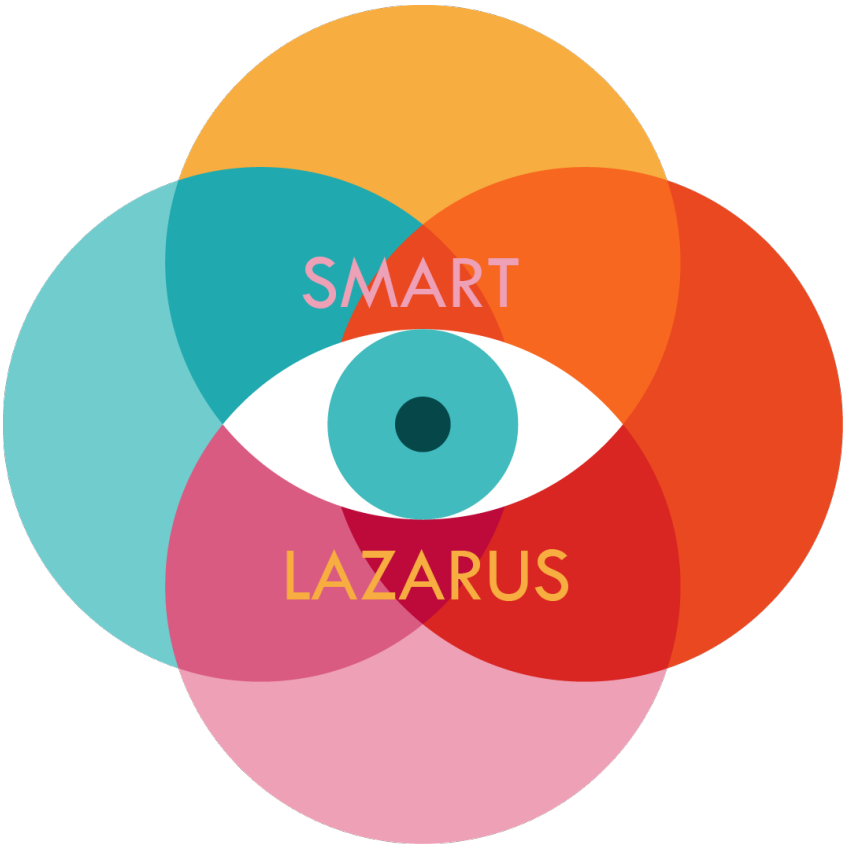

SMARTLAZARUS

CÁTEDRAS DE TECNOLOGÍA ACCESIBLE

2019

indra

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

CONTENIDO

INTRODUCCIÓN

La identificación y guiado de personas y activos es una pieza clave para una adecuada personalización de servicios e interacción con el entorno. En este sentido, el conocimiento de la ubicación geográfica exacta de personas y objetos puede resultar de gran utilidad en múltiples ámbitos, tales como el sector de la industria o el de los servicios, entre otros muchos.

De hecho, el desarrollo de sistemas de localización de personas es un aspecto actualmente muy demandado en el sector de los servicios y la accesibilidad por su elevado nivel de utilidad.

Las soluciones de localización y guiado constan, generalmente, de tres componentes básicos:

1. Un **subsistema de localización y actuación para elementos móviles y fijos,** que faciliten la interacción entre todos los sistemas y con el entorno.
2. Un **subsistema de transmisión** de dicha información a una unidad central.
3. Una **plataforma de gestión, planificación y monitorización** que recoge la información para su posterior procesamiento.

El principal problema de estas soluciones es que no existen herramientas plenamente funcionales que permitan efectuar una localización y guiado tanto en entornos interiores como exteriores. Este proyecto surge con la motivación de obtener una plataforma que sea capaz de aplicar estas características fundamentales de los sistemas de localización y guiado para que las personas puedan ser guiadas en ambos tipos de entorno.

A partir de los requisitos iniciales marcados al comienzo del proyecto SmartLazarus (especificados en la memoria técnica de solicitud del proyecto), se ha trabajado en el desarrollo de una innovadora plataforma unificada de sensorización, geolocalización y guiado en tiempo real, orientada al seguimiento y monitorización de recursos en movilidad, tanto humanos como técnicos.

Este documento recoge, a modo de sumario, los principales objetivos del proyecto, así como una descripción la planificación del trabajo y de las actividades realizadas hasta el momento. Se trata de un documento vivo que se irá actualizando a medida que las diferentes tareas marcadas en la hoja de ruta del proyecto se vayan cumpliendo.

OBJETIVOS DEL PROYECTO

OBJETIVOS GENERALES

A continuación, se presentan los principales objetivos que se definieron en la memoria técnica de propuesta del proyecto:

1. Obtener una plataforma basada en el modelo ***Cloud Computing*** que facilite la construcción de sistemas de sensorización y geolocalización a partir de múltiples fuentes y que proporcione un impacto tecnológico e industrial relevante en España, introduciendo nuevas oportunidades de empleo, mercado y desarrollo.
2. Proporcionar una mejora sustancial y objetiva en el campo de los sistemas de localización en tiempo real y las redes de sensores, sobre todo en la capacidad para hacer uso de múltiples tecnologías y de infraestructuras prexistentes.
3. Hacer posible el despliegue de entornos inteligentes con una mínima inversión, aprovechando en cada momento la infraestructura existente y adaptando las técnicas, procedimientos y funcionalidades de la plataforma en función de las características del entorno.
4. Introducir herramientas de última generación en lo que respecta a los sistemas de geolocalización hasta el momento no utilizadas en España, de cara a competir a nivel internacional, lo que se traducirá en un crecimiento considerable en aquellos sectores que hagan uso de esta tecnología y, por tanto, en un aumento significativo en la competitividad y la generación de empleo.

Una vez presentados los objetivos generales, a continuación, se presentan las principales innovaciones tecnológicas que se pretenden obtener tras el desarrollo de la plataforma SmartLazarus.

- **Una plataforma integral y universal** de neocartografía y geolocalización en exteriores e interiores con funcionalidades de sensorización y actuación sobre el entorno, la cual facilitará el acceso a las distintas funcionalidades de manera unificada y marcará un nuevo hito que revolucionará diferentes áreas tecnológicas:
    - Asimismo, SMARTLAZARUS revolucionará la forma de interactuar con los sistemas de localización y sensorización, pues los usuarios tendrán a su disposición una plataforma que proporcionará tanto localización en interiores y exteriores como funcionalidades de sensorización y actuación sobre el entorno, sin importar las tecnologías disponibles en cada momento.

- Un motor de localización en tiempo real que proporcione una alta precisión tanto en exteriores como en interiores, pudiendo ser alimentado por las diversas tecnologías disponibles en cada momento (por ejemplo, GPS, Wi-Fi, GPRS, ZigBee o Bluetooth). Este motor permitirá obtener la ubicación de personas, animales y objetos con una precisión inferior a 3 metros tanto en entornos interiores como exteriores.

APLICACIÓN PARA LA INTEGRACIÓN LABORAL Y SOCIAL DE PERSONAS CON DISCAPACIDAD

Las sociedades modernas se caracterizan por dos tendencias: la primera es el rápido desarrollo de las TIC, que ha influido en la vida de las personas en todos los sentidos; la otra es la sensibilidad que existe en gobiernos, empresas y asociaciones por conseguir que las personas con discapacidad o en riesgo de exclusión puedan llevar una vida independiente, lo que incluye de manera muy relevante la capacidad para desarrollar una actividad profesional remunerada. La integración efectiva de personas con discapacidad en el mundo laboral supone un enorme reto para la sociedad y a la vez plantea una oportunidad para hacer uso de las TIC. Un ejemplo típico es como Internet se ha convertido en un factor clave para la inclusión social.

Actualmente existen numerosas barreras que dificultan que las personas con discapacidad puedan integrarse al mercado laboral y por lo tanto que las empresas cuenten con estos profesionales en sus plantillas. En el caso de la incorporación al mercado laboral de las personas con discapacidad los principales retos son la autonomía personal (movilidad), procesado de la información (lenguaje, conocimiento numérico, aprendizaje de tareas, orientación espacial), actitud ante el trabajo (responsabilidad, atención, ritmo, organización, relaciones de trabajo, seguridad, interés…), control emocional, relaciones interpersonales, autodeterminación y otras.

Hay ciertos colectivos que se enfrentan a desventajas claras, uno de estos grupos es el de los discapacitados. Según estudios realizados por Naciones Unidas, en torno al 15% de la población mundial sufre algún tipo de discapacidad y, de acuerdo con International Labour Organization (ILO), existen alrededor de 386 millones de personas en edad de trabajar que sufren alguna discapacidad, llegando la tasa de desempleo entre éstos al 80% en el caso de algunos países.

Por ello es importante intervenir de manera que se mejore y favorezca su integración social, y por extensión, ocupacional. Ante esta problemática, hay cada vez mayor preocupación por parte de los distintos gobiernos, quienes promueven la integración laboral de los discapacitados a través de

distintas normativas y directivas, por ejemplo, las Normas Uniformes sobre la igualdad de oportunidades para las personas con discapacidad.

La falta de accesibilidad de los edificios y la integración de los discapacitados en el mundo laboral son dos de los elementos clave sobre los que se asientan las distintas normativas. Por norma general, los gobiernos obligan por ley a las empresas con un determinado número de trabajadores a la contratación de un cierto número de personas discapacitadas, por lo que los edificios donde estas empresas desarrollan su actividad deben estar adaptados para facilitar la accesibilidad a las personas discapacitadas según la legislación vigente.

Este estudio se centra tanto en la adecuación de entornos laborales, como de lugares públicos, como pueden ser aeropuertos, estaciones de tren o autobús, o grandes superficies comerciales. Es, tal vez, en este tipo de espacios públicos donde el sistema puede resultar de mayor utilidad, puesto que a priori no son conocidos por la persona y llegar al destino deseado puede ser una ardua tarea. En el caso de los entornos laborales, ocurre lo mismo cuando una persona se incorpora a la plantilla de una nueva empresa, ya que se ve sumergida en un entorno nuevo, al que se tiene que ir adaptando, y no son las personas las que se deberían adaptar al entorno, sino el entorno a las personas.

En el caso de personas con movilidad reducida, en este tipo de contexto, seremos capaces a través de este sistema de determinar los posibles problemas arquitectónicos del edificio, pudiendo guiar a las personas discapacitadas para facilitar su trayecto hasta su punto de destino deseado. Podremos facilitar la adecuación y accesibilidad con la eliminación de obstáculos en su trayecto, reduciendo así el aislamiento y dependencia de este colectivo. En el caso de personas invidentes o con visión reducida, el caso se agrava, ya que las referencias del entorno se ven limitadas, haciendo que tareas tan simples como coger un avión en una determinada terminal sean inalcanzables para estas personas, siendo siempre necesaria la ayuda de una persona. Con el uso de este sistema, la autonomía que puede llegar a alcanzar este colectivo es total ya que, gracias al sistema de guiado en interiores, será capaz de alcanzar cualquier destino sin necesidad de recurrir a la asistencia de nadie.

DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS

En este proyecto de I+D+I existe una fuerte componente aplicada de investigación. Se ha desarrollado una metodología que permite establecer un conjunto de objetivos, evaluarlos y definir los siguientes objetivos que se pretenden alcanzar. Se ha planteado una organización del desarrollo basada en el desarrollo de prototipos software en los que se han incorporado gradualmente.

La metodología que se va a emplear a lo largo del desarrollo del proyecto se basa en un modelo secuencial de fases:

1. Análisis y evaluación de las propuestas conceptuales (**formulación del problema**).
2. Requerimientos del sistema a obtener (**análisis**).
3. **Diseño** de la solución.
4. Realización de la solución/prototipo (**pruebas y validación**).
5. **Integración** de la solución con lo desarrollado en otros paquetes de trabajo.
6. **Validación** de los resultados: pruebas del prototipo.

ESTADO DEL ARTE

Este apartado forma parte del trabajo resultante del proyecto SmartLazarus por el cual es usado para conocer en qué estado se encuentran actualmente desarrollados los temas a tratar en este proyecto.

A nivel plataforma, el proyecto **SmartLazarus** se centra en el **desarrollo de un motor de localización y sensorización**. Este sistema se basa en el uso de diversas fuentes de datos heterogéneas para mejorar las prestaciones del sistema en función de las características del entorno. De esta forma, podrán emplearse las tecnologías más extendidas en materia de localización (tanto en exteriores como en interiores). Por otro lado, se busca obtener una plataforma basada en el paradigma ***Cloud Computing***, que permite ofrecer servicios para **maximizar las capacidades** de los sistemas software que hacen uso de ellos, facilitando en este caso la construcción de sistemas de sensorización y geolocalización a partir de múltiples fuentes. En el marco de este entregable se analizan ambas tecnologías con el objetivo de determinar las principales características de cada una, permitiendo así integrarlas en el marco del proyecto SmartLazarus.

Así, en la siguiente subsección (Localización), se analizan en profundidad las diferentes alternativas más relevantes para el desarrollo de sistemas de localización, tanto en interiores como en exteriores. A continuación, en la subsección *El modelo Cloud Computing* se presentan las principales características del paradigma Cloud Computing. Finalmente, en la subsección de conclusiones sobre el estado del arte se proponen los siguientes pasos de cara al diseño de la plataforma.

LOCALIZACIÓN

INTRODUCCIÓN

Este apartado tiene como objetivo realizar una clasificación general de los diferentes sistemas de localización que permiten conocer la posición precisa de un objeto móvil. En primer lugar, se debe introducir al lector en el mundo del posicionamiento de las redes inalámbricas explicando el significado del término "posicionamiento" y posterior a esta introducción, es necesario realizar una pequeña clasificación y explicación de uso. Una vez realizada la clasificación inicial se profundizará en el ámbito en el que se centra este trabajo, el funcionamiento de las redes inalámbricas y los sistemas de localización basados en ellos. Finalmente concluiremos este apartado analizando los sistemas de localización WiFi que existen en el mercado.

POSICIONAMIENTO

Igual que ocurre con el teléfono móvil, los sistemas de localización o también denominados de posicionamiento cada vez nos rodean más y empiezan a ser indispensables en nuestro entorno. La gran evolución que ha sufrido la tecnología duramente los últimos años ha permitido la creación de sistemas sencillos de localización, permitiendo conocer la ubicación de un objeto o persona en un área determinada o bien para la utilización conjunta con sistemas de gestión centralizados (control de flotas, seguimiento de usuarios, etc.). La utilización combinada de la localización precisa junto con la gestión centralizada nos permite ofrecer una multitud de servicios como, por ejemplo, conocer la parada de taxis más cercana, la localización de personas maltratadoras, la ubicación de un producto dentro de un centro comercial o incluso ofrecer algún tipo de guiado para encontrar algún establecimiento concreto desde el punto de origen.

El uso de los sistemas de localización tiene su origen en el ámbito militar, mediante sistemas denominados globales que utilizan la posición de los satélites como puntos de referencia. Actualmente los sistemas de localización se clasifican en dos grandes grupos:

- Posicionamiento vía satélite.
- Posicionamiento basado en redes (área local y redes metropolitanas)

POSICIONAMIENTO VÍA SATÉLITE

El posicionamiento vía satélite originó los primeros sistemas de localización, siendo el más importante y conocido el GPS (Global Positioning System). Fue originado por la empresa NAVSTAR en el año 1970

y tenía un uso militar que permitía la estimación de la posición y de la velocidad de un vehículo. Posteriormente surge el sistema ruso denominado GLONASS y el sistema chino llamado Beidou que presenta las mismas características que GPS. Actualmente y a pesar de su origen militar, han surgido diferentes aplicaciones civiles que han tenido gran aceptación en los últimos años, una de las más conocidas es Google Maps o Tom Tom Navigator.

