi And Cancer Risk". [En línea]. Disponible en: https://www.forbes.com/sites/quora/2016/05/19/a-radiation-oncologist-says-everything-you-need-to-hear-about-wifi-and-cancer-risk/#15a7d57e7267. [Accedido: 24-ago-2018]

[34] "UniFi". [En línea]. Disponible en: https://unifi-sdn.ubnt.com/. [Accedido: 25-ago-2018]

[35] "5GHz WiFi Disadvantages". [En línea]. Disponible en: https://www.alternativewireless.com/resources/wifi-networking/5ghz-vs-2-4ghz-wireless-lan/5ghz-wifi-disadvantages.html. [Accedido: 26-ago-2018]

[36] "Red digital de servicios integrados - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Red_digital_de_servicios_integrados. [Accedido: 27-ago-2018]

[37] "Opus (audio format) - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Opus_(audio_format)#VoIP_support. [Accedido: 28-ago-2018]

[38] "Modulación por impulsos codificados - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_impulsos_codificados. [Accedido: 29-ago-2018]

[39] "Codec - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Codec. [Accedido: 1-sept-2018]

[40] "Digital container format - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Digital_container_format. [Accedido: 1-sept-2018]

[41] "WAV - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/WAV.

[Accedido: 2-sept-2018]
[42] "RFC 3261 - SIP: Session Initiation Protocol". [En línea]. Disponible en: https://tools.ietf.org/html/rfc3261. [Accedido: 4-sept-2018]
[43] "Understanding SIP Addresses | SIP Adventures". [En línea]. Disponible en: https://andrewjprokop.wordpress.com/2014/03/24/understanding-sip-addresses/. [Accedido: 5-sept-2018]
[44] "Home - Asterisk Project - Asterisk Project Wiki". [En línea]. Disponible en: https://wiki.asterisk.org/wiki/display/AST/Home. [Accedido: 5-sept-2018]
[45] "SIP trunking - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/SIP_trunking. [Accedido: 6-sept-2018]
[46] "GitHub - zalando/connexion: Swagger/OpenAPI First framework for Python on top of Flask with automatic endpoint validation & OAuth2 support". [En línea]. Disponible en: https://github.com/zalando/connexion. [Accedido: 6-sept-2018]
[47] " SQLAlchemy - The Database Toolkit for Python ". [En línea]. Disponible en: https://www.sqlalchemy.org/. [Accedido: 7-sept-2018]
[48] "Redis". [En línea]. Disponible en: https://redis.io/. [Accedido: 7-sept-2018]
[49] "JSON Web Tokens - jwt.io". [En línea]. Disponible en: https://jwt.io/. [Accedido: 7-sept-2018]
[50] "Security Assertion Markup Language - Wikipedia, la enciclopedia libre". [En línea]. Disponible en: https://es.wikipedia.org/wiki/Security_Assertion_Markup_Language. [Accedido: 8-sept-2018]
[51] "Representational state transfer - Wikipedia". [En línea]. Disponible en: https://en.wikipedia.org/wiki/Representational_state_transfer. [Accedido: 8-sept-2018]
[52] " OpenAPI Specification | Swagger ". [En línea]. Disponible en: https://swagger.io/specification/v2/. [Accedido: 9-sept-2018]
[53] "Postman | API Development Environment". [En línea]. Disponible en: https://www.getpostman.com/. [Accedido: 9-sept-2018]
[54] Melchor Alejo Garau Madrigal, “Despliegue de infraestructura y servicios de red en la Residencia Universitaria “Alberto Jiménez Fraud” Despliegue de infraestructura, implantación de telefonía IP y desarrollo del backend de la aplicación de gestión “Panel del residente””, trabajo de fin de grado, Universidad de Málaga, 2018.

Fuentes de las figuras

A continuación se proporciona un listado con las fuentes de las figuras empleadas en el documento:

1. https://en.wikipedia.org/wiki/IEEE_802.1X#/media/File:802.1X_wired_protocols.png
2. https://www.interlinknetworks.com/app_notes/eap-peap.htm
3. https://www.docker.com/resources/what-container
4. https://hackernoon.com/how-it-feels-to-learn-javascript-in-2016-d3a717dd577f
5. http://www.infouma.uma.es/fichas/resiuma.html
6. Diagrama elaborado por Melchor [54]
7. Diagrama elaborado por Melchor [54]
8. Diagrama elaborado por Melchor [54]
9. Heatmap elaborado con el Software “Acrylic WiFi”
10. Plano suministrado por el SCI UMA
11. Plano suministrado por el SCI UMA
12. Plano suministrado por el SCI UMA
13. Plano suministrado por el SCI UMA
14. Elaboración propia a partir de foto aérea de Google Maps
15. Elaboración propia a partir de fragmentos de planos suministrados por la UMA
16. Sitio web de Unifi
17. Elaboración propia a partir de fragmentos de planos suministrados por la UMA
18. Elaboración propia a partir de fragmentos de planos suministrados por la UMA
19. Elaboración propia a partir de fragmentos de planos suministrados por la UMA
20. Elaboración propia a partir de heatmaps elaborado con el Software “Acrylic WiFi”
21. Sitio web de Unifi
22. Foto propia
23. Foto propia
24. Foto propia

25. Foto propia
26. Foto propia
27. Foto propia
28. Foto propia
29. Foto propia
30. Elaboración propia a partir de fragmentos de planos suministrados por la UMA
31. Imagen extraída del Software de gestión de red Unifi
32. Imagen extraída del Software de gestión de red Unifi
33. Imagen extraída del Software de gestión de red Unifi
34. Imagen elaborada por Melchor
35. Creación propi a partir de de otras imágenes
36. Imagen extraída del sitio web de Yealink
37. Foto propia
38. Foto propia
39. https://www.voipmechanic.com/sip-call-example.htm
40. https://www.eetimes.com/document.asp?doc_id=1274451
41. Elaboración propia
42. Elaboración propia
43. https://www.researchgate.net/figure/Docker-Client-Server-Architecture-5_fig3_325390165
44. Elaborada con la herramienta online de dibujo draw.io
45. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
46. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
47. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
48. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
49. Elaborado con MySQL Workbench
50. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
51. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"

52. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
53. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
54. Elaborada con la herramienta online "http://www.umlet.com/umletino/umletino.html"
55. Captura de pantalla propia
56. Captura de pantalla propia
57. Captura de pantalla propia
58. Captura de pantalla propia
59. Captura de pantalla propia

Glosario

TIC	Tecnologías de la Información y la Comunicación
UTP	Unshielded Twisted Pair
FTP	Foiled Twisted Pair
SFTP	Shielded Foiled Twisted Pair
OSI	Open System Interconnection
PoE	Power over Ethernet
WiFi/Wi-Fi/Wifi/wifi	Wireless Fidelity
EAP	Extensible Authentication Protocol
EAP-TTLS	EAP Tunneled Transport Layer Security
PAP	Password Authentication Protocol
RADIUS	Remote Authentication Dial-In User Service
EAPOL	EAP over LAN
LAN	Local Area Network
AVP	Attribute Value Pair
RDSI	Red Digital de Servicios Integrados
VoIP	Voice over IP
IP	Internet Protocol
SIP	Session Initiation Protocol
PBX	Private Branch Exchange
RTP	Real-time Transport Protocol
CGI	Common Gateway Interface
REST	Representational State Transfer
LXC	Linux Containers
VLAN	Virtual Local Area Network
HTTP	Hypertext Transfer Protocol
CSS	Cascading Style Sheets

HTML	Hypertext Markup Language
SQL	Structured Query language
SAI	Sistema de Alimentación Ininterrumpida
QoS	Quality of Service
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
EPI	Equipo de Protección Individual
WPA	Wi-Fi Protected Access
MAC	Media Access Control
ARI	Asterisk Restful Interface
SIPS	Secure Session Initiation Protocol
SRTP	Secure Real-time Transport Protocol
PMR	Persona de Movilidad Reducida
DAC	Digital-to-Analog Conversion
ADC	Analog-to-Digital Conversion
LPCM	Linear Pulse Code Modulation
PCM	Pulse Code Modulation
AoR	Address of Record
ISP	Internet Service Provider
REST	REpresentational State Transfer
JSON	JavaScript Object Notation
API	Application Programming Interface
JWT	JSON Web Token
DUMA	Directorio de la Universidad de Málaga
ORM	Object-Relational Mapping
RFC	Request For Comments
IdP	IDentity Provider
NIU	Número de Identificación Universitaria
URL	Uniform Resource Locator

SSO	Single Sign On
OAS	Open API Specification
YAML	Yet Another Markup Language
WSDL	Web Services Description Language
SOAP	Simple Object Access Protocol
XHTML	eXtensible HyperText Markup Language
SSL	Secure Sockets Layer
TLS	Transport Layer Security
TFG	Trabajo de Fin de Grado

Anexos

En las siguientes páginas se adjuntan una serie de figuras las cuales son referenciadas a lo largo del documento. Estas consisten principalmente en planos u otras imágenes de dimensiones considerables.

Si las dimensiones de las figuras son tales que aún así no es posible apreciarlas con claridad o nitidez, se adjuntan estas de forma independiente a este documento como archivo de imágen.

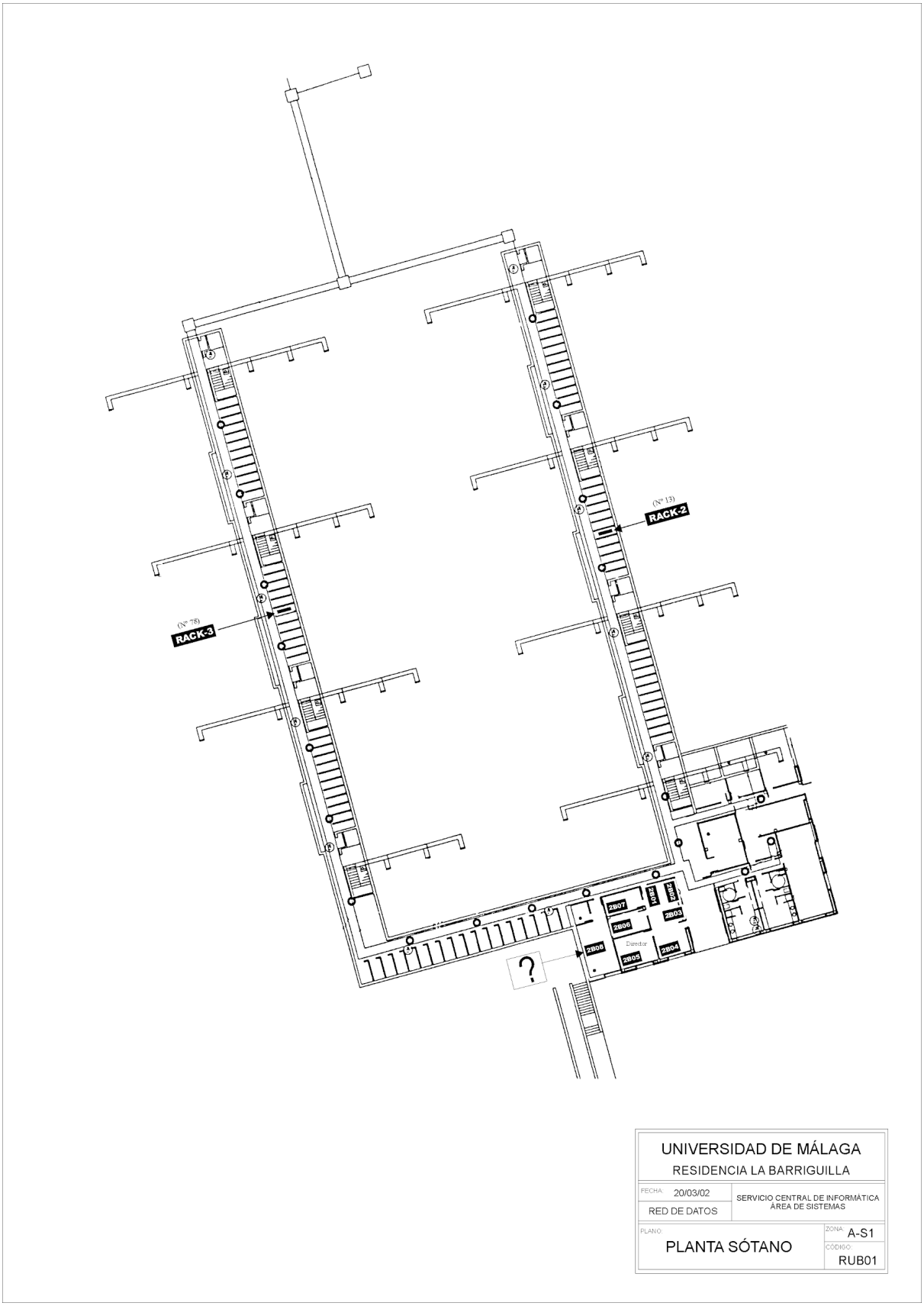


Figura 10: Plano de la red de datos cableada de la residencia de la planta sótano

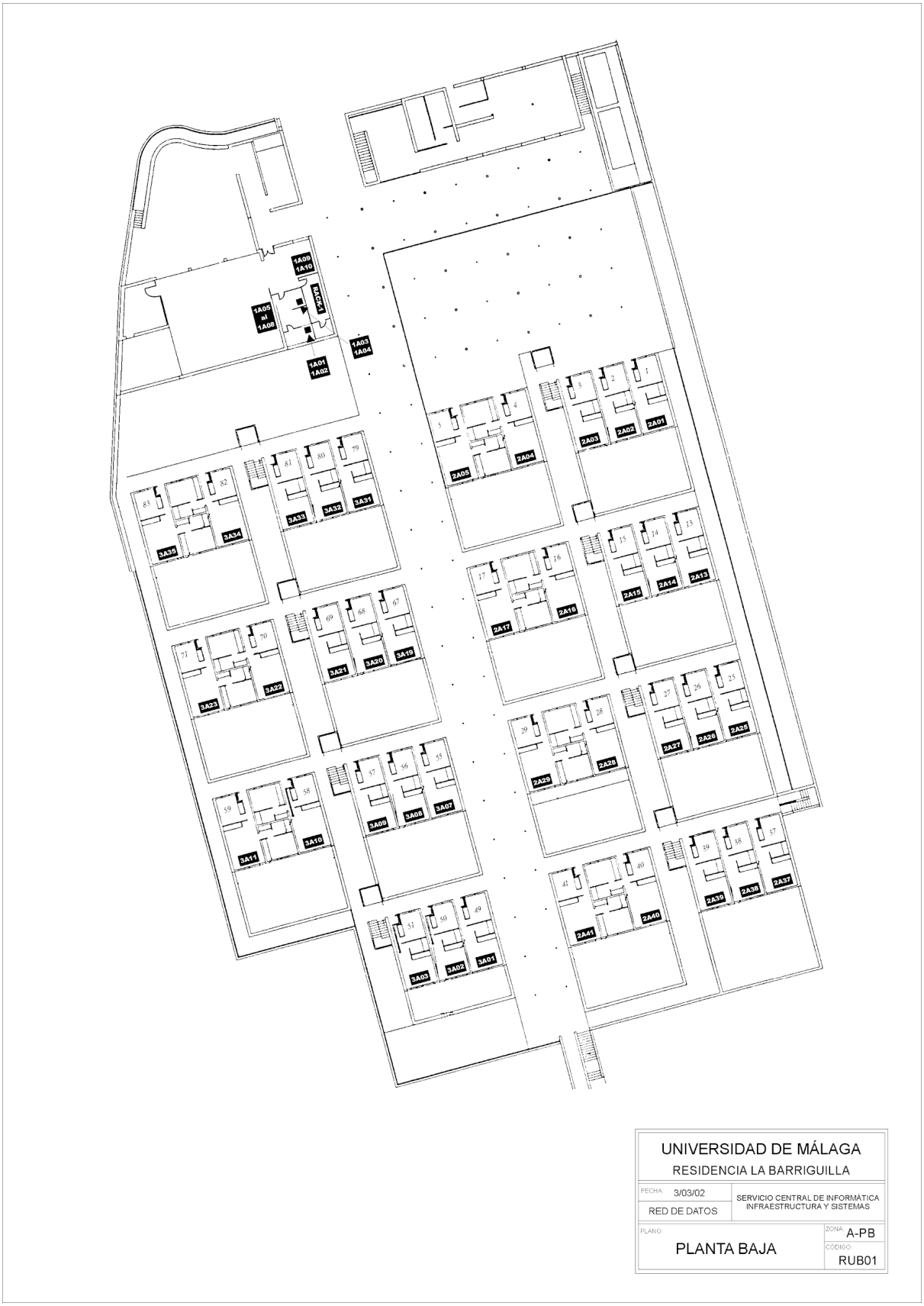


Figura 11: Plano de la red de datos cableada de la residencia de la planta baja

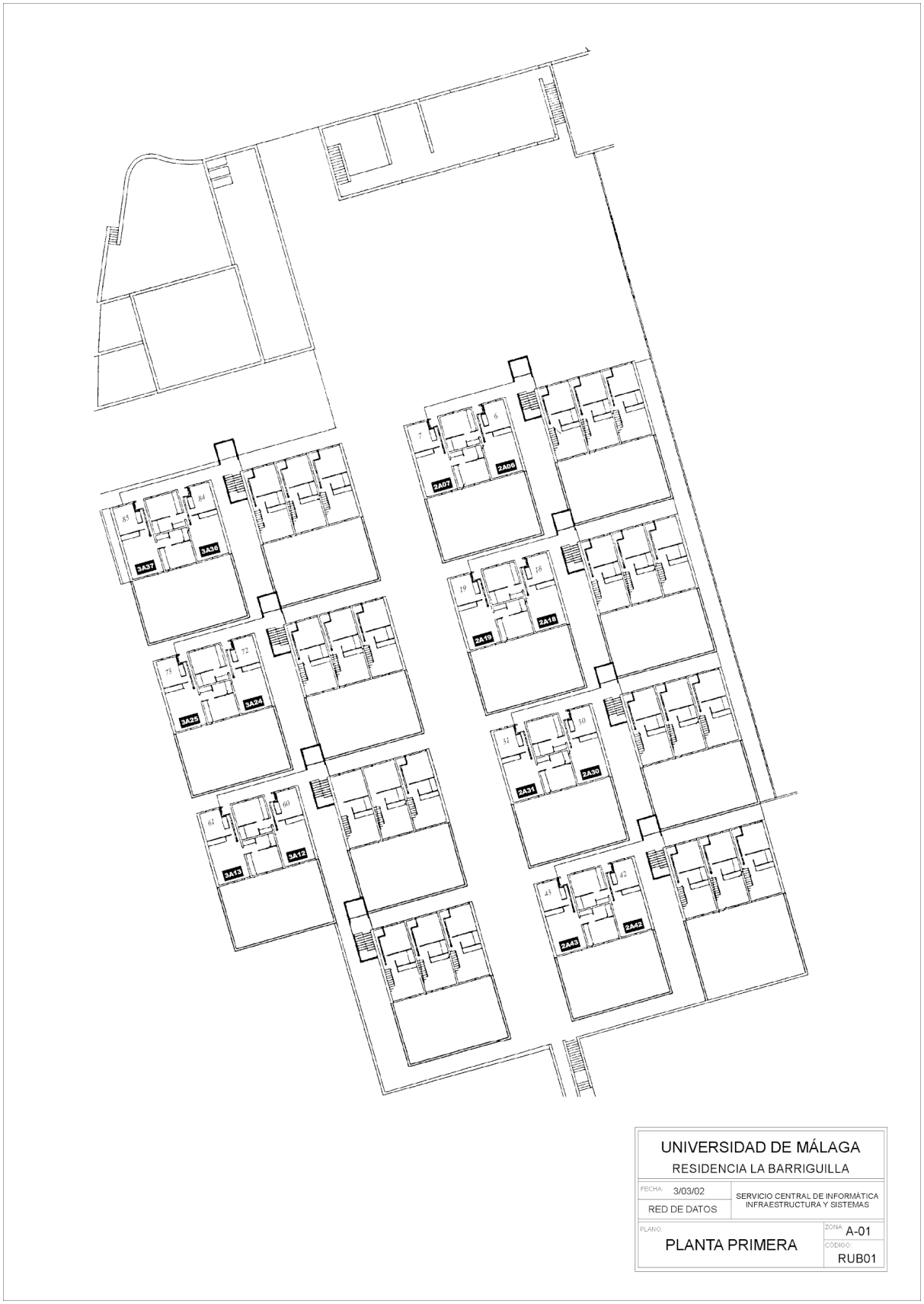