Debido a que los sistemas GPS dependen de un sistema militar de Estados Unidos, la Unión Europea decidió que era imprescindible contar con un sistema de localización en exteriores propio, por ello, actualmente se encuentra en desarrollo el sistema Galileo.

GPS

La localización basada en GPS recibe el soporte de una red formada de un conjunto de ordenadores y una constelación de hasta 24 satélites que se encuentran orbitando por todo el globo para determinar la posición en tres dimensiones (la altitud, longitud, y latitud) de cualquier objeto cerca de la superficie terrestre. Para que un objeto pueda ser localizado debe localizar al menos 3 satélites, siendo mayor la precisión en función del número de satélites encontrados. De cada uno de los satélites encontrados se obtiene la posición del satélite emisor y el tiempo de envío de cada muestra recibida. Con este conjunto de datos recibido, el receptor puede conocer la posición absoluta dentro del globo terrestre mediante triangulación. El funcionamiento puede comprenderse más fácilmente mediante la Figura 1:

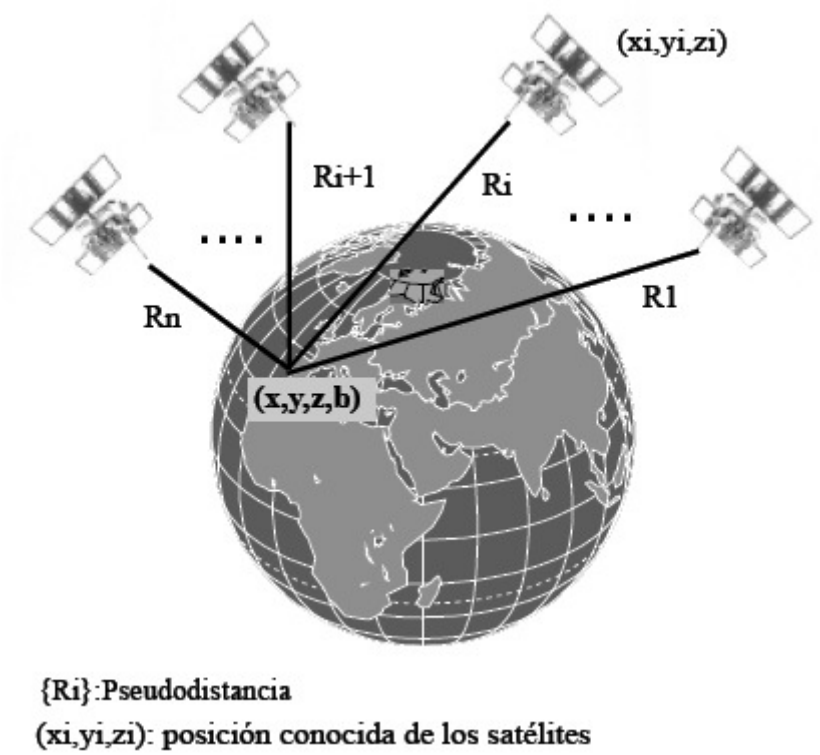

Figura 1 Funcionamiento localización GPS

El cálculo de la posición en el globo terrestre consiste en determinar los valores de x,y,z a partir de las distancias del punto con respecto de al menos 3 satélites. La localización de cada satélite es conocida, ya que es enviada al receptor terrestre. Por tanto, la distancia entre el usuario receptor GPS y un satélite es trivial (se multiplica el tiempo de vuelo de la señal emitida desde el satélite por su velocidad de propagación). Para medir el tiempo de vuelo de la señal de radio es necesario que los relojes de los satélites y de los receptores estén sincronizados, pues deben generar simultáneamente el mismo código. Ahora bien, mientras los relojes de los satélites son muy precisos, los de los receptores son osciladores de cuarzo de bajo coste y, por tanto, imprecisos. Las distancias con errores debidos al sincronismo se denominan pseudo distancias. La desviación en los relojes de los receptores añade una incógnita más que hace necesario un mínimo de cuatro satélites para estimar correctamente las posiciones.

El sistema GPS tiene un gran inconveniente: no funciona en interiores o en lugares donde la línea de visión no es directa, ya que la señal es muy débil o nula y no se logran localizar satélites suficientes para conocer una ubicación precisa. Además, el sistema GPS únicamente informa de la longitud, latitud y altitud, por ello es necesario un dispositivo (móvil, pda, etc.) que disponga de un software de mapas que te permita interpretar estos datos.

Otra característica importante que hace que el sistema GPS no sea apropiado para realizar un sistema de localización en interiores, es que dentro de un edificio no necesitamos la referencia absoluta, sino conocer la habitación en la que estamos o en qué esquina de la habitación nos encontramos, es decir, necesitamos una referencia relativa. Es más intuitivo conocer que estamos en el despacho de la persona "X" que nos encontramos en la posición del espacio x,y,z.

GALILEO

Es el sistema europeo de similares características al GPS, actualmente presenta 18 satélites operativos en órbita. Es un Sistema Global de Navegación por Satélite "GNSS" que persigue eliminar las dependencias del sistema americano "GPS" y ruso denominado "GLONASS" pero que será compatible simultáneamente con estos dos. El sistema Galieo fundamentalmente está diseñado para ofrecer cinco tipos de servicios:

1. Servicio abierto (Open Service – OS), servicio público gratis.
2. Servicio para aplicaciones críticas (Safety of Live -SoL)
3. Servicio público regulado (Public Regulated Service -PRS) para policía
4. Servicio de búsqueda y salvamento (Search and Rescue Service – SAR).
5. Servicio Comercial (Commercial Service-CS)

El sistema Galileo contempla la puesta en órbita de 30 satélites (inicialmente previsto para el año 2021) distribuidos en tres planos inclinados con un ángulo de 56º hacia el ecuador sobre unos 23.616 Km sobre la tierra. A diferencia de los satélites GPS, los satélites Galileo estarán inclinados hacia los polos para mejorar la precisión, ya que es precisamente en las zonas cercanas a los polos donde menos cobertura proporciona el sistema GPS:

POSICIONAMIENTO BASADO EN REDES

Todo sistema de posicionamiento basado en redes puede a su vez clasificarse en:

- Posicionamiento mediante redes móviles (CDMA,GSM…)
- Posicionamiento mediante redes inalámbricas

A continuación, se muestran las características más importantes de ambos grupos.

POSICIONAMIENTO MEDIANTE REDES MÓVILES

Se caracterizan por la propia estructura de las redes, no es necesario ningún hardware adicional, simplemente mediante el uso del teléfono móvil son capaces de estimar la posición del usuario

mediante la celda en la que se encuentra el usuario. Actualmente hay compañías telefónicas que ofrecen la localización vía móvil a sus abonados. La posición es una mera aproximación ya que es un sistema poco preciso. La falta de precisión de estos sistemas lo sitúan en clara desventaja respecto de otros, ya que no ofrecen una precisión por debajo de los 80metros, por lo que su uso se descarta en interiores. Los sistemas de posicionamiento basados en redes utilizan conjuntamente algoritmos de posicionamiento para ubicar al usuario, siendo los más importantes los siguientes:

- Diferencia del tiempo de llegada en el enlace ascendente. (U-TDOA): cálculo de la ubicación mediante la resta de los tiempos de llegada de la señal procedente del terminal móvil a distintas estaciones base.
- En función de la diferencia de tiempos de llegada perfeccionado (E-OTD): es uno de los algoritmos más utilizados en redes de tipo GSM y GPRS, es necesaria la instalación de software y hardware específico en los receptores. Los receptores realizan mediciones esporádicas de al menos tres estaciones base, estimando la posición mediante la diferencia de tiempo de la llegada de al menos 2 señales que, en conjunto, producen un campo hiperbólico e intersectarán en el lugar donde se encuentra el terminal.
- Dirección o ángulo de llegada (AOA, Angle Of Arrival): se estima en ángulo de la señal incidente mediante antenas multiarray situadas en la estación base. Si un terminal se encuentra emitiendo con visión directa, la antena multiarray puede determinar de qué dirección viene la señal.
- Identificación de celdas (CELL ID): es un tipo de identificación muy utilizado, ya que permite ubicar todo tipos de dispositivos móviles en redes GSM, GPRS, UMTS y CDMA. Su mecanismo es identificar la celda desde la que el móvil está obteniendo cobertura.
- GPS asistido (AGPS). recogen información de navegación y datos de corrección diferencial para los satélites GPS que están en la zona de cobertura del servidor de localización.

POSICIONAMIENTO MEDIANTE REDES INALÁMBRICAS

La característica principal para determinar el posicionamiento de un objeto o persona mediante el uso de las redes inalámbricas (WLAN) es medir la potencia de la señal recibida en el receptor por parte de los puntos de acceso (AP). Esta técnica es conocida como *location fingerprinting*, resultando muy útil en la mayoría de los escenarios de interiores en los cuales un sistema de posicionamiento global no es funcional. Además, la técnica *fingerprinting* puede utilizarse en exteriores, donde es posible obtener precisiones mayores, debido a que la señal de una red inalámbrica es mucho más potente que GPS. La comunicación en estas redes se fundamenta en el protocolo 802.11, que presenta un modelo de

comunicación centralizado. Una red consta de dos tipos de elementos, los puntos de acceso y un conjunto de clientes que se encuentran conectados a un AP. Los puntos de acceso se anuncian a los clientes mediante un paquete específico denominado *beacon,* que es enviado periódicamente para hacer notar su presencia, de este modo los clientes conocen las redes inalámbricas que están próximas a su entorno. Las redes basadas en los protocolos 802.11b y 802.11g son las más conocidas, emitiendo en uno de los 14 canales que están disponibles en Europa en la banda 2,4Ghz. La principal ventaja de la señal de este tipo de redes es la facilidad que presenta para atravesar muros, por ello se le da un uso de WLAN que permite ofrecer un servicio de acceso a internet en todo un edificio.

Las principales ventajas que tiene el uso de la tecnología inalámbrica son:

- Permite la movilidad de los usuarios.
- Flexibilidad en la ubicación de los diferentes puntos de acceso.
- Sistema escalable, requiere poca planificación, es muy sencillo extender el número de nodos, añadir nuevos puestos es inmediato.
- Es un sistema robusto y tiene como gran ventaja el poco mantenimiento, no es necesario mantener un cableado, evitando así los problemas asociados (cortes, conectores, etc.).

Por el contrario, las redes inalámbricas presentan una serie de inconvenientes frente a las redes cableadas, podemos destacar entre ellas:

- Menor seguridad, el no disponer de un medio guiado hace que las comunicaciones fluyan en un medio menos seguro.
- Las redes inalámbricas presenten velocidades inferiores a las redes cableadas con una tasa de error mayor.
- El alcance máximo está limitado a las características físicas del lugar, como paredes y muros.
- Las leyes gubernamentales restringen los espacios radio-eléctricos de emisión, por tanto, restringe la capacidad del sistema y hace que varios nodos coexistan en el mismo espacio radioeléctrico.
- La capacidad de retransmisión de un punto de acceso se comparte entre todos los usuarios de ese nodo, lo que limita el número de usuarios por nodo para ofrecer un servicio robusto y fiable.

Si evaluamos las características de la señal, podemos destacar las siguientes desventajas:

- Atenuación por distancia: a mayor distancia del foco emisor (AP), la potencia de la señal decrece con el tiempo de forma logarítmica, por lo que la señal recibida en las proximidades del emisor la potencia recibida decrece rápidamente, además, en las distancias medias la potencia decrece más lentamente. Si la distancia entre el emisor y receptor se duplica, la pérdida de potencia se multiplica por 4.
- Atenuaciones adicionales: si aparecen obstáculos entre emisor y receptor, la atenuación es mucho mayor que en espacio libre, esto es debido al fenómeno de la absorción. La absorción consiste en la colisión de una onda con un obstáculo, lo que provoca que éste absorba parte de su energía.
- Reflexión: cuando una señal colisiona contra un objeto, este absorbe parte de la energía de la onda como se ha mencionado anteriormente, sin embargo, si el obstáculo es demasiado grande comparado con la longitud de la onda de la señal, ésta se refleja del mismo modo que lo hace la luz al llegar a un espejo.
- Difracción: cuando la señal se encuentra con el borde de un obstáculo, como puede ser la esquina de una habitación, se forman frentes de ondas secundarios en todas las direcciones. Que exista una mayor o menos difracción dependerá de las características y materiales del obstáculo.
- Dispersión: La dispersión está causada por objetos que tienen dimensiones muy reducidas comparables con la longitud de la onda de forma que se forman frentes de onda secundarios que se unen a todos los efectos anteriores.

Los fenómenos de reflexión, difracción y dispersión componen el efecto del multitrayecto, es decir, no existe un único camino entre la antena transmisora y la antena receptora. Todos los frentes de onda que se forman son sumados o restados vectorialmente con otros, por ello, la señal que se recibe en el receptor no sólo proviene del camino más corto. Esto hace posible que en una zona donde no hay señal, si se produce un movimiento de muy pocos centímetros, la potencia de la señal recibida será muy grande.

LOCALIZACIÓN POR INFRARROJOS

La localización por infrarrojos no es apropiada para una localización en interiores, debido a dos factores. El primero de ellos es su corto alcance (aproximadamente dos metros) y porque, además, se requiere del uso de enlaces auxiliares con una línea de visión directa entre extremo y extremo. Debido a su corto alcance, sería indispensable el uso de un número muy elevado de emisores infrarrojos y aún así el problema de no tener visión directa desde todas las ubicaciones nos impediría detectar ciertas

posiciones. Se debe destacar la existencia de un proyecto muy conocido denominado Wireless Indoor Positioning System (WISP) que utiliza localización por infrarrojos para estimar la posición de un usuario.

WI-MAX

Se basa en el protocolo 802.16, con una tasa de transmisión de hasta 70Mbps, fue pensado para interconectar áreas muy extensas de aproximadamente 50Km. Debido al gran alcance que presenta esta tecnología, no se recomienda su uso para determinar posiciones.

BLUETOOTH

La tecnología bluetooth es muy popular y puede encontrarse en una gran cantidad de dispositivos electrónicos. Utiliza una banda de 2.4 GHz y está orientada hacia un uso en comunicaciones de un rango corto. Los estándares más extendidos de Bluetooth son Bluetooth 2 y Bluetooth 4, siendo este último el más apropiado para ser empleado en aplicaciones de localización (especialmente BLE). Uno de los métodos basados en Bluetooth más difundidos es la tecnología iBeacon de Apple (Newman, 2014). Se basa en el uso de transmisores de hardware BLE (balizas) que emiten su identificador. Las balizas pueden establecer la proximidad del usuario en tres regiones, en un rango de 50 cm a 30 metros.

UWB (ULTRA WIDEBAND)

Es una de las tecnologías más utilizadas para la localización en interiores, se caracteriza por presentar una tasa de transferencia muy elevada y una precisión de aproximadamente un metro. Es una tecnología candidata para favorecer la problemática existente en la poca precisión que ofrecen los demás sistemas, además ofrece una gran robustez frente a cambios del entorno (puertas, paredes, presencia de objetos o movimientos de personas). La principal desventaja que presenta esta tecnología es que presenta un consumo de batería alto para su uso.