Figura 12: Plano de la red de datos cableada de la residencia de la planta 1

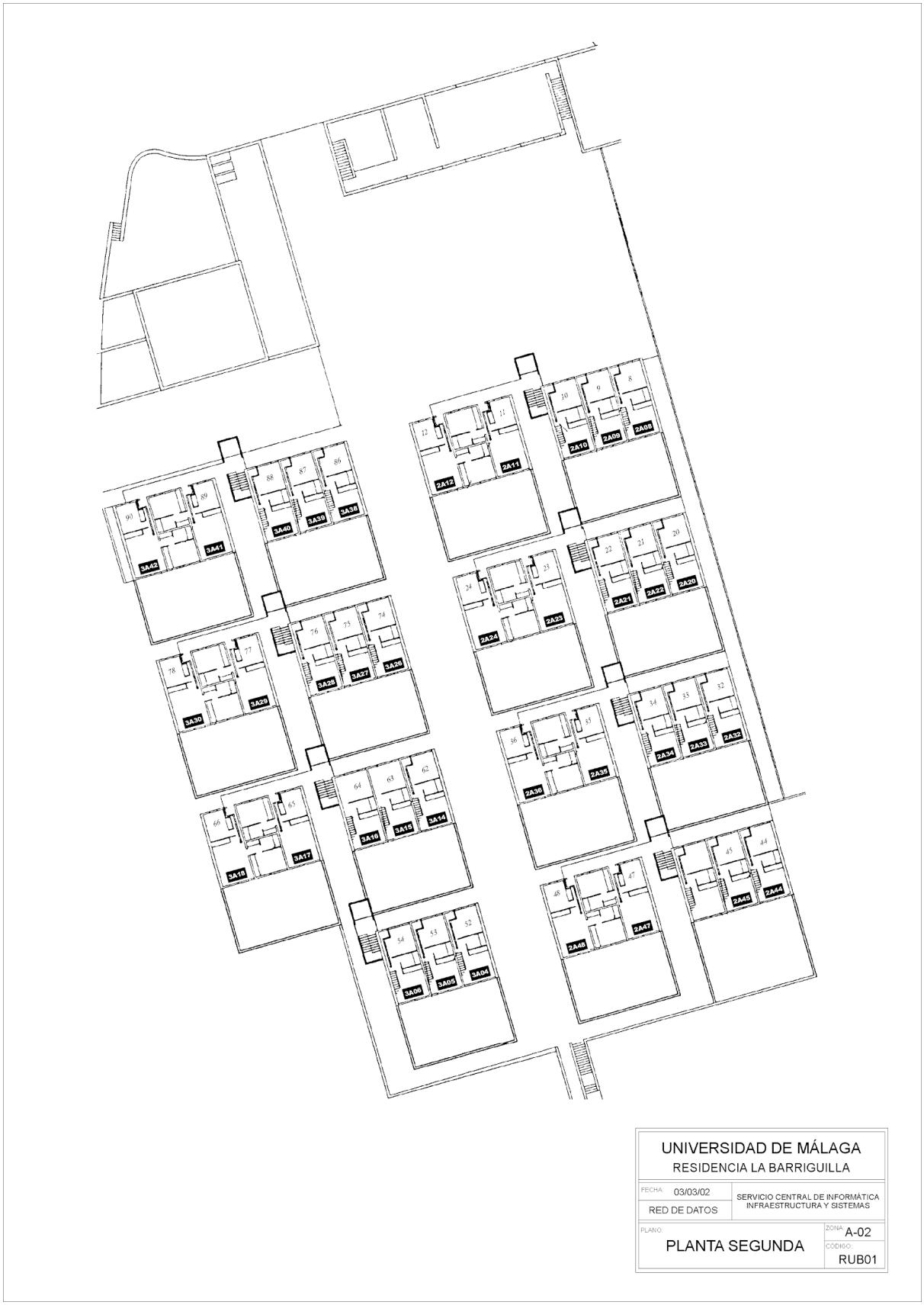
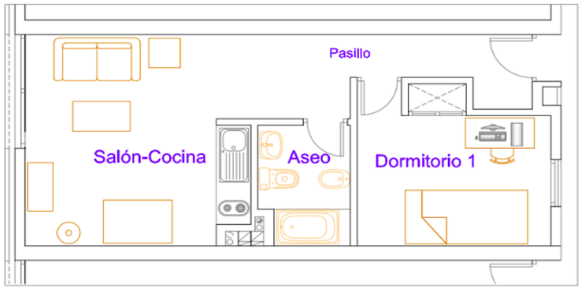