ZIG-BEE

La tecnología ZigBee se basa en el estándar IEEE 802.15.4, se caracteriza porque sus dispositivos presentan un bajo consumo, una topología en malla y son muy fáciles de fabricar, ya que la electrónica presente en los dispositivos que usan esta tecnología es muy simple. Su principal ventaja es su bajo coste y su baja potencia de emisión, sin embargo, presenta como desventaja principal un ancho de banda muy reducido que hace que no se puedan ofrecer otros servicios como streaming, internet etc.

ALGORITMOS DE LOCALIZACIÓN

Existen tres algoritmos principales empleados por los sistemas de localización en tiempo real para determinar la ubicación de los nodos móviles (tags): Triangulación, Fingerprinting y Multilateración (Glassner, 1995). La triangulación permite obtener coordenadas de localización mediante el cálculo de la longitud de los lados de un triángulo a partir de los ángulos de entrada de la señal recibida en cada antena, para lo cual es necesario disponer de al menos 3 puntos de referencia. El Fingerprinting, también conocido como signpost o localización simbólica, se basa en el estudio de las características de cada zona de localización, realizando mediciones de las características de radiofrecuencia y estimando en qué área de influencia se encuentra cada tag(Capera, Georgé, & Gleizes, n.d.). Por último, la Multilateración se basa en la estimación de distancias desde los lectores a los tags midiendo parámetros como el RSSI (*Received Signal Strength Indication*) o el TDOA (*Time Difference Of Arrival*) (Tapia, Rodríguez, Bajo, & Corchado Rodríguez, 2008), de modo que, intersectando las distancias estimadas desde cada tag a tres o más nodos fijos, se pueden determinar los puntos en los que se encuentran dichos tags. La multilateración permite obtener mejores resultados en exteriores que con triangulación, pero su rendimiento baja notablemente en interiores. Este hecho se debe a que en espacios interiores los niveles de RSSI varían en función de la presencia de elementos (personas, objetos o animales) y, además, se basa en el cálculo de distancias, por lo que es necesario realizar una estimación previa de dichas distancias a partir de valores RSSI que cambian constantemente.

Las técnicas de localización basados en Triangulación y Multilateración no son eficientes puesto que las señales se ven muy atenuadas por los diferentes elementos de las habitaciones. Para estos casos, es más recomendable aplicar heurísticas aplicadas a procesos de clasificación y realizar entrenamientos para crear algoritmos de localización, ya que de esta forma se consigue mejorar la precisión final de los algoritmos. Las heurísticas permitirán recoger diferentes medidas en posiciones concretas y, a partir de dichas medidas, se puede calcular la posición más probable.

HEURÍSTICAS BASADAS EN CLASIFICADORES

Las técnicas de clasificación permiten asociar casos a grupos existentes. El comportamiento de estos algoritmos es similar a los existentes en clustering, pero mucho más sencillo en la mayoría de los casos. También, se pueden considerar como técnicas de clasificación a los árboles de decisión, reglas de decisión o a las redes bayesianas, puesto que permiten realizar la misma funcionalidad, por lo que se podría considerar su estudio en un mismo apartado. No obstante, se ha optado por separarlo, puesto que las técnicas de clasificación básicas no proporcionan la posibilidad de la extracción del conocimiento.

Las técnicas existentes se pueden agrupar en los siguientes, tipos en base a la naturaleza de los algoritmos empleados:

- Reglas de decisión y árboles de decisión.
- Modelos probabilísticos: Naive Bayes (Duda & Hart, 1973a), Redes bayesianas.
- Lógica borrosa: K-NN (K-Nearest Neighbours), NN (K-Nearest Neighbours).
- Definición de funciones: Sequential Minimal Optimization (SMO) (Platt, 1999).
- Redes neuronales.

A continuación, se describen de modo resumido las diferentes alternativas existentes en el proceso de clasificación.

REGLAS DE DECISIÓN

Son algoritmos que permiten obtener, a partir de una serie de grupos predefinidos, la información que permite clasificar un caso a un determinado *cluster*. Existen diversos tipos de reglas. Por un lado, los árboles de decisión, por otro lado, las reglas de decisión que generan modelos de reglas no jerárquicos. La principal diferencia entre ambos es que, en los árboles de decisión, se puede generar a partir de las reglas que los componen una representación del conocimiento en forma de árbol, puesto que los antecedentes de las reglas poseen elementos comunes, mientras que en las reglas de decisión tradicionales, los antecedentes no tienen por qué tener elementos comunes.

Dentro de las reglas de decisión, existen diversos algoritmos muchos de ellos están implementados en software de minería de datos como Weka. Entre los algoritmos existentes que han sido probados están RIPPER (Cohen, 1995), M5 (Hall Eibe Frank, Holmes, Pfahringer Peter Reutemann, & Witten, 2009), One-R (Holte, 1993) y PART (Frank & Witten, 1998). En algunas pruebas realizadas, se han comportado de manera muy similar a los árboles de decisión y han generado las mismas reglas para algún caso de estudio.

RIPPER se trata de un algoritmo con carga computacional en la búsqueda de los literales en cada una de las reglas, puesto que realiza una búsqueda en amplitud de todas las alternativas. El algoritmo se puede extender fácilmente con el simple hecho de incluir en la categoría negativos a todos los elementos que no pertenecen a la clase positiva.

ÁRBOLES DE DECISIÓN

Son un caso particular de las reglas de decisión que permiten, además, representar las reglas en forma de árbol (Quinlan, 1993). Se trata de una técnica bastante evolucionada y con una diversa variedad de

propuestas. Entre ellos es posible enumerar a CLS (Concept Learning System) (Hunt *et al.*, 1966), ID3 (Induction Decision Trees) (Quinlan, 1986), CART (Classication and Regression Trees) (Leo, Friedman, Olshen, & Stone, 1984) (Timofeev, 2004), OC1 (Oblique Classier 1) [Murthy, 1994], ASSISTANT (Cestnik, n.d.) o C4.5, C5.0/See5 (Quinlan, 1993), que han alcanzado una notable repercusión dentro del ámbito de la biomedicina.

Los árboles se componen de dos tipos de nodos: los nodos hoja y los nodos de test. Los nodos hoja se corresponden con un nodo final del árbol que contiene la información del grupo final en el que los casos son clasificados, mientras que los nodos de test contienen una regla lógica que determina el subárbol a seleccionar según el valor que tome. En caso de ser cierto, se selecciona la rama de la izquierda y, en caso de no cumplirse la condición, se selecciona la rama de la derecha. De este modo, los casos se van disgregando de acuerdo con el valor de los nodos de test aplicando la estrategia de divide y vencerás.

Un ejemplo de árbol de decisión se puede ver en la siguiente figura (Figura 2). Se puede ver que se parte de un nodo raíz y en cada uno de los nodos intermedios se tiene la regla de decisión. Finalmente, se tiene la clasificación final a la que se llega y el número de individuos clasificados de cada tipo. Este árbol se ha generado mediante CART (Classication and Regression Trees) (Leo et al., 1984).

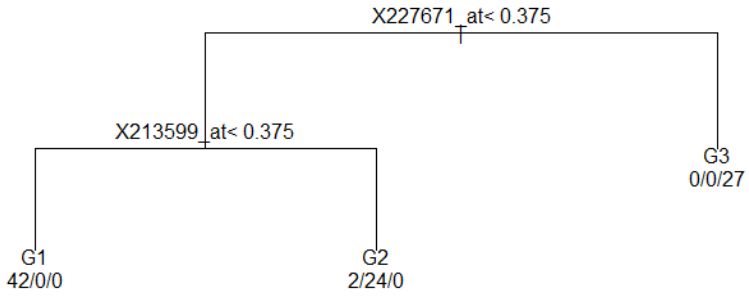

Figura 2 Árbol de decisión generado por CART

De modo general, los árboles de decisión se construyen a partir de la maximización de funciones que tienen en cuenta la tasa de acierto y fallo de los nodos clasificados en función de los atributos seleccionados y la condición establecida.

Los árboles de decisión son una técnica que permite extraer de forma sencilla las reglas que permiten explicar las clasificaciones de los individuos. Las reglas se pueden expresar mediante antecedentes y consecuentes o bien se puede representar la información en forma de árbol lo que facilita la comprensión. Las últimas variantes de los algoritmos permiten la inclusión de condiciones de decisión

más variables, permitiendo incorporar atributos de tipo numérico bien se tratante de valores continuos u ordinales.

MODELOS PROBABILÍSTICOS

Los modelos probabilísticos establecen la probabilidad de pertenencia de un caso a cada uno de los grupos. Se fundamentan en el estudio del valor de cada atributo del nuevo caso con respecto a la frecuencia que aparece en cada uno de los clusters existentes, de modo que se asigna una probabilidad de pertenencia de un elemento a cada uno de los grupos. Entre los clasificadores probabilísticos se tiene el clasificador de Naive Bayes o las redes bayesianas. Para aplicar estos métodos es necesario realizar una discretización de los valores continuos.

El clasificador de Naive Bayes permite realizar clasificaciones de datos basándose en la aplicación del teorema de Bayes (Duda & Hart, 1973b). El clasificador de Naive Bayes, permite asociar un determinado elemento a una clase en base a las diferentes características que presenta. No obstante, para realizar dicha tarea, se realiza la suposición de que las características son independientes entre sí.

De modo muy resumido, se estima la probabilidad de asignar un determinado caso a una determinada a cada una de las clases en función de los valores de los atributos. A partir de los valores obtenidos, se asigna a la clase que proporciona el valor máximo.

En diversas pruebas realizadas, cuando se trabaja con un número elevando de variables y un número bajo de casos, el clasificador presenta claras deficiencias en cuanto a lo genérico del modelo. Posee una tasa de acierto muy elevada para los casos con los que se ha entrenado el modelo, pero no es capaz de predecir de manera muy correcta casos previos.

Una red bayesiana es un modelo probabilístico multivariado que relaciona a un conjunto de variables aleatorias mediante un grafo dirigido acíclico que permite inferencia bayesiana para la estimación de probabilidades de variables no conocidas a partir de variables conocidas. Las redes bayesianas constan de los siguientes elementos:

- Un conjunto de nodos, uno por cada variable aleatoria.
- Un conjunto de arcos dirigidos que conectan los nodos.

Cada nodo contiene la distribución de probabilidad condicional que lo relaciona con el nodo padre.

Se fundamentan en teorema de Bayes, que permite calcular la probabilidad de un suceso a priori A_i sabiendo que ha ocurrido a posteriori el suceso B. Se define de la siguiente manera.

$$P(A_i \mid B) = \frac{P(B \mid A_i) \cdot P(A_i)}{P(B)} = \frac{P(B \mid A_i) \cdot P(A_i)}{\sum_{j=1}^{n} P(B \mid A_j) \cdot P(A_j)}$$

Donde:

$P(A_i)$ es la probabilidad de que ocurra el suceso A_i

$P(B \mid A_i)$ es la probabilidad de que ocurra el suceso B sabiendo que ha ocurrido A_i

$P(A_i \mid B)$ es la probabilidad de que haya ocurrido anteriormente el suceso A_i sabiendo que posteriormente se ha dado B

$P(B)$ es la probabilidad de ocurra el suceso B.

Los sucesos A_i son excluyentes

Inicialmente, las redes bayesianas se construían manualmente mediante el conocimiento de un experto. Actualmente, se han desarrollado procedimientos para realizar el proceso de modo automático o semiautomático. Para definir la red de Bayes (Duda & Hart, 1973b), es necesario establecer el conjunto de variables que forman parte del problema que se está tratando.

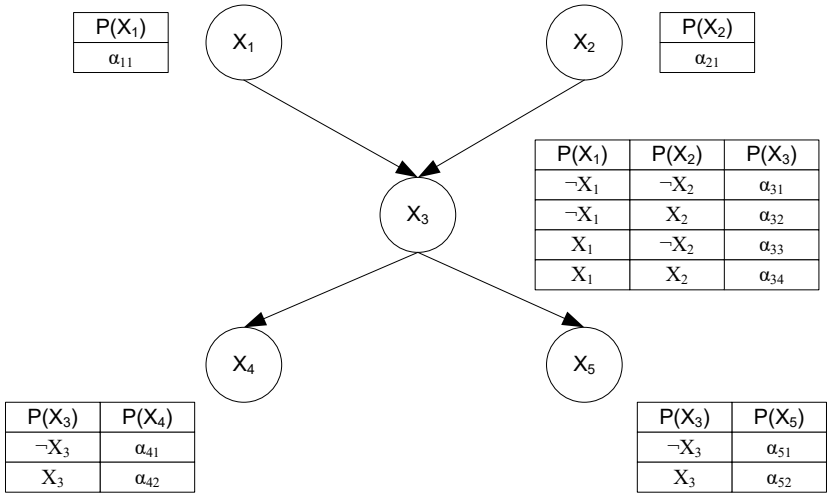

Figura 3 Red bayesiana

Siguiendo este procedimiento, se podría calcular la probabilidad de que ocurran las diferentes combinaciones de sucesos, consiguiendo así determinar la probabilidad de las diferentes combinaciones.

LÓGICA BORROSA

Las técnicas de lógica borrosa son posiblemente las más sencillas e inmediatas de usar. Se fundamentan en el uso de medidas de distancias o matrices de disimilaridad para determinar los casos más similares previamente clasificados. Una vez establecida una jerarquía de similitud con los casos previamente clasificados, se procede a recuperar un determinado número de individuos por orden de similitud. A partir de los individuos recuperados, se estudia la proporción existente de cada uno de los grupos y así se establece el grado de pertenencia del nuevo caso a cada uno de los grupos. Algunos de los algoritmos que siguen esta técnica son K-NN (K-Nearest Neighbours) o NN (Nearest Neighbours) (Aha, Kibler, & Albert, 1991).

DEFINICIÓN DE FUNCIONES

En la actualidad, es una alternativa muy extendida dentro de los clasificadores. Consiste en la definición de funciones que permiten separar las diferentes agrupaciones. El caso más simple es la definición de funciones lineales. El principal problema es que el uso de funciones lineales no permite separar cualquier tipo de problema, por lo que es necesario aplicar algoritmos que permitan transformar problemas no separables linealmente en problemas que sí lo son. Ante esta necesidad surge Support Vector Machine (SVM) (V. N. Vapnik, 1999).

Support Vector Machine (SVM) es una técnica de aprendizaje supervisado aplicada a la clasificación y regresión de elementos. SVM tiene aplicaciones en diversos campos como son la química, inteligencia ambiental, modelado y simulación, minería de datos o text mining. El algoritmo representa una extensión a los modelos no lineales (V. Vapnik, 1963), que inicialmente se desarrolló para la clasificación en problemas separables linealmente y básicamente consistía en encontrar la recta o hiperplano (en más de dos dimensiones) que permitiera separar a los elementos de un conjunto. SVM también permite separar clases de elementos que no son separables linealmente y para ello mapea el espacio de coordenadas inicial en un espacio de alta dimensionalidad mediante el uso de funciones. Debido a que la dimensionalidad del nuevo espacio puede ser muy alta, no es practicable el cálculo de hiperplanos que permiten realizar la separabilidad lineal, para ello, se usan una serie de funciones no lineales denominadas núcleos.

Si se considera que se tiene el conjunto de patrones marcados por $T = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$ donde x_i es un vector de dimensión n, la idea es convertir los elementos x_i en un espacio de alta dimensionalidad mediante la aplicación de una función $\Phi(x)$, de modo que el conjunto de patrones original se convierte en siguiente conjunto $\Phi(T) = \{(\Phi(x_1), y_1), (\Phi(x_2), y_2), ..., (\Phi(x_m), y_m)\}$, dependiendo de la función $\Phi(x)$ seleccionada, $\Phi(T)$ podría ser separable linealmente.

Para realizar la clasificación se sigue la ecuación siguiente. En el artículo (Lipkowitz, 2007) se puede ver el proceso de cálculo de la ecuación.

$$clase(x_k) = signo\left(\sum_{i=1}^{m} \lambda_i y_i \Phi(x_i)\Phi(x_k) + b\right)$$

Tal y como se puede ver, se tiene un producto $\Phi(x_i)\Phi(x_k)$ que según la dimensionalidad del nuevo espacio puede ser muy costoso de calcular, por lo que es necesario seleccionar una serie de funciones denominadas núcleo que permitan operar en el espacio original para poder realizar estos cálculos sin tener que requerir gran carga computacional. Bajo ciertas condiciones el producto en el espacio multidimensional tiene un resultado equivalente en el espacio de entrada, es decir se da que

$$k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$$

Para los casos k en los que se da esta condición, se dice que la función es una función núcleo. En el artículo (Lipkowitz, 2007) se muestran varias funciones núcleo como lineal, polinomial, gaussiana, exponencial..., por lo que existe gran variedad de funciones núcleo que permite realizar este cálculo de modo sencillo. Por ejemplo, para el caso de la función polinomial se tiene lo siguiente:

$$k(x_i, x_j) = (1 + x_i x_j)^d$$

Como se puede ver, para realizar el cálculo no es necesario tener en cuenta el nuevo espacio de alta dimensionalidad, por lo que las operaciones a realizar para la clasificación de individuos $\Phi(x_i)\Phi(x_k)$ se reduce a lo indicado en la parte derecha de la igualdad.

Para el cálculo del clasificador $clase(x_k)$ existen métodos como el Sequential Minimal Optimization (SMO) (Platt, 1999) que permiten calcular de modo eficiente la función clasificadora de modo

iterativo. Al permitir el cálculo de modo iterativo facilita la acotación temporal del algoritmo pudiendo acotar temporalmente la solución.

Para el cálculo del clasificador, existen métodos como el Sequential Minimal Optimization (SMO) (Platt, 1999) que permiten calcular de modo eficiente la función clasificadora de modo iterativo. Al permitir el cálculo de modo iterativo facilita la acotación temporal del algoritmo pudiendo acotar temporalmente la solución.

Implementaciones como SMO permiten realizar el proceso de modo iterativo facilitando la acotación temporal frente a otras técnicas como los árboles de decisión. Los árboles de decisión necesitan un formateo previo de los datos en caso de trabajar con muchas variables para que el funcionamiento sea eficiente. Es necesario convertir las variables de continuas a discretas y categorizarlas en niveles para que el funcionamiento sea eficiente.

EL MODELO CLOUD COMPUTING

INTRODUCCIÓN

Cloud Computing (CC) (Mell & Grance, 2011), entendido como paradigma computacional, está emergiendo en los últimos años con gran fuerza, hasta el punto de que está empezando a ser muy habitual en cualquier ámbito relacionado con la computación, más allá del contexto social de Internet. Empresas punteras en el ámbito tecnológico tales como IBM, Amazon, Microsoft, etc. apuestan decididamente por este modelo computacional en su modelo de negocio y oferta de servicios. Aunque a primera vista pueda ser considerado como un paradigma meramente tecnológico, la realidad demuestra que su rápida progresión está principalmente motivada por los intereses económicos que subyacen alrededor de las características puramente computacionales (Armbrust et al., 2010; Buyya, Yeo, Venugopal, Broberg, & Brandic, 2009).

Dada la complejidad que presenta los sistemas CC donde existen una gran variedad de tecnologías, componentes, sistemas, etc. que tienen que trabajar de forma conjunta.

EL PARADIGMA CLOUD COMPUTING

En términos históricos, el término CC fue utilizado por primera vez por el profesor Chellappa (Chellappa, 1997), quien ya avanzaba que el modelo de computación en el futuro estaría mucho más ligado a los intereses económicos que a las limitaciones impuestas por la tecnología. Aunque hace más de una década, en un contexto con continuos avances tecnológicos esta afirmación resultaba utópica,

hoy en día, la realidad demuestra que el desarrollo de la tecnológica está motivado por los intereses del mercado (Buyya, Yeo, & Venugopal, 2008).

El concepto fue haciéndose popular de la mano de Salesforce.com Inc., empresa que centraba su estrategia comercial en ofrecer Software como Servicio a grandes empresas (Weissman & Bobrowski, 2009). Su modelo de negocio era radicalmente diferente al del resto de empresas de la época. Así, Salesforce.com, en lugar de ofrecer paquetes software a medida, comercializaba el acceso a software genérico desplegado en sus propios servidores y accesible a través de Internet (Cusumano & Michael, 2010). Este nuevo modelo de negocio, pese a la innovación en su concepción, ya había sido enunciado previamente previamente por Carr (Carr, 2005).

Desde un punto de vista de investigación, fue IBM quien avanzó, a través del documento Autonomic Computing Manifesto (Horn, 2001) las directrices que guiarían el modelo computacional (auto-coonfiguración, auto-monitorización, auto-optimización, etc.). Partiendo de estas directrices, diferentes empresas (Yahoo , Amazon, eBay y Microsoft), también investigaron ampliamente en el concepto entre los años 2003 y 2007, con resultados muy dispares, donde la mayoría de aproximaciones estuvieron centradas en la capa de infraestructura (Ogrizović, Sviličić, & Tijan, 2010).

El desarrollo del paradigma CC tal y como hoy lo conocemos, tanto a nivel tecnológico como de comercialización, fue desarrollado por el consorcio formado en 2007 por Google e IBM junto con 6 prestigiosas universidades americanas: Massachusetts Institute of Technology (MIT), Standford University, University of California, University of Berkeley, University of Meryland y University of Washington (Lohr, 2007). Este esfuerzo, se debe a la ventaja competitiva que implica dominar este marco tecnológico, poseer una plataforma propia permite por un lado, (i) al proveedor Cloud ofrecer sus servicios siguiendo un modelo de pago por uso (Buyya et al., 2008), siguiendo los principios propuestos por el concepto Utility Computing. Y por otro lado, (ii) al usuario Cloud, le permite no tener la obligación de provisionarse con recursos computacionales adicionales para contener los picos en la demanda de sus productos o servicios, ni tampoco disponer de la infraestructura para proporcionarlos; transformando los gastos de capital en gastos operacionales (Armbrust et al., 2010).

TECNOLOGÍAS RELACIONADAS

No cabe duda de que el desarrollo de este paradigma computacional CC ha sido motivado por el rápido desarrollo de un conjunto de tecnologías sobre las que se sustenta para ofrecer el catálogo de servicios computacionales bajo demanda. A continuación se presentarán de forma resumida el conjunto de

tecnologías precedentes, así como su aportación individual al paradigma CC (Buyya et al., 2009; Wang et al., 2010; Q. Zhang, Cheng, & Boutaba, 2010):

- Grid Computing. Es un paradigma de computación distribuida cuyo objetivo es coordinar recursos computacionales y de red para satisfacer un objetivo (Q. Zhang et al., 2010). El desarrollo de este paradigma distribuido fue motivado para satisfacer las necesidades computacionales del software científico. Las características principales (Bote-Lorenzo, Dimitriadis, & Gómez-Sánchez, 2004) de este paradigma son la heterogeneidad, reparto de recursos, múltiples administradores, acceso consistente y transparente, y finalmente, distribución geográfica a gran escala. Todos estos rasgos identificadores están orientados a satisfacer los objetivos a nivel de aplicación, y también son compartidos por el paradigma Cloud. Sin embargo, CC avanza sobre estos principios para proporcionar un entorno hardware virtualizado a diferentes niveles (hardware y plataforma) gracias al cual se puede realizar un aprovisionamiento automático de recursos (pooling). Dentro de CC, se habla de Big Data en lugar de Grid Computing cuando se refiere al análisis de grandes cantidades de información.
- Utility computing. Es un modelo de aprovisionamiento de recursos computacionales (procesamiento y almacenamiento) en el que el proveedor del servicio suministra los recursos a medida que los usuarios hacen uso de los mismos (Ross & Westerman, 2004). Dentro del concepto es mucho más relevante los aspectos micro y macroeconómicos, que los tecnológicos en sí. Así, la literatura existente (Ross & Westerman, 2004), trata de justificar los beneficios, en términos de ahorro de costes, que obtiene una empresa cuando externaliza sus necesidades computacionales, refiriéndose habitualmente al centro de proceso de datos. Este modelo desde el punto de vista del mercado sigue una dirección paralela al paradigma CC, salvando la distancia temporal entre ambos. Pero, a nivel tecnológico, Utility Computing centra sus esfuerzos en el sector empresarial; mientras que CC abarca también al gran público debido a que no centra sus esfuerzos en la externacionalización del centro de proceso de datos, sino en cualquier tipo de servicio computacional, con independencia de la escala a la que vaya a ser suministrado. No se encuentran, en el estado del arte, plataformas que provean servicios computacionales siguiendo un modelo de utilidad a gran escala, por lo que se puede afirmar, que CC es la implementación a gran escala de los principios propuestos por Utility Computing.
- Autonomic computing (Horn, 2001; Parashar & Hariri, 2005). Aunque no es un término tan conocido como pueden ser los anteriores, resulta clave a nivel computacional dentro del paradigma que nos incumbe. Las directrices de esta tecnología determinaban que los sistemas

computacionales fueran capaces de auto-gestionarse y reaccionar al entorno (interno o externo) sin la necesidad de la intervención humana. Aunque esta tecnología sólo fue desarrolla a nivel teórico, en la actualidad no cabe duda de que ha sentado las bases sobre las que se han construido el paradigma computacional CC.

- Virtualization Technology (Anderson, Peterson, Shenker, & Turner, 2005). Es la tecnología que más ha favorecido al rápido desarrollo del paradigma CC. Permite crear abstracciones hardware de forma dinámica y automática en forma de máquinas virtuales sobre el hardware disponible (Barham et al., 2003). Además, los últimos avances permiten variar los recursos asignados a cada uno de los nodos virtualizados e incluso migrarlos entre diferentes equipos físicos sin tener que detenerlos. Así mismo también permite crear y destruir nodos de una red. Gracias a estas nuevas y potentes características, la virtualización facilita la gestión de los recursos hardware, tarea que en el pasado era altamente compleja, siendo un proceso manual y lento, que habitualmente requería personal altamente especializado. Destacar, en último lugar que gracias a que el usuario final, no tiene acceso a los recursos físicos, sino a la capa superior virtualizada, es posible presentarle una visión ilimitada de los recursos disponibles. Como única desventaja de esta tecnología, se observa que la gestión del entorno virtualizado requiere de un Hypervisor, que es el módulo que gestiona las abstracciones hardware en cada nodo físico para lo cuál consume recursos (McDougall & Anderson, 2010). No obstante, este consumo computacional depende del modelo de virtualización elegido. En la actualidad existen varios sistemas de virtualización que utilizan diferentes técnicas para minimizar el consumo de recursos, destacando Xen, KVM, VMWARE, OpenVZ, etc.
- High Availability Computing. El nacimiento de la tecnología de virtualización y la organización de los centros de procesos de datos en clústers de tipo HPC, ha propiciado el desarrollo de nuevas técnicas de alta disponibilidad (Petrucci, Loques, & Mossé, 2010), que van mucho más allá de las vetustas técnicas basadas en la replicación de elementos hardware (Gray & Siewiorek, 1991). CC toma los principios de esta tecnología en tanto en cuanto los servicios computacionales ofrecidos se tienen que ofrecer sin interrupciones. Sin embargo, CC avanza en este sentido, ya que los parámetros de calidad de los servicios también deben ser constantes con independencia en la demanda de los mismos (Andrzejak, Kondo, & Yi, 2010), asociados a un acuerdo calidad de servicio (SLA)
- Service Oriented Architecture, Service Flow y Workflow. Como ya se ha comentado, el nacimiento de los servicios web (Staab et al., 2003) y las arquitecturas orientadas a servicios (SOA) (Newcomer & Lomow, 2004) han contribuido al desarrollo del paradigma CC,

especialmente en la capa de los servicios de plataforma, en la que se exponen un conjunto API (Application Programming Interface) a los programadores cuya funcionalidad es muy heterogénea.

Finalmente, el desarrollo de la Web 2.0, Internet del Futuro (Anderson et al., 2005) y las nuevas técnicas de programación web enriquecida (Fraternali, Rossi, & Sánchez-Figueroa, 2010) han contribuido también en gran medida al notable crecimiento del paradigma CC.

CLOUD COMPUTING HOY EN DÍA: DEFINICIÓN

El concepto CC se ha ido afianzando tanto en el plano empresarial, como en el de la investigación. Por lo que han ido surgiendo una gran variedad de definiciones (Mell & Grance, 2011)(Vaquero, Rodero-Merino, Caceres, & Lindner, 2008)(Foster, Zhao, Raicu, & Lu, 2008) que se presentan a continuación. En cada una de ellas, los autores tratan de resaltar aquellas características que a su juicio son más relevantes. A finales del año 2011, el NIST (National Institute of Standards and Technology) americano (Mell & Grance, 2011), propone la definición que a nuestro juicio es la más acertada desde un punto de vista técnico y funcional (aunque en ella no se presta especial atención a la economía de escala que ha promovido el desarrollo incesante de este modelo computacional):

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

En este caso los autores, además de referirse a cualquier tipo de recursos computacionales, indican un concepto clave: la reducción al mínimo el esfuerzo de obtener los servicios proporcionados; algo que, sin duda, ha sido clave en el gran auge que ha tenido esta tecnología.

Como complemento, esta definición incluye 5 características obligatorias que cualquier entorno CC debería poseer:

- Servicios automáticos bajo demanda, refiriéndose a que los servicios, con independencia de su tipo, se tienen que proporcionar automáticamente y sin interacción humana en función de la demanda de los usuarios. Estas características de simplicidad y automaticidad, ya la habían apuntado numerosos autores de forma previa, como se ha visto a lo largo de este documento.

Pero como crítica, no se habla de la necesidad de alcanzar un acuerdo en la calidad del servicio, lo que quizás constituye en sí mismo una de las mayores debilidades en la definición.