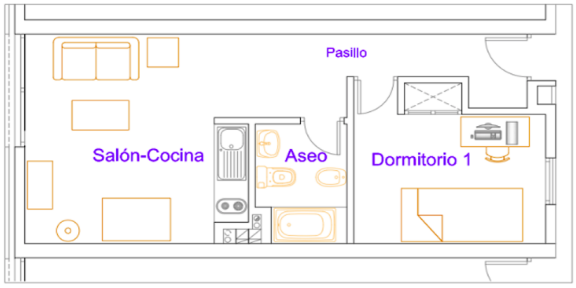
Figura 13: Plano de la red de datos cableada de la residencia de la planta 2

Tipología de apartamentos

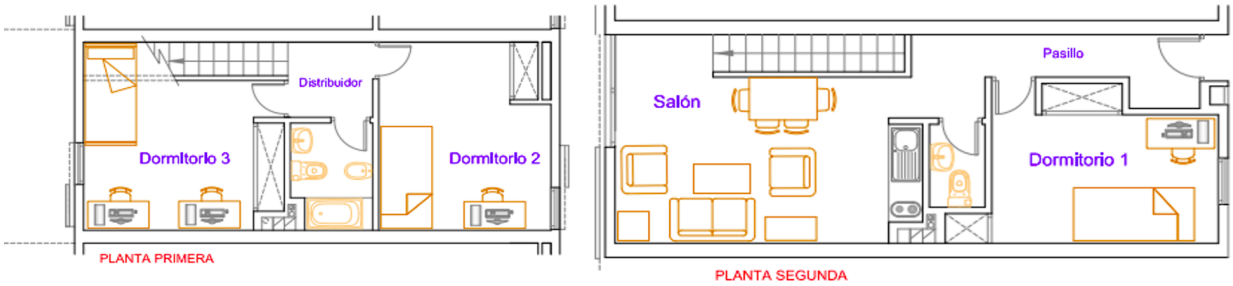
Tipo A



Tipo B



Tipo C



Ubicación de los apartamentos en el contexto de un bloque

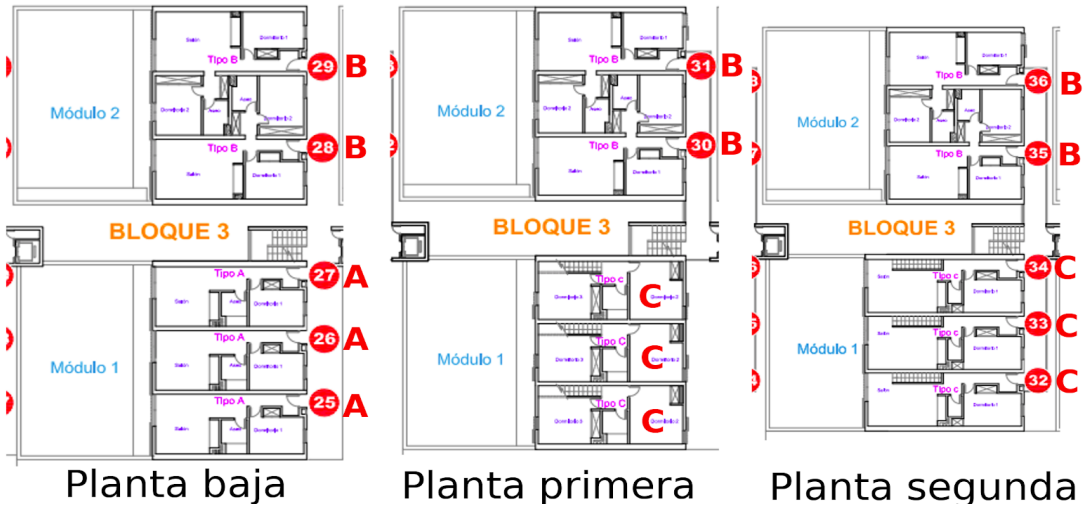


Figura 15: Tipología de apartamentos y ubicación de estos en el contexto de un bloque

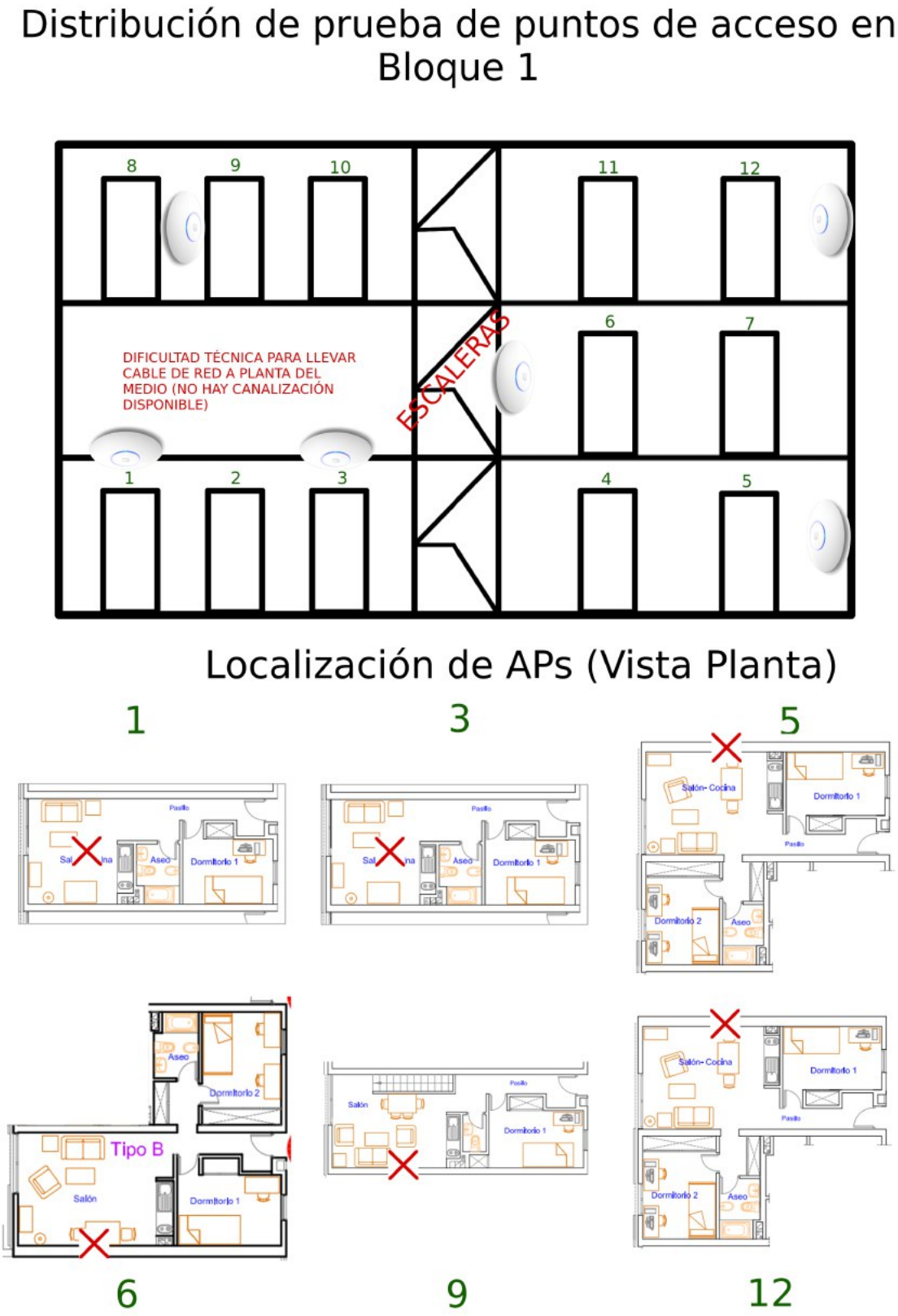


Figura 17: Disposición de los puntos de acceso

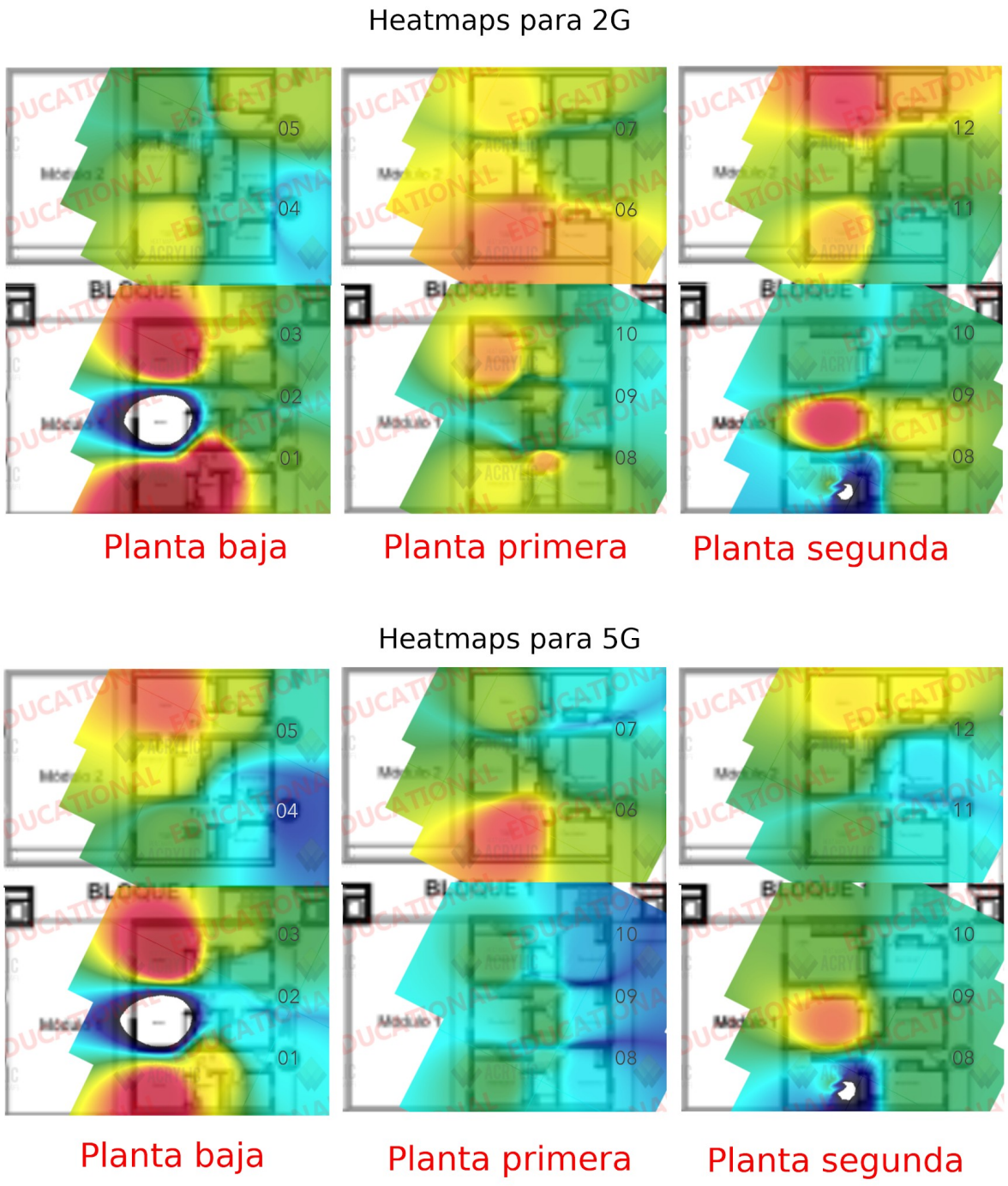


Figura 20: Heatmaps de la instalación de prueba propuesta, desplegada en el bloque 1, como se puede observar la intensidad de señal es menor para 5G al operar en mayor frecuencia.