- Disponibilidad de servicios a través de la red, los clientes deben acceder a los servicios a través de la red y, por lo tanto, los proveedores tienen que utilizar este medio para proporcionar sus servicios.
- Disponibilidad de recursos, el proveedor debe disponer de la capacidad de ofrecer los servicios con independencia de la demanda de éstos, utilizando recursos hardware físicos o virtuales asignados dinámicamente a cada servicio y reasignados en función de la demanda. Esta característica implica que el usuario no conoce dónde está su información en cada momento, ya que existe un principio de no localidad (o deslocalización), donde los datos se trasladan entre los recursos virtuales (o no) que tienen asignados en cada momento. En este sentido, existen autores como (Buyya et al., 2009; Q. Zhang et al., 2010) que hablan directamente de servicios de alta disponibilidad, siguiendo un modelo de computación de alta disponibilidad.
- Elasticidad. Los diferentes recursos se deben proporcionar de forma elástica e incluso automáticamente en función de la demanda. Esta característica está directamente ligada con la anterior, la alta disponibilidad de los servicios, lo que implica que el proveedor debe asignar los recursos en función de la demanda de los servicios, algo que ya se venía haciendo en tecnologías anteriores. Sin embargo, CC ofrece servicios elásticos, esto significa, que los recursos asignados a cada servicio varían automáticamente en función de la demanda, lo que implica proporcionar más recursos cuando más demanda existe, pero también reducir recursos cuando disminuye la demanda. Todo este proceso se debería realizar de forma automática sin que el cliente de los servicios tenga conocimiento de la readaptación interna. Gracias a esta característica, los usuarios tienen la sensación de interactuar ante un sistema con recursos ilimitados, aunque en realidad no sea así. La existencia de niveles de acuerdo en cuanto a la calidad del servicio introduce una complejidad mayor en la gestión de la elasticidad, ya que no sólo es necesario proporcionar mayores recursos a más demanda en los servicios, sino que el tiempo necesario para la readaptación está limitado en base a la calidad acordada.
- Servicios a medida. Los servicios proporcionados por un sistema CC deben estar totalmente monitorizados, y su control tiene que realizarse de forma automática. Gracias a ello, se puede optimizar el uso de recursos; en este sentido, sólo teniendo un conocimiento total sobre el propio sistema, es posible ofrecer un servicio de pago por uso eficiente y competitivo.

Permitiendo ofrecer una transparencia total de la información entre el proveedor del servicio y el usuario del mismo.

Estas características, junto a la definición previa, acotan en gran medida el concepto que hoy se tiene de un entorno de CC y de los servicios que éste ofrece, los cuáles se conocen habitualmente como Cloud Services (Sosinsky, 2011). No obstante, casi parece lógico, teniendo en cuenta que la última versión de la definición es relativamente reciente. Lo que ha provocado que se hayan ido eliminando los vestigios de tecnologías predecesoras a CC; aunque en algunos casos, estén relacionadas con éste nuevo paradigma (Utility computing, Service Computing, Data Centers, Market-oriented computing, etc.), en otros casos el concepto se ha ido alejando (Grid Computing, P2P Computing, etc.).

Sin embargo, como debilidades en la definición, en cuanto a las características hardware, destacar que, por un lado, el NIST menciona la elasticidad dinámica, incluyendo como rasgo opcional la automatización de esta característica. Desde nuestro punto de vista, y el de algunos autores (Q. Zhang et al., 2010), esta es una de las características clave del entorno CC, siendo objetivo de estudio a lo largo de este trabajo. Y por otro lado, según (F. Zhang, Chen, Chen, & Zang, 2011; Q. Zhang et al., 2010) también hablan del principio de “multi-tenencia” (multitenant), el cual se refiere a que pueden existir varios entornos CC en un mismo centro de proceso de datos, lo que obliga a una clara división de responsabilidad en la gestión del entorno.

La definición del NIST sin duda es la más acertada y completa, aunque podría completarse incluyendo explícitamente el concepto de SLA, algo que sin duda ayudaría a completar todos los aspectos comerciales y negocios del paradigma (Buyya et al., 2008); dado que además existe una gran cantidad de estudios centrados en este campo. Finalmente, la definición y características de CC propuestas por el NIST, pueden complementarse mediante el conjunto de características propuestas por Zhang et al.: Multi-tenancy, Shared resource pooling, Geo-distribution and ubiquitous network Access, service oriented, dynamic resource provisioning, self-organizing y Utility-based pricing (Q. Zhang et al., 2010). Dentro de estas características, cabe destacar la última, que es una propiedad estrechamente relacionadas con los aspectos económicos que envuelven al paradigma. El precio del servicio está basado en su utilidad, es decir, dado que CC emplea un modelo de pago por uso, el precio concreto de un servicio, depende del propio servicio y del uso que el cliente hace de este, basándose en la teoría clásica de la regulación (oferta y demanda) (Peltzman, 1976). Lo que a priori logra reducir el coste operacional debido a:

- No existe inversión inicial, ya que, al implementar un modelo de negocio de pago por uso, desde el punto de vista del usuario de servicios CC no es necesario realizar una inversión previa en infraestructura, sino que se alquilan un conjunto de servicios acordes a las necesidades en cada momento.
- Reducción de los riesgos comerciales y los costes de mantenimiento. Dado que siguiendo el modelo CC, los recursos computacionales no se encuentran ligados a la empresa, los costes derivados de su mantenimiento, actualización, fallos, desconexión por actualización, etc. no constituyen un riesgo empresarial para la empresa.

A lo largo de este apartado se han presentado diferentes definiciones sobre CC que denotan la evolución del paradigma tecnológico desde su nacimiento en 2007. En primer lugar, destaca el interés que existe tanto en la comunidad científica, como en el entorno empresarial acerca de este paradigma computacional. Así mismo, también es reseñable la estrecha relación que existe con tecnologías previas, así como que la unión de todas ellas en una plataforma única, lo que da como resultado un entorno tecnológico mejorado. Finalmente, la clave en el desarrollo y la rápida introducción en el mercado ha sido el modelo de comercialización basados en acuerdos específicos de uso, que resultan favorables tanto para los proveedores, como para los consumidores del servicio.

En el modelo de negocio, las tecnologías implicadas y los servicios ofertados, hacen que un sistema CC sea complejo y en algunos aspectos difícil de comprender. Por lo que en los siguientes apartados se presentará con más detalle los tipos de servicios ofertados, los modelos de despliegue posibles y roles de usuario implicados en el modelo de comercialización.

CAPACIDADES OFERTADAS, SOMETHING AS A SERVICE

Como se indicó anteriormente, en la definición del NIST (Mell & Grance, 2011) proponen tres tipos de servicios y cuatro modelos de despliegue, ampliamente conocidos. Se comenzará describiendo los modelos de servicio, en este sentido dado que se ofrece casi cualquier tipo de servicio computacional, ya sea hardware o software y siempre a través de Internet, es habitual hablar de Something as a Service. No obstante, antes de enumerar estos servicios, resulta clave indicar que en la propia definición se habla de capabilities o capacidades, en lugar de servicios. Según el NIST, los principales modelos de servicios ofrecidos son los siguientes:

- SaaS. Software as a Service. Este tipo de capacidad permite al proveedor, ofrecer al consumidor sus aplicaciones, de forma que éstas se ejecutan directamente en la infraestructura en nube. Lo que conlleva una gran cantidad de ventajas como son la ubicuidad

de las aplicaciones o el uso de clientes ligeros. Sin embargo, también lleva asociadas un conjunto de dificultades directamente relacionadas con que el hecho de que el consumidor pierde el control sobre la infraestructura (red, almacenamiento, sistema operativo, dificultad de configuración, etc.), aunque quizás desde un punto de vista más moderno, pueden ser incluso consideradas como fortalezas.

- PaaS. Platform as a Service. Este tipo de capacidades proporcionadas por los proveedores, permiten al consumidor disponer de las herramientas necesarias para crear sus propias aplicaciones. Entre estos servicios se encuentran entornos de programación, librerías, herramientas, etc. De la misma manera que los servicios de la capa anterior, el programador no controla la infraestructura subyacente, ni el entorno de despliegue de sus aplicaciones.
- IaaS. Infrastructure as a Service (o Hardware as a Service –HaaS– para (Wang et al., 2010)). El tipo de capacidad que se proporciona al consumidor es de tipo hardware, es decir, procesamiento, almacenamiento, red, etc. Al igual que en servicios anteriores, el consumidor o usuario no tiene el control sobre la infraestructura subyacente, en este caso, sistemas operativos nativos, características de red, etc. Aunque sí que puede tener la configuración sobre el entorno hardware virtualizado que se le presenta.

Finalmente, con respecto a las capacidades proporcionadas, añadir que algunos autores (Lenk, Klems, Nimis, Tai, & Sandholm, 2009) introducen el concepto de Human as a Service (HuaaS), dentro de esta categorización de modelos de servicios que ofrecen las plataformas CC. Dentro esta categoría se enmarcan los servicios de recolección de información de forma masiva procedente de las bien conocidas plataformas sociales (Facebook, Twitter, etc.). Esta información es utilizada, principalmente, como complemento de otros servicios.

TIPOS DE USUARIOS Y MODELOS DE DESPLIEGUE

De forma previa a enumerar los entornos de despliegue, resulta conveniente hacer referencia a los actores y roles que intervienen en el modelo de negocio CC. Así pues, en este apartado se presentarán en primer lugar los diferentes roles que intervienen en el paradigma CC, teniendo en cuenta las diferentes perspectivas existentes. En la literatura, es posible encontrar varios trabajos sobre los roles o tipos de usuarios de un entorno CC (Armbrust et al., 2010; Buyya et al., 2009; Grance, Patt-Corner, Voas, & Badger, 2012; Q. Zhang et al., 2010).

Por un lado, a nivel técnico, siguiendo a [Vouk, 2008] se definen cuatro categorías de actores no excluyentes:

- Service User. Son los consumidores de servicios del entorno CC, su rol es el más importante y del que dependen el resto de los usuarios.
- Service Integration & Provisioning. Expertos en composición de servicios, cuyo objetivo es satisfacer la demanda de un cliente concreto.
- Service Author. Responsables del desarrollo de servicios que puedan ser utilizados de forma individual, o como parte de otros.
- Cyberinfrastructure developer. Responsables del desarrollo y mantenimiento de la infraestructura.

Por otro lado, existe otra clasificación de roles que constituye un estándar, está más aceptada y constituye un estándar de facto. En ella, no sólo se tiene en cuenta la perspectiva técnica del despliegue hardware y software, sino también los intereses económicos que motivan el desarrollo del paradigma. Dentro de estos roles, se distinguen los siguientes, no siendo excluyentes ninguno de ellos:

Cloud provider. Es el rol que proporciona capacidades Cloud. Normalmente, dentro de esta categoría están los servicios de tipo IaaS y PaaS. Fue propuesto inicialmente por [Armbrust et al., 2010]. A su vez puede subdividirse en:

- Infrastructure Providers. Identifica al rol que proporciona servicios hardware, es decir, IaaS. Propuesto por (Vaquero et al., 2008).
- Service Provider. Rol que proporciona servicios de tipo software en el modelo Cloud. Puede convertirse de forma simultánea en Cloud User si a su vez utiliza otros servicios de tipo SaaS, mediante la composición de servicios a este nivel. Propuesto por (Q. Zhang et al., 2010), aunque Armbrust et al. lo denomina SaaS Provider (Armbrust et al., 2010).
- Cloud User. Este rol hace uso de las capacidades ofrecidas por el rol Cloud provider, con referencia a los niveles PaaS e IaaS. Fue propuesto por (Armbrust et al., 2010), denominado User/Broker en (Buyya et al., 2009).
- SaaS User/End User. Es el usuario final de las aplicaciones SaaS.

Además, (Buyya et al., 2009) propone la existencia de un actor adicional, SLA Resource Allocator, entidad intermedia entre el entorno Cloud y los usuarios externos, con las funciones de monitorización de peticiones, control de admisión, gestión de precios, gestión de cuentas y monitor de infraestructura.

En el marco de investigación del NIST, se propone una división mucho más sencilla, en 6 actores que modelan las interacciones tanto a nivel de mercado, como técnico, integrando por tanto las dos

perspectivas anteriores. Así pues, por un lado, se proponen tres actores desde un punto de vista de mercado: (i) customer, que es un usuario u organización que utiliza las capacidades que ofrece un entorno CC; (ii) client, que es una agente artificial que accede al entorno CC para utilizar y desplegar sus servicios y ofrecer productos al anterior rol; y finalmente (iii) provider, que es una organización que proporciona las capacidades CC. Por otro lado, otros tres actores que modelan la relación entre el consumidor y el proveedor desde un punto de vista técnico: (iv) Cloud broker que es el intermediario que gestiona la relación comercial entre proveedor y consumidor; (v) Cloud Carrier, que proporciona la conexión desde el proveedor y el consumidor; (vi) Cloud Auditor, que evalúa de forma independiente los servicios ofrecidos en cuanto a rendimiento y seguridad.

Una vez que se han presentado los diferentes roles, siguiendo con la definición propuesta por el NIST, y por lo que componen un entorno CC, el último paso consiste en identificar los diferentes modelos de despliegue que existen, comenzando por los públicos y privados:

- Private Cloud. La infraestructura CC es utilizada por una única organización, que a su vez puede incluir diferentes consumidores. La infraestructura puede pertenecer a la propia organización, o ser ofrecida por un tercero.
- Public Cloud. Este tipo de infraestructuras se proporcionan para el uso abierto del público en general. Están alojadas por organizaciones, universidades, gobiernos o una combinación de ellos. Obligatoriamente debe existir el actor proveedor.

A partir de estos dos modelos de despliegue propuestos por el NIST, (Q. Zhang et al., 2010)(Grance et al., 2012) proponen la existencia de un modelo híbrido denominado Virtual Private Cloud, también denominado Outsources Private Cloud por (Liu et al., 2011) en el que se combinan entornos CC públicos y privados. De forma que se construye un entorno Cloud privado sobre las capacidades de infraestructura a modo de entorno subyacente que proporciona un entorno public Cloud. (Q. Zhang et al., 2010) define este tipo de despliegue como una plataforma situada por encima de los sistemas CC públicos, siendo la principal diferencia el hecho de que se permite el diseño de una topología específica y un modelo de seguridad específico, aunque en contrapartida el entorno CC esté deslocalizado.

Como consecuencia de la existencia de Virtual Private Cloud, se puede dar la situación de que varios entornos CC, que aún perteneciendo a organizaciones diferentes tengan que compartir la misma infraestructura, es decir Cloud público. En este caso se estaría hablando de multitenencia (Bezemer, Zaidman, Platzbeecker, Hurkmans, & Hart, 2010; Mell & Grance, 2011).

Finalmente, el NIST, también define otros dos modelos de despliegue, el de comunidad y el híbrido:

- Community cloud. La infraestructura del sistema CC es utilizada por un grupo de consumidores u organizaciones específicos que comparten intereses relacionados. La infraestructura puede estar alojada por una o varias organizaciones dentro de los grupos de interés (on-site Comunity Cloud), o incluso, puede pertenecer a un tercero (out-source comunity cloud).
- Hybrid Cloud. Este tipo de infraestructuras es la combinación de alguna de las anteriores, cada una de las cuáles tiene una entidad propia, pero trabajando de forma coordinada. La Figura 4 muestra un ejemplo de un sistema CC híbrido que se ha construido mediante la agregación de 5 variantes.

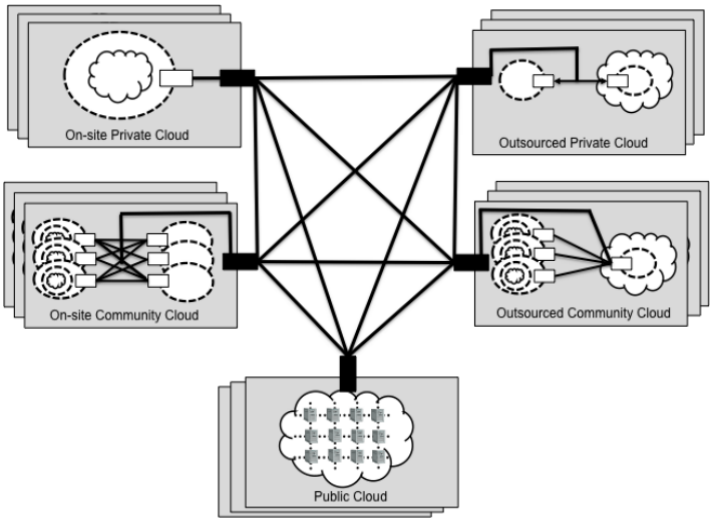

Figura 4 Cloud híbrido [Liu et al., 2011].

Cuando se habla de Hybrid clouds es necesario hablar de interoperabilidad entre plataformas (Ortiz, 2011), algo que también puede hacerse extensible al resto de despliegues CC. Un usuario de este tipo de entornos habitualmente trabaja con diferentes plataformas al mismo tiempo, cada una de ellas tiene su propia interfaz de comunicación, habitualmente en forma de API, lo que las hace cerradas y dificulta su interoperabilidad con el resto de plataformas.

MODELOS DE REFERENCIA: PLATAFORMAS Y ARQUITECTURAS

No cabe duda de que un sistema CC es complejo y forma parte de un entorno abierto en el que se conjugan diferentes tecnologías, usuarios e intereses económicos para dar lugar a un nuevo modelo computacional que ha revolucionado la forma de ofertar servicios a través de Internet. Para que todos estos aspectos trabajen de forma coordinada con el objetivo de lograr unos objetivos comunes, y

además teniendo que tener presentes las limitaciones técnicas existentes, se han desarrollado arquitecturas complejas.

Hasta hace no mucho tiempo las grandes compañías como IBM, Intel, Google, Amazon, etc. no mostraban información relevante sobre las arquitecturas de sus respectivos modelos CC. De hecho hoy en día el ocultismo en este aspecto sigue siendo la tónica general ya que, aunque muchas de ellas han hecho públicas las directrices generales, en ningún caso se han liberado algoritmos u otros aspectos técnicos o computacionales que permitan el avance de la tecnología sobre la base existente. Así por ejemplo, las Figura 5 y Figura 6 presentan la pila tecnológica de sistemas CC propuestas por Intel (Chahal et al., 2010) y Cisco (Strategy, 2009), respectivamente. En ellas se pueden observar los componentes básicos de cualquier sistema CC como infraestructura subyacente (servidores, equipamiento de red, almacenamiento, etc.), así como componentes software (entorno de virtualización, gestión, monitorización, seguridad, etc.). Sin embargo, ninguna de estas compañías es un gran proveedor de servicios CC, por lo que ambas utilizan esta información para ofrecer sus productos a los verdaderos proveedores CC, los cuales no publican sus arquitecturas CC.

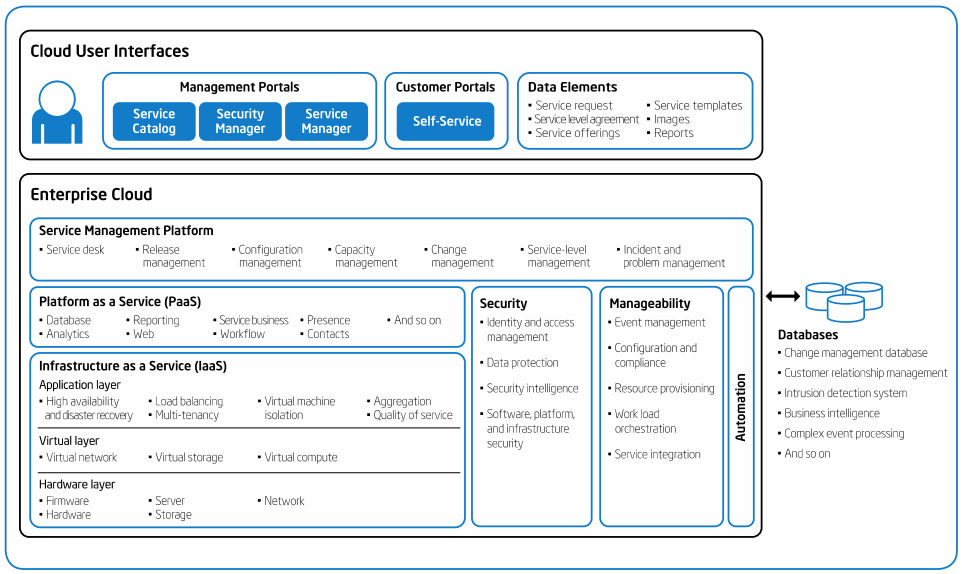

Figura 5 Arquitectura Intel [Chahal et al., 2010]

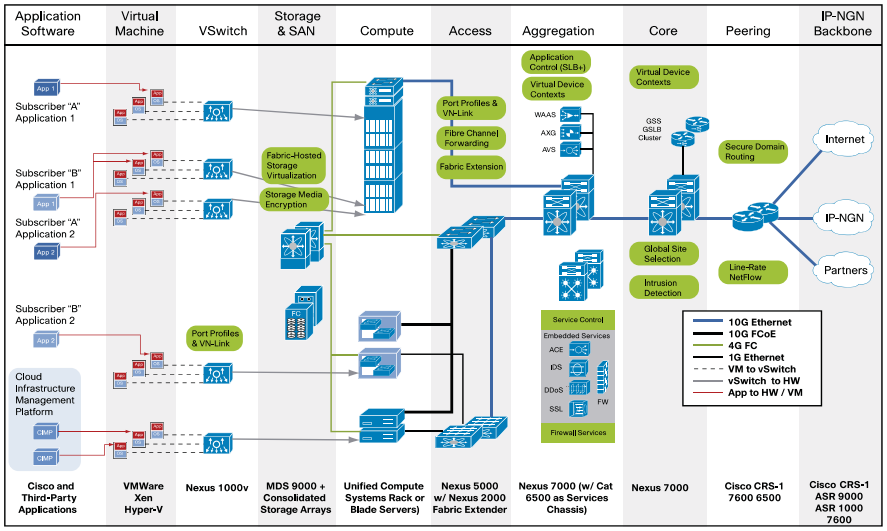

Figura 6 Modelo de Arquitectura para Cisco [Cisco, 2009]

Siguiendo el trabajo realizado por el NIST, este organismo también propone una arquitectura de referencia (Liu et al., 2011), que trata de describir y acotar correctamente todos los aspectos a tener en cuenta en un entorno CC. Esta arquitectura se presentará a continuación. No obstante, resulta interesante presentar de forma previa la clasificación que propone Tianfield (Tianfield, 2011) respecto a las arquitecturas CC:

- La arquitectura de plataforma, es aquella que tiene en cuenta el entorno tecnológico subyacente al paradigma CC
- La arquitectura de aplicación, es aquella que encapsula la gestión de la calidad de los servicios que se le ofrecen a los usuarios finales.

El modelo de referencia de arquitectura CC propuesto por el NIST (Liu et al., 2011) aúna ambos modelos (según (Tianfield, 2011)), es decir, tecnología de plataforma y nivel de calidad hacia el usuario final, tal y como se muestra en la Figura 7.

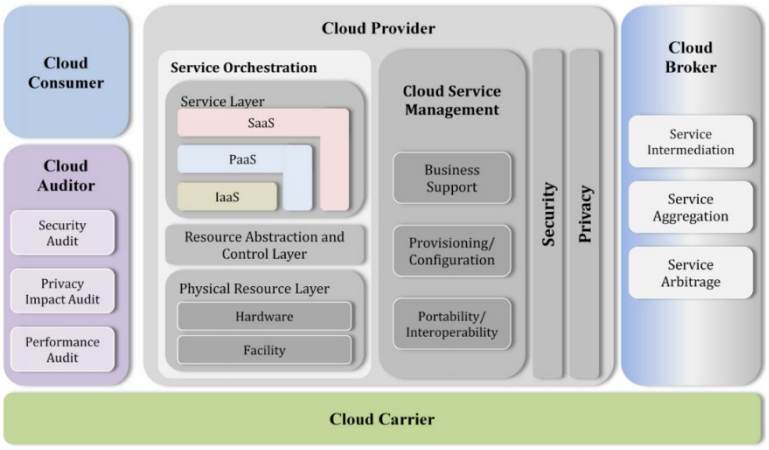

Figura 7 Arquitectura Cloud de referencia para el NIST [Liu et al., 2011]

La arquitectura que se presenta en la Figura 6 tiene en cuenta no sólo los componentes tecnológicos, sino también los roles que participan dentro del paradigma CC. Así, en función de cada uno de los roles del sistema, tenemos los siguientes módulos o componentes:

- **Cloud Provider**. Es el rol encargado de la (i) Coordinación de Servicios, proporcionando éstos a terceros, con independencia de su tipo (IaaS, PaaS o SaaS), por lo que necesita gestionar la infraestructura subyacente (física y virtual) a través de una capa de control. Así mismo, también debe (ii) facilitar la gestión de esos servicios ofertados a través de una capa de soporte a la comercialización y negocio que facilite la monitorización acerca del uso de esos servicios (características de calidad y acuerdos en el nivel del servicio) y su cobro en función del uso que se realice (facturación, consumos, etc.), así mismo también deberá proporcionar características de portabilidad e interoperabilidad entre plataformas, y un modelo que permita la rápida configuración y la gestión del cambio de aprovisionamiento de los recursos. Finalmente, también tendrá que hacerse cargo de la (iii) seguridad y (iv) privacidad, que son características clave para que un Cloud Provider sea aceptado por los consumidores de sus servicios.
- **Cloud Auditor**. Es un agente externo que capaz de monitorizar el servicio con el objetivo de validar si se están cumpliendo los requisitos acordados mediante SLA de seguridad, privacidad y rendimiento.
- **Cloud Broker**. Es un agente externo que actúa como intermediario entre consumidores y proveedores con el objetivo de buscar y proporcionar los servicios más adecuados a los objetivos de los consumidores con independencia del proveedor. Sus labores principales son la (i) intermediación, encargada de buscar y encontrar los mejores servicios y validar que efectivamente se están cumpliendo los valores de calidad acordados; (ii) agregación,

proporcionando la capacidad de integrar servicios de diferentes entornos CC; y, finalmente (iii) arbitraje, proporcionando un entorno similar a la agregación de servicios, pero en el que estos servicios no están predefinidos y cambian dinámicamente.

- **Cloud Carrier**. Que es el agente que proporciona conectividad entre el proveedor y el consumidor. Resulta importante, ya que muchos de los objetivos acordados en el SLA dependen de que el Cloud Carrier sea capaz de proveer con la tecnología para transportar la información con la suficiente rapidez.

No obstante, más allá de esta arquitectura de referencia también es posible estudiar las arquitecturas de las plataformas abiertas CC, cuya información es pública. Así pues, hoy en día existe una gran variedad de plataformas CC abiertas, distribuidas bajo la licencia Open Source y como no podía ser de otro modo también existen diferentes trabajos que comparan estas plataformas (Wen, Gu, Li, Gao, & Zhang, 2012)(von Laszewski, Diaz, Wang, & Fox, 2012).

A continuación se presentarán aquellas plataformas CC, que a nuestro juicio son las más importantes y que además son las más utilizadas, es decir, OpenStack , OpenNebula y, en menor medida, Eucalyptus . No cabe duda de que existen otras plataformas, a las que a priori se les asigna el apellido cloud, como por ejemplo OpenQRM, Nimbus, CloudStack, Cloud Fondry, Open Monbster, etc. Sin embargo, después de analizar estas plataformas no están diseñadas para cubrir un espectro generalista, o bien su único interés es el de facilitar la gestión de la capa hardware física y virtual, tarea que la mayoría de los sistemas de virtualización modernos ya incorpora como tarea por defecto.

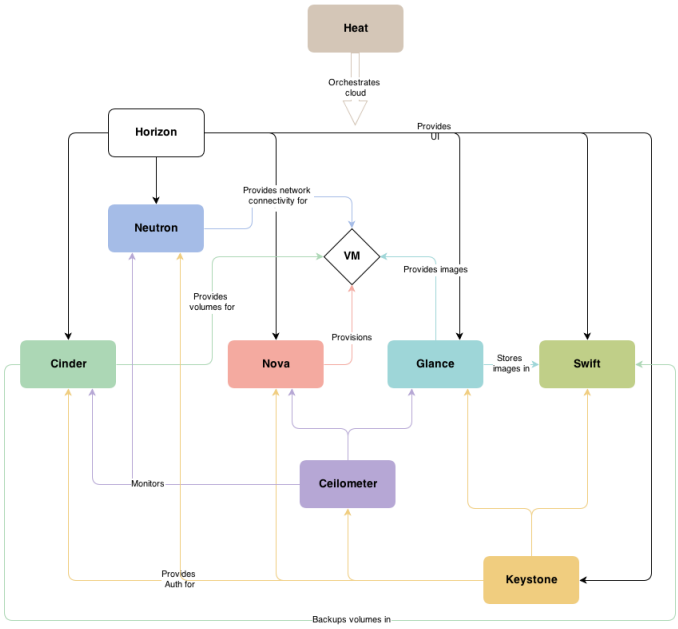

Figura 8 Arquitectura conceptual de OpenStack

En primer lugar, OpenStack apareció en escena en torno a Julio de 2010, y fue convirtiéndose en popular debido a que grandes empresas fueron aceptando su modelo (Citrix , Dell , Suse , Telefónica , etc.), hoy en día agrupa a más de 13.500 personas en 132 países. Es un conjunto de proyectos de código abierto que pueden ser utilizados para configurar un entorno CC a medida.

En la Figura 8 se presentan los principales componentes de esta arquitectura (Sefraoui, Aissaoui, & Eleuldj, 2012). Éstos son el servicio de identidad (Keystone), computación que incluye diferentes subservicios de almacenamiento, red, planificador y comunicaciones (Nova), almacenamiento (Swift), imágenes (Glance) y el servicio de configuración (Dashboard). Openstack es abierta y cada uno de estos servicios puede desplegarse en cualquier servidor, ya que están desacoplados entre sí, comunicándose a través de una cola de mensajes basada en el protocolo Advanced Message Queuing Protocol (AMQP), no obstante, el componente NOVA que gestiona los recursos, debe estar disponible en todos servidores, o al menos en sus componentes básicos.

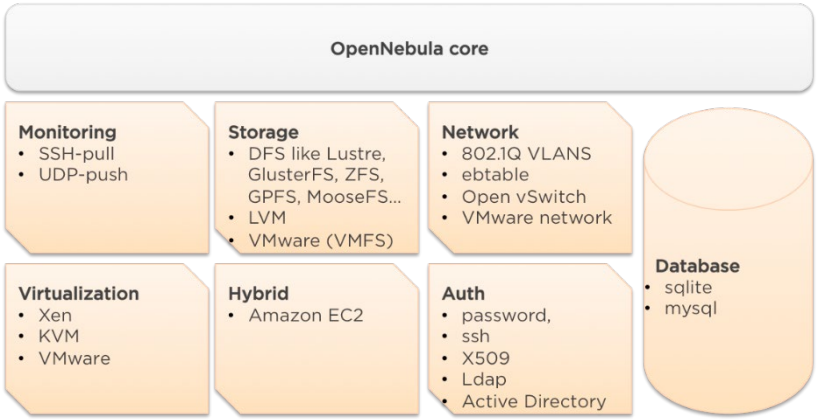

Figura 9 Componentes OpenNebula

Por su parte, OpenNebula (Milojičić, Llorente, & Montero, 2011) fue la primera versión de un sistema CC abierto, su desarrollo empezó en 2005 y su primera versión estuvo disponible en 2008. Es un proyecto mucho más extendido que Openstack y es utilizado por una gran cantidad de instituciones y empresas para construir su propio entorno CC. OpenNebula está pensado para su uso en grandes centros de datos de forma que pueda transformar esos recursos en un entorno CC para ofrecer servicios de infraestructura. Su arquitectura (Figura 9) es flexible y modular, permitiendo integrar una gran cantidad de almacenamiento, red y entornos de virtualización.