Figura 22: Armario rack antiguo

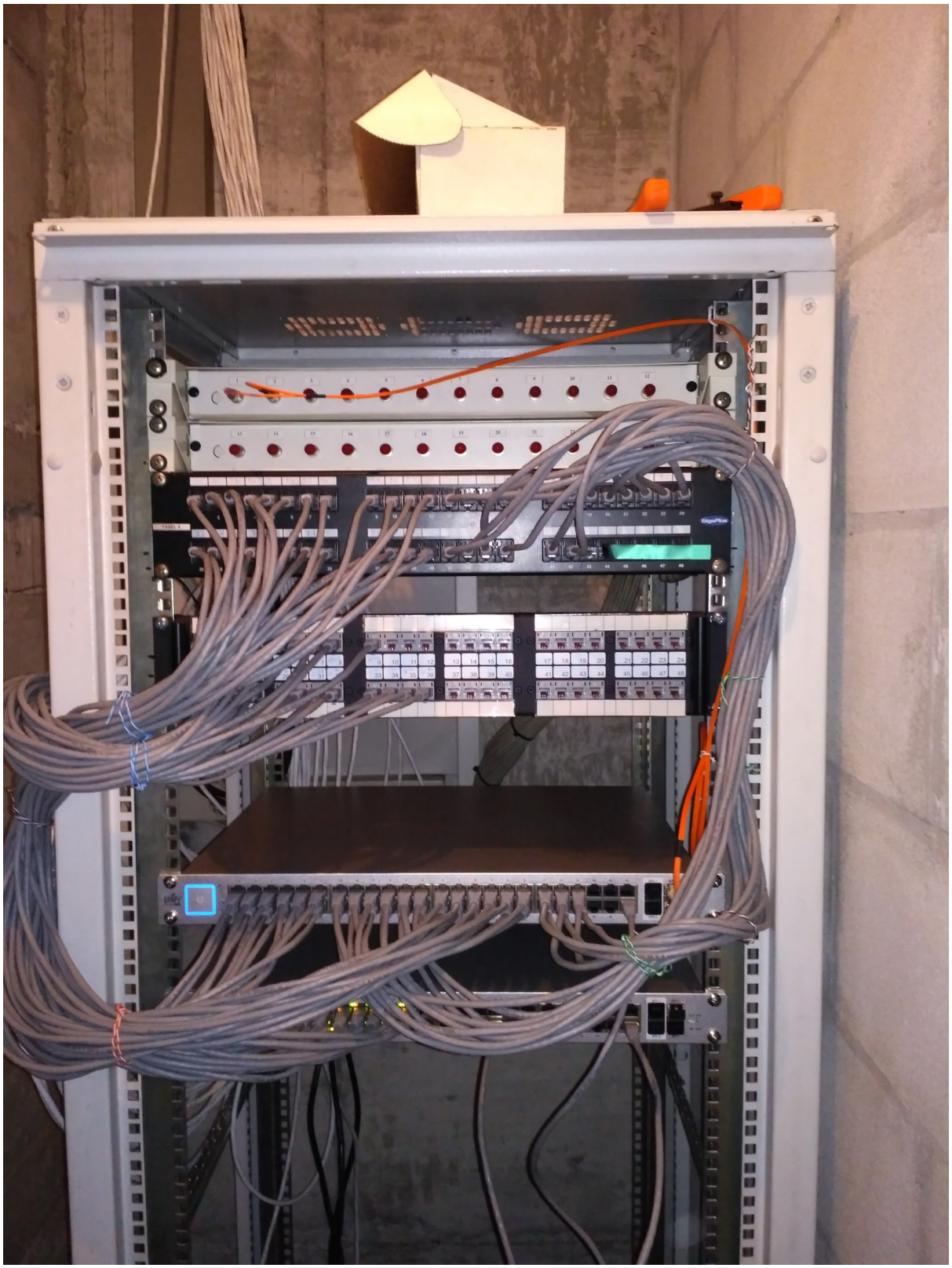


Figura 23: Nuevo armario rack, más organizado

Figura 24: Se recicló todo lo posible

Figura 25

Figura 26

Figura 27

Figura 28

Figura 29

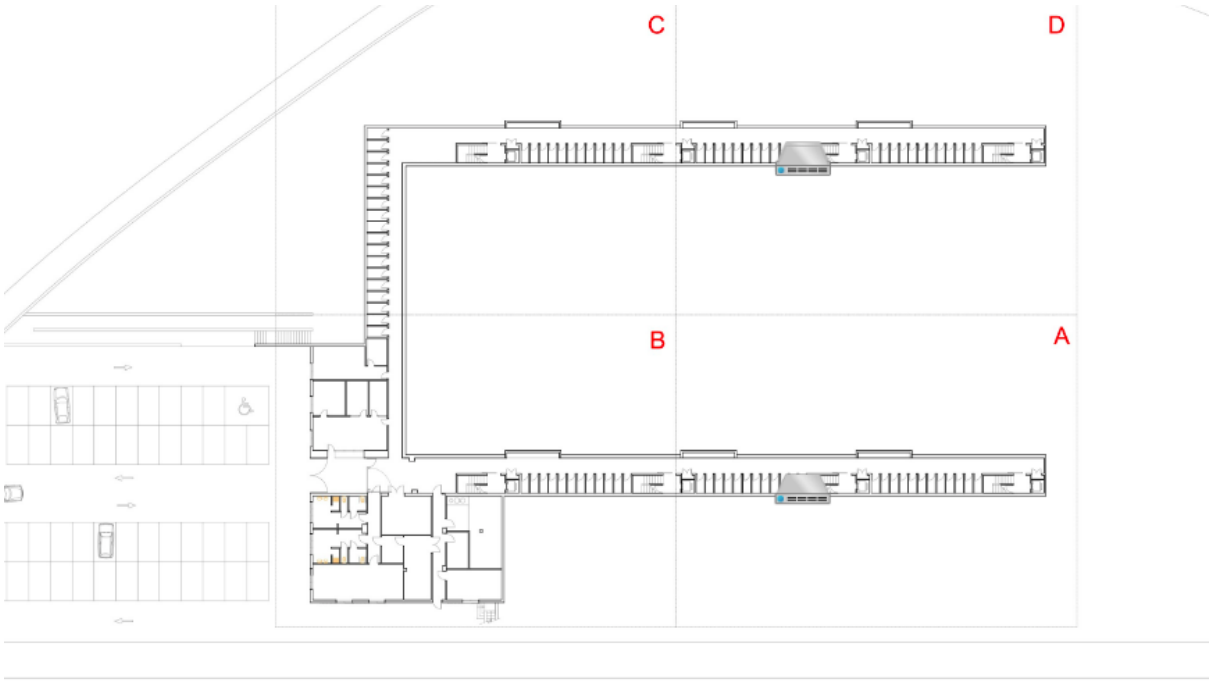


Figura 30: Disposición del equipamiento en el sótano

Figura 31: Disposición del equipamiento en la planta baja

Figura 32: Disposición del equipamiento en la planta primera