Finalmente, Eucalyptus (Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems) fue lanzada inicialmente en el año 2008. Al igual que las anteriores, su objetivo es la transformación de un centro de proceso de datos tradicional en otro basado en CC con la capacidad

de ofrecer servicios de tipo infraestructura. Sus componentes principales son un el NC (Node controller), situado en cada máquina física, el SC (Storage controller) y el CLC (Cloud Controller) que es único en cada entorno de despliegue, adicionalmente dispone del módulo Walrus que actúa de puerta de entrada de las peticiones del entorno CC. Su principal característica es la compatibilidad con Amazon EC2 y S3, permitiendo incluso un modo de funcionamiento mixto, con gestión simultánea de máquinas virtuales en la instalación local y en EC2.

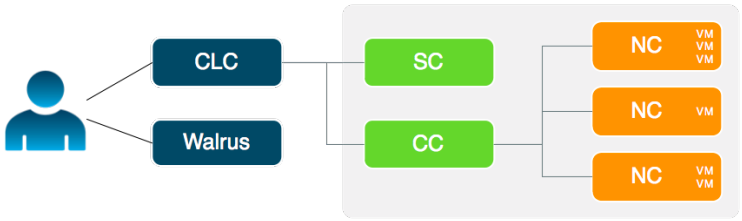

Figura 10 Arquitectura de referencia en Eucalyptus

Aunque Eucalyptus, OpenNebula y OpenStack (así como el resto de las plataformas abiertas) comienzan a ser muy utilizadas, la realidad demuestra que su desarrollo todavía está lejos de las características deseables de una plataforma CC, ya que en su amplia mayoría se centran en la gestión de la infraestructura, sin tener en cuenta al usuario final. Una arquitectura moderna debería tener en cuenta la calidad de los servicios que ofrece (QoS) y los acuerdos (SLA) alcanzados con los usuarios. Sin embargo, estas arquitecturas abiertas están más centradas en la gestión de la compleja infraestructura subyacente, es decir, en las capas inferiores, que en los servicios que se les ofrecen a los usuarios finales.

Finalmente, en la Tabla 1 se presenta una relación de otras plataformas CC existentes en la actualidad.

Tabla 1 Plataformas Cloud Computing

Prov.	Niv.	Empr.	Lic.	Plataf.	Descripción
vSphere	IaaS	VMWare	Comer.	N/A	Sistema operativo orientado a la formación de infraestructuras CC a través de virtualización.
Cloud Foundry	PaaS	VMWare	Libre	vSphere EC2	Capa PaaS para *vSphere*, que puede ser desplegada en infraestructuras de otros proveedores.
CloudStack	IaaS	Citrix	Libre	VMWare, Oracle VM, KVM, Xen	Capa de gestión de infraestructura que abstrae la plataforma tecnológica de virtualización subyacente.

AppScale	PaaS	Comun.	Libre	KVM, XEN, EC2, Eucalyptus	Ejecución en CC privado de aplicaciones de *Google App Engine*.
Nimbus	IaaS	Comun.	Libre	KVM, Xen	Gestión de infraestructura.
OpenQRM	IaaS	Comun.	Libre	KVM, Xen, VMWare	Gestión de infraestructura de centros de proceso de datos.

HERRAMIENTAS CLOUD DE AUTOMATIZACIÓN Y DESPLIEGUE

Existe una clara tendencia actual de la utilización de contenedores en los procesos de desarrollo y despliegue de software. Los contenedores suponen un mecanismo de empaquetado lógico que permite a las aplicaciones desarrolladas abstraerse de los entornos en los que éstas se ejecuten. Los contenedores encapsulan en una única imagen tanto archivos de configuración, binarios como ficheros adicionales que se puedan necesitar. De esta forma, se proporciona una mayor facilidad a la hora de desplegar las aplicaciones independientemente de los entornos finales (privados, nubes públicas, ordenadores personales…).

Una de las principales ventajas que aporta el uso de contenedores durante los procesos de desarrollo y despliegue es que permite una clara división de tareas y responsabilidades entre desarrolladores y equipos de TI, ya que abstraen a ambos equipos de cuestiones como las versiones específicas de software o las configuraciones de las aplicaciones. Docker (Docker, n.d.) es uno de los sistemas más extendidos que permiten la construcción, transferencia, despliegue y ejecución de los contenedores en los que se encapsulan las aplicaciones de forma ágil, sencilla y segura, asegurando un correcto despliegue con independencia de las características del sistema final. Sin embargo, una de las principales debilidades de Docker es su dificultad a la hora de escalar una infraestructura distribuida de un gran número de contenedores que se ejecutan simultáneamente en un gran número de servidores. Ante esta dificultad, surgen los sistemas conocidos como orquestadores, que permiten la gestión del cluster de servidores. Dentro de los principales de orquestadores con licencia OpenSource existen tres alternativas claramente destacadas: Kubernete (Kubernetes, n.d.)s, Mesos (Mesos, n.d.) y Swarm (Swarm, n.d.).

Swarm es la alternativa más sencilla, basada en una arquitectura de controlador (Managers) y nodos (Workers), pero que muestra una principal desventaja: no permite manejar posibles fallas de los nodos, por lo que es un inconveniente para servicios de alta disponibilidad. Por su parte, mesos cuenta con un kernel para cluster que corren en cada nodo y ofrecen plataformas como Hadoop o ejecución de contenedores Docker, Rkt y AppC y está basada en una configuración Master-Slave, en la que el nodo Master gestiona los recursos disponibles que se asignarán finalmente en función de la

disponibilidad del cluster. Por último, Kubernets es la solución más compleja y completa de las tres. Diseñado inicialmente por Google, permite el despliegue en diferentes entornos cloud y soporta el uso de runtimes de contenedores como Docker, rkt, cri-o o frakti.

Gracias a ser el sistema de orquestación más completo, Kubernetes podríamos decir que se está convirtiendo en el estándar de facto. Las principales características y funcionalidades que aporta el uso de Kubernates son:

- Escalado y autoescalado: de manera manual o automática según el uso computacional de las máquinas.
- Descubrimiento de servicios y balanceo de carga.
- Autorreparación: pudiendo reiniciar, reemplazar, replanificar o parar contenedores en función de los diferentes fallos que puedan surgir.
- Despliegues y rollbacks automáticos: despliegue progresivo de aplicaciones que permite la realización de rollbacks en caso de error.
- Planificación: planificación de la carga de trabajo en función a los recursos y restricciones.
- Gestión de la configuración y secrets: garantizando la confidencialidad de la información sensible, como contraseñas o claves ssh.
- Orquestación del almacenamiento: posibilidad de montar sistemas de almacenamiento locales, de proveedores de cloud pública o almacenamiento en red.
- Ejecución Batch: permite la gestión de las cargas de trabajo batch y CI, reemplazando los contenedores que fallen.

RIESGOS Y VULNERABILIDADES

Tal y como se ha presentado, el paradigma computacional CC ha crecido con fuerza en los últimos años y su desarrollo ha motivado el avance de una gran cantidad de plataformas, tanto públicas, como privadas. También es relevante el cambio que ha producido en el modelo de negocio tradicional, así como la externalización de la información empresarial a los grandes centros de datos.

Sin embargo, todavía existen una serie de retos que la tecnología tiene que superar para que el uso de este paradigma computacional se extienda a todos los niveles. Estos retos existentes son principalmente los riesgos relacionados con la seguridad, entre las que se destacan las siguientes vulnerabilidades (Subashini & Kavitha, 2011): problemas de acceso, riesgos de seguridad en la

tecnología de virtualización, vulnerabilidades en la tecnología web (SQL injection, cross-site scripting, seguridad física, IP spoofing, etc.) Así mismo, también existen dificultades debido a la pérdida del control de la información que se produce al alojar las bases de datos y el software de forma externa en grandes centros de datos, que son una caja negra para la empresa que hace uso de servicios. Entre estas vulnerabilidades destaca (Subashini & Kavitha, 2011)(Hamdi, 2012): la privacidad y confidencialidad de los datos, la pérdida y robo de información, problemas de autenticación, cumplimiento con las leyes de protección de datos y alta disponibilidad.

Diferentes autores han tratado de clasificar y acotar los problemas abiertos (Subashini & Kavitha, 2011)(Bhadauria & Sanyal, 2012). Estos estudios han propuestos diferentes taxonomías para organizar las vulnerabilidades, principalmente, atendiendo respectivamente (i) al tipo final de servicios en el que se produce (SaaS, PaaS o IaaS); (ii) la tecnología a la que se asocia el problema de seguridad; y (iii) finalmente, una clasificación atendiendo al tipo de despliegue (Público, Privado y/o de Comunidad).

Para analizar los problemas de seguridad, siguiendo el trabajo de (Grobauer, Walloschek, & Stocker, 2011), en primer lugar hay que destacar que CC está basado en la integración de diferentes tecnologías subyacentes que trabajan de forma coordinada. Así, resulta necesario distinguir entre los problemas de seguridad intrínsecos a cada una de esas tecnologías y los problemas derivados de su uso de forma conjunta en un entorno CC. En este sentido es posible referirse a las vulnerabilidades de las tecnologías asociadas a un entorno CC:

- Aplicaciones web y servicios. Donde se enmarcan las vulnerabilidades relativas a las capas PaaS y SaaS de la pila de servicios del paradigma. Destacan vulnerabilidades relacionadas con la gestión de los datos (Subashini & Kavitha, 2011) (localidad, integridad, segregación, control de acceso, confidencialidad, disponibilidad, violación de la información y la gestión de copia de seguridad), así como las vulnerabilidades clásicas de la tecnología Web (Subashini & Kavitha, 2011)(Cross-site scripting, inyecciones OS, SQL o LDAP, manipulación de cookies, problemas de configuración, vulnerabilidades en la capa TCP, ataques Captcha, etc.). No obstante, la mayoría de estas vulnerabilidades no están relacionadas con la capa de aplicación, sino que en torno al 60% están relacionadas con los niveles inferiores de la pila de tecnologías (sistema operativo y vulnerabilidades conocidas y no conocidas) (Bhadauria & Sanyal, 2012).
- Protocolos de Comunicaciones. Dado que la información es accedida de forma ubicua a través de diferentes redes, habitualmente Internet, utilizando para ello protocolos estándares de redes. Todas las vulnerabilidades asociadas a estos protocolos también constituyen riesgos de seguridad para el entorno CC. Entre estos riesgos destacan (Bhadauria & Sanyal, 2012;

Subashini & Kavitha, 2011) el análisis de paquetes y la violación de redes, la gestión débil de sesiones, las peticiones SSL inseguras, ataques DNS, sniffers, reutilización de direcciones IP y Prefix hijacking. Todas estas vulnerabilidades pueden ser utilizadas para realizar ataques de tipo Man-In-The-Middle lo que puede llegar a comprometer la seguridad de los datos y de la infraestructura que subyace.

- Persistencia de la información. Como no podía ser de otro modo, también existen problemas relacionados con las bases de datos utilizadas en los entornos CC, que habitualmente son bases de datos orientadas a documentos [Han et al., 2011]. Los problemas habituales están realizados por el uso de protocolo de encriptación de la información obsoletos (Grobauer et al., 2011), los cuales están ampliamente estudiados y las soluciones propuestas ofrecen un nivel de seguridad aceptable (Kamara & Lauter, 2010). También existen problemas de confidencialidad, en cuanto a la verificación de la identidad (Subashini & Kavitha, 2011) y, fundamentalmente, debido a que los datos de diferentes empresas (Cloud users) son almacenados en el mismo centro de proceso de datos, lo que puede provocar que puedan tener acceso a información confidencial de otros usuarios. En este sentido, el rol Cloud provider debe asegurar la confidencialidad de los datos entre sus clientes, pero además implementar medidas de seguridad adicionales para impedir el acceso de sus propios empleados a datos confidenciales de terceros. Finalmente, se observan problemas relativos a la necesidad de disponibilidad de Internet para acceder a la información o los problemas de latencia derivados del uso de redes como medio de transmisión.
- Gestión de la infraestructura. El uso de una capa de virtualización tiene innumerables ventajas a la hora de gestionar de forma dinámica la infraestructura. Sin embargo, también tiene más problemas de seguridad debido a que los usuarios potencialmente malintencionados comparten un mismo medio, e incluso un mismo servidor físico. Aunque la virtualización en sí misma puede ser considerada como una técnica de seguridad debido al aislamiento que proporciona. Los principales riesgos se enmarcan en la posibilidad de romper este aislamiento y conceder acceso a la máquina física, lo que por ende, da acceso al núcleo del entorno CC (Subashini & Kavitha, 2011). En este sentido, existen amplios estudio sobre las posibles vulnerabilidades y las soluciones a las mismas (Pal, Khatua, Chaki, & Sanyal, 2011). Finalmente, en cuanto a los riesgos en la capa de virtualización, teniendo en cuenta que un entorno CC ofrece servidores virtualizados en el nivel IaaS, los proveedores de servicios tienen un nuevo reto, ya que deben implementar un modelo de seguridad compartida, donde el Cloud Provider debe asegurar la seguridad del entorno CC a nivel físico y el Cloud User debe ocuparse de la

seguridad de la máquina virtual que adquiere como servicio (Grobauer et al., 2011). El reto reside en que el Cloud provider debe asumir problemas de seguridad dentro de su propia infraestructura debido a una gestión incorrecta de la configuración por parte del Cloud user.

En cuanto a los problemas de seguridad inherentes propios al paradigma, cabe destacar, en primer lugar las (i) Vulnerabilidades en la monitorización, ya que un entorno CC es aquel que tiene que autogestionarse, constituyendo un paradigma a medio camino entre Utility Computing (Ross & Westerman, 2004) y Autonomic Computing (Horn, 2001), es indudable que debe recoger datos sobre el uso a lo largo de toda su infraestructura (hardware y software). Con esta información, se evalúa el uso que cada usuario realiza del entorno CC y por lo tanto cuál es el coste asociado. Un problema de seguridad en esta monitorización, puede provocar la pérdida de los rendimientos derivados del alquiler de infraestructura o servicios a terceros (Grobauer et al., 2011). En segundo lugar, la (ii) dificultad para la gestión de la identidad, que debe seguir un modelo de tipo Single sing-on (De Clercq, 2002) y que es uno de los retos de este tipo de plataformas, no sólo por el hecho de asegurar la autenticación de usuarios en diferentes aplicaciones desplegadas en entornos geográficamente distribuidos, sino por el hecho de un almacenamiento compartido de las credenciales de seguridad de cada usuario. Las soluciones a estos problemas de seguridad son (Subashini & Kavitha, 2011) sistemas de gestión de la identidad independientes entre los diferentes Cloud users, sincronización de credenciales de seguridad a nivel SaaS, y la federación del modelo de gestión de la identidad. Y, finalmente, la (iii) Interoperabilidad entre plataformas, que como ya se ha comentado es un campo de investigación en la actualidad, pero que además dificulta en gran medida su implantación a gran escala. Para una empresa deslocalizar su información en el centro de procesos de datos de un tercero es un reto en sí mismo, pero además existe el problema de lock-in, que consiste en que una empresa no está dispuesta a depender únicamente de un solo proveedor debido a la dificultad de conexión entre plataformas.