Figura 33: Disposición del equipamiento en la planta segunda

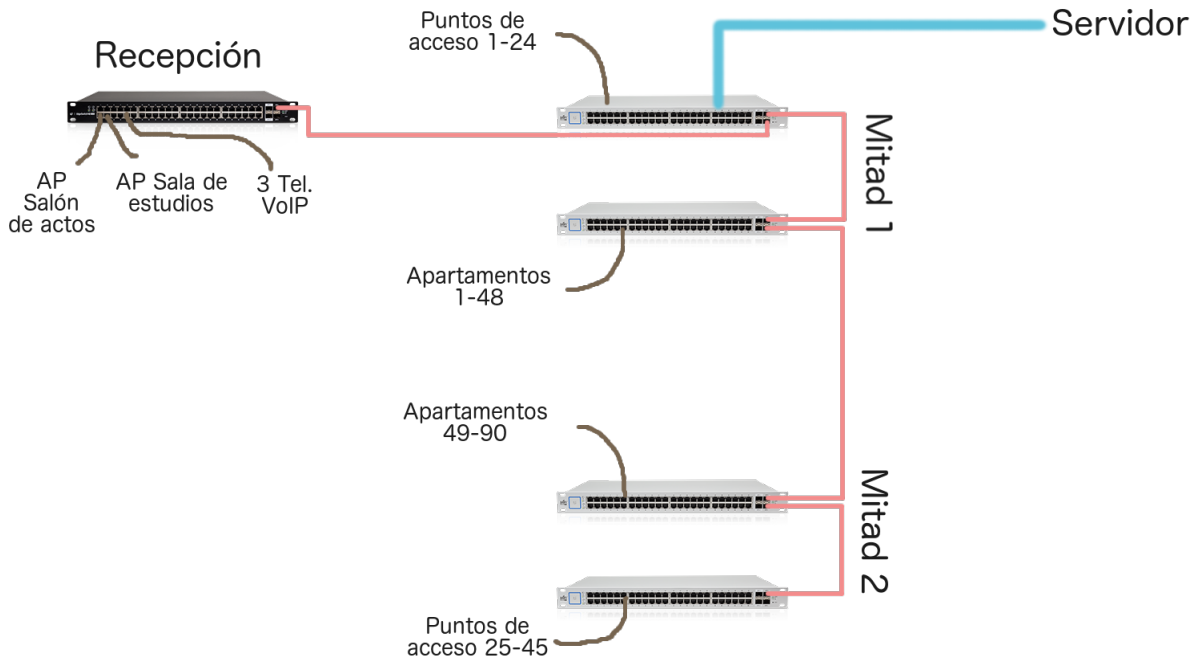


Figura 34: Instalación final de cable, las línea rosas representan enlaces de fibra, el resto son enlaces cableados

Figura 36: Teléfono IP Yealink SIP-T21P E2

Figura 37: Instalación resultando del teléfono en un apartamento

Figura 38: Sistema empleado para extender la toma del switch

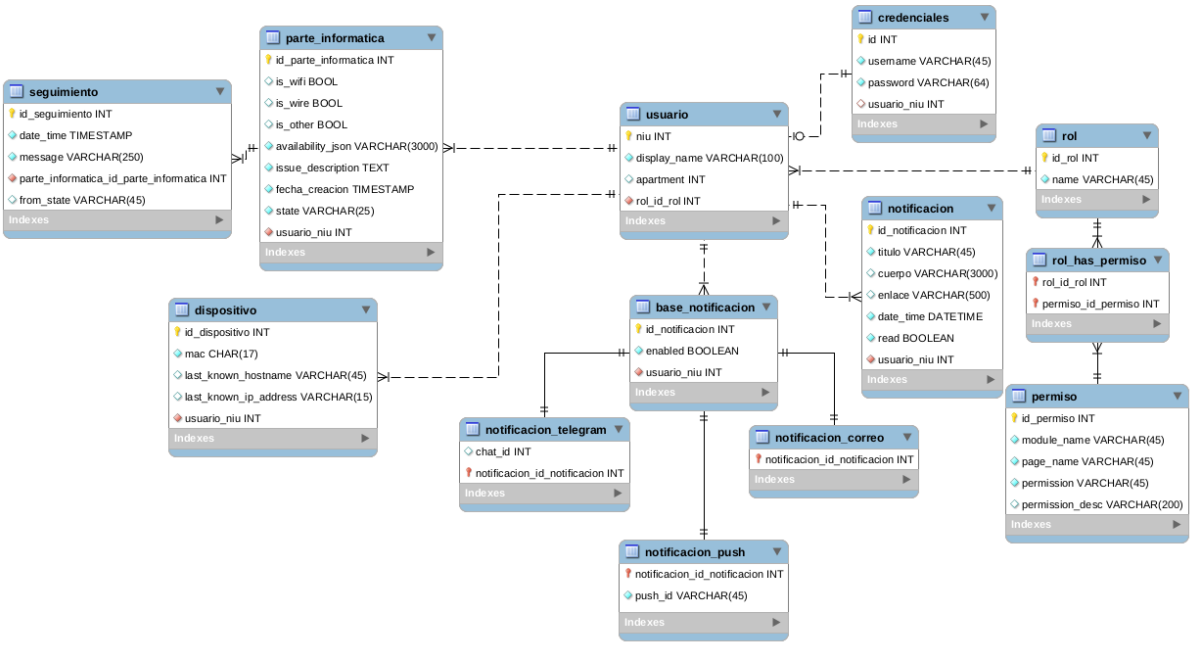


Figura 49: Diagrama entidad relación del sistema