Tal y como se ha presentado, los entornos CC como cualquier sistema de información y especialmente aquellos que son distribuidos, tiene un conjunto de riesgos y vulnerabilidades asociadas no sólo a las tecnologías subyacentes, sino al propio paradigma en sí mismo. Existen investigaciones activas y estudios sobre cómo asegurar la seguridad e interoperabilidad de las diferentes plataformas CC. La organización Cloud Security Alliance (CSA) publica una guía sobre buenas prácticas en materia de seguridad en el marco de los entornos Cloud (CSA, 2011). Aunque actualmente existen trabajos en todas las áreas vulnerables anteriormente señaladas, cuyo estudio se escapa de este trabajo, destacar que la gestión de seguridad depende en gran medida del modelo de despliegue del entorno CC

(público o privado) (Bhadauria & Sanyal, 2012) ya que el tipo de usuarios y servicios que implementa son totalmente opuestos.

CONCLUSIONES DEL ESTADO DEL ARTE

A lo largo de este apartado se han analizado las principales tecnologías a nivel plataforma que se emplearan en el proyecto SmartLazarus. A partir de este análisis del estado del arte, resulta necesario hacer un balance que guíe los siguientes pasos del desarrollo del proyecto.

En primer lugar, en tanto en cuanto a los sistemas de localización, dadas las características propias de la plataforma a desarrollar, se trabajará en el diseño de una plataforma que permita integrar soluciones de localización tanto en espacios exteriores como interiores. A lo largo de las siguientes tareas se definirán las técnicas y algoritmos más apropiados para cada situación, teniendo como base los trabajos ya existentes que han sido analizados en este documento.

En segundo lugar, en cuanto al entorno Cloud Computing, no se realizará una investigación en tanto en el desarrollo de este tipo de plataformas y se optará por el uso de plataformas ya existentes.

En definitiva, la combinación en el marco del proyecto de la tecnología de localización y el paradigma Cloud Computing permitirá modelar una plataforma inteligente que cumpla con los requisitos de la propuesta inicial.

HARDWARE UTILIZADO

Este apartado se utiliza para explicar cuál ha sido el hardware escogido y utilizado a partir de los requerimientos del proyecto.

TECNOLOGÍA BLUETOOTH

La tecnología Bluetooth es un protocolo de comunicaciones que permite la transmisión inalámbrica de datos entre diferentes dispositivos que se hallan a una distancia cercana, es decir, dentro del radio de alcance. Esta transmisión de datos se realiza de forma inalámbrica a través de ondas de radio que operan en la banda ISM (bandas de radio reservadas para fines industriales, científicos y médicos distintos a las telecomunicaciones) de los 2.4 GHz, para ello, hace uso de Redes Inalámbricas de Área Personal (WPAN). Puesto que la transferencia se hace por radiofrecuencia, los dispositivos no deben hallarse alineados, aunque sí deben encontrarse dentro del radio de alcance que varía según el dispositivo de forma que se clasifican como se explica a continuación:

- **Dispositivos de Clase 1:** tienen una potencia máxima permitida de 100mW y, por tanto, un alcance de 100 metros.
- **Dispositivos de Clase 2:** tienen una potencia máxima permitida de 2.5 mW y, por tanto, un alcance de entre 5 y 10 metros. Son los más habituales.
- **Dispositivos de Clase 3:** tienen una potencia máxima permitida de 1mW y, por tanto, un alcance de tan sólo 1 metro.

Esta tecnología apareció en 1994 y ha ido evolucionando a través de distintos estándares hasta la actualidad:

- **Bluetooth 1.0 y 1.0b:** estándar con dificultades de interoperabilidad, además, era obligatorio incluir la dirección del dispositivo, por tanto, no había anonimato.
- **Bluetooth 1.1:** introdujo la corrección de errores presentes en el estándar anterior, además, añadió soporte para canales no cifrados, es decir, no había necesidad de incluir la dirección del dispositivo.
- **Bluetooth 1.2:** proporcionó la posibilidad de una conexión y velocidad de transmisión más rápida. Por otra parte, mejoró la resistencia a las interferencias en las ondas de radio
- **Bluetooth 2.0:** se introdujo una mayor velocidad de transferencia de datos.
- **Bluetooth 2.1:** mejoró el emparejamiento entre dos dispositivos y aumentó la seguridad de la tecnología.
- **Bluetooth 3.0:** incrementó la velocidad de transferencia de datos hasta los 24 Mbit/s.

- **Bluetooth 4.0:** reúne el bluetooth clásico, el de alta velocidad y protocolos de bajo consumo.
- **Bluetooth 5.0:** transmisión de datos eficiente y menor consumo.

Las posibilidades que ofrece esta tecnología son numerosas por su facilidad en la transmisión de datos:

- Transmisión de datos.
- Conexión Inalámbrica entre distintos dispositivos.
- Acceso a contenidos específicos en áreas públicas.
- Conexión inalámbrica entre un dispositivo e internet.

Por estas posibilidades es que apareció su viabilidad en el uso de sistemas de localización. Dichos sistemas son redes de dispositivos utilizados para localizar inalámbricamente objetos o personas en espacios interiores. Dichas redes suelen ser redes WSN. Comparativa del Bluetooth con otras tecnologías inalámbricas:

- **Bluetooth:**
  - Precisión en sistemas de posicionamiento: variable según el dispositivo.
  - Consumo energético: Bajo.
  - Fiabilidad: Algo sensible a obstáculos e interferencias.
  - Velocidad de envío de datos: más de 2 Mbps.
  - Alcanzabilidad: 15m a 100m.
  - Seguridad: Difícil de hackear.
  - Latencia: Alrededor de 2 segundos para obtener posiciones (x, y, z).
  - Escalabilidad: Cientos o miles de nodos.
  - Coste (infraestructura, nodos, mantenimiento): Bajo.
  - Uso: Interior y exterior.
  - Banda de frecuencia: 2.4GHz.

- **GPS:**
  - Precisión en sistemas de posicionamiento: 5 a 20 metros.
  - Consumo energético: Alto.
  - Fiabilidad: Muy sensible contra obstáculos.
  - Alcanzabilidad: Ilimitado (considerando en la Tierra).
  - Latencia: 100 milisegundos para obtener posiciones (x, y, z).
  - Escalabilidad: Ilimitado.

- Coste (infraestructura, nodos, mantenimiento): Alto.
- Uso: Exterior.

- **RFID:**
  - Precisión en sistemas de posicionamiento: Desde centímetros a un metro.
  - Consumo energético: Es pasivo, no necesita batería.
  - Fiabilidad: Alto.
  - Alcanzabilidad: 1m a 5m.
  - Seguridad: Puede ser falsificado con relay attack.
  - Latencia: 1 segundo para obtener posiciones (x, y, z).
  - Escalabilidad: Ilimitado.
  - Coste (infraestructura, nodos, mantenimiento): Alto.
  - Uso: Interior y exterior.

- **UWB:**
  - Precisión en sistemas de posicionamiento: Centímetros.
  - Consumo energético: Alto.
  - Fiabilidad: Cierta inmunidad contra radiación e interferencias.
  - Velocidad de envío de datos: más de 27 Mbps (Megabits por segundo).
  - Alcanzabilidad: 70m a 250m.
  - Seguridad: Distance-Time Bounded Protocol.
  - Latencia: Menos de 1 milisegundo para obtener posiciones (x, y, z).
  - Escalabilidad: Según el protocolo de comunicación utilizado.
  - Coste (infraestructura, nodos, mantenimiento): Alto.
  - Uso: Interior.
  - Banda de frecuencia: Entre 3.6 GHz y 10.1 GHz.

- **WIFI:**
  - Precisión en sistemas de posicionamiento: 5 a 15 metros.
  - Consumo energético: Alto.
  - Fiabilidad: Muy sensible contra radiación, obstáculos e interferencias.
  - Velocidad de envío de datos: más de 1 Gbps (Gigabits por segundo).

- Alcanzabilidad: 50m a 150m.
- Seguridad: Puede ser falsificado con relay attack.

Como se puede observar, existen varias tecnologías que pueden adaptarse al requisito de localización del sistema. A pesar de que pueden existir aquellas más rápidas que el Bluetooth, es este último el indicado para los requisitos del proyecto puesto que proporciona la posibilidad de crear una infraestructura con una gran cantidad de nodos, robusta y con una duración de la batería bastante larga, lo que hace que la infraestructura tenga un coste de mantenimiento muy bajo. Por otra parte, debido a su estándar y tiempo en funcionamiento, permite ser utilizado en sistemas propio. Por estas cosas, para el proyecto se decidió utilizar esta tecnología.

BALIZAS IBKS105

Con el objetivo de poder localizar al usuario dentro del edificio, es necesario utilizar dispositivos Bluetooth configurados para las distintas localizaciones de forma que se pueda establecer una conexión entre el dispositivo del usuario y el de la localización para determinar dónde se encuentra en usuario y poder darle las indicaciones pertinentes. Para ello, se ha decidido utilizar las balizas (beacons) IBKS105. Esta baliza se puede observar en la Figura 11.

Figura 11: Baliza IBKS105

Este dispositivo cumple con la Parte 15 de las normas FCC, es decir, su funcionamiento está sujeto a dos condiciones:

- El dispositivo no puede causar interferencias perjudiciales.
- Este dispositivo debe aceptar cualquier interferencia recibida.

Esto permite que pueda ser utilizado sin perjudicar a otros sistemas más importantes, haciéndola segura para su instalación en cualquier lugar asegurando que no interferirá con cualquier otro sistema.

Las especificaciones del dispositivo de pueden observar en la Figura 12.

Dimensions	Ø52.6 x 11.3 mm	**Case material**	ABS
Weight	24g	**Case finish**	Matte white
Core	Nordic nRF51822	**Fixing method**	Double side sticker
Radio Protocol	Bluetooth® Low Energy	**Operating Temperature**	-25 to +60°C
Distance Range	Up to 50m	**Storage Temperature**	0 to +35°C
Battery	Coin Cell CR2477 3V – 1000mAh	**Beacon Protocols**	iBeacon Eddystone: UID, URL, TLM & EID
Optional Sensors	Hall Accelerometer	**Firmware Update**	OTA (Over The Air)
Idle Current Consumption	2.4µA	**Certifications**	CE, FCC, IC, Anatel

Figura 12: Especificaciones de la baliza IBKS105

Como se puede observar, consta de una batería de 1000mAh cuya estimación de duración depende de los siguientes factores:

- **Número de ranuras habilitadas para la transmisión de información**.
- **Tipo de ranuras utilizadas para la transmisión:** iBeacon o Eddystone (en sus distintos formatos).
- **Potencia TX definida**.
- **Período de envío de mensajes**.
- **Modo de la baliza:** conectable o no conectable.

Por otra parte, el rango d escucha que poseen las balizas depende de la potencia en la transmisión, siendo la correspondencia la observada en las Figuras 13 y 14.

Distance (m)	TX Power (dBm)							
	-30	-20	-16	-12	-8	-4	0	+4
0	-63	-47	-42	-40	-40	-34	-27	-23
1	-92	-79	-71	-68	-66	-58	-59	-57
3	-99	-84	-82	-74	-73	-70	-67	-62
5	-101	-90	-88	-84	-82	-74	-71	-67
10		-95	-95	-89	-87	-81	-79	-75
15		-98	-97	-95	-92	-90	-84	-82
20		-102	-102	-97	-94	-92	-87	-85
30				-102	-96	-93	-92	-88
40					-99	-99	-94	-91
50					-102	-102	-100	-99

Figura 13: Tabla con la correspondencia Potencia-Distancia

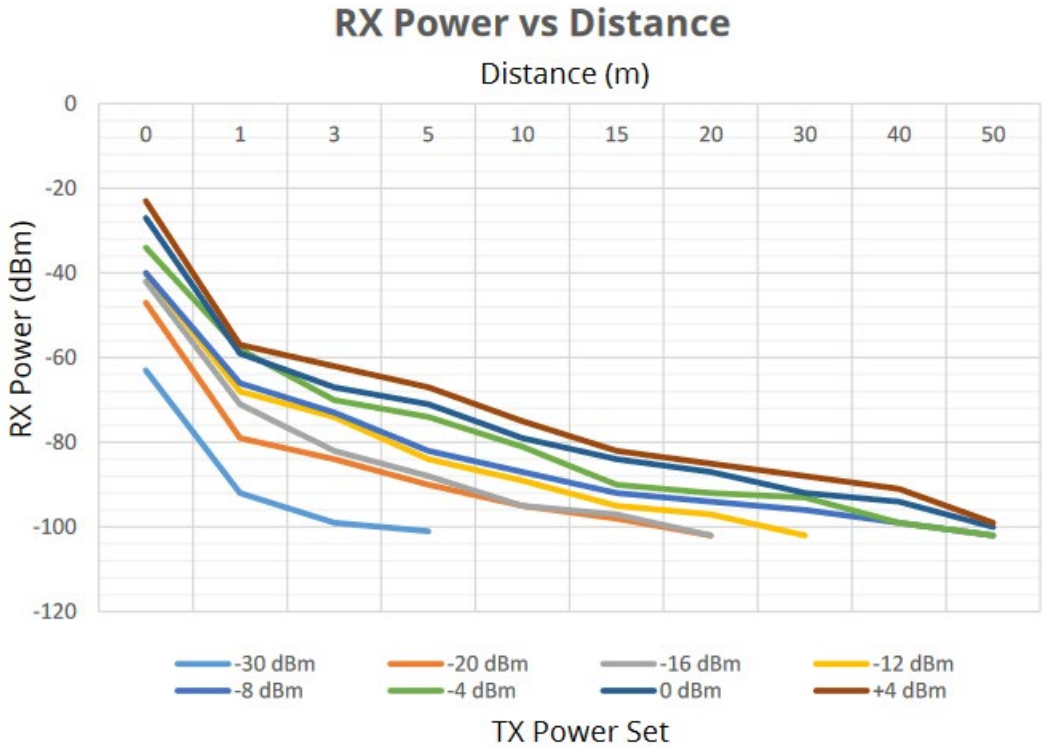

Figura 14: Diagrama en el que se observa la relación Potencia-Distancia

Para el caso de uso que nos concierne, se utilizarán las balizas IBK105 con la siguiente configuración:

- Todas las ranuras disponibles para transmitir una mayor cantidad de información.
- Todas las ranuras con el tipo iBeacon.
- Potencia TX de -12dBm con el que se logran 10 metros de rango.
- Periodo de envío de datos cada 60 segundos.
- Baliza en modo no conectable.

Teniendo en cuenta esta configuración, se logra una duración de la batería de 44 meses, es decir, para montar la infraestructura del proyecto se necesitan balizas IBKS105 de bajos coste que deben ser colocadas cada 10 metros y que, cuya duración de batería, permite que el mantenimiento de la infraestructura sea bajo, por tanto, todo esto lo hace una tecnología idónea para este proyecto.

En este ap.viar dichos datos a una red externa como podemos observar en la Figura 15.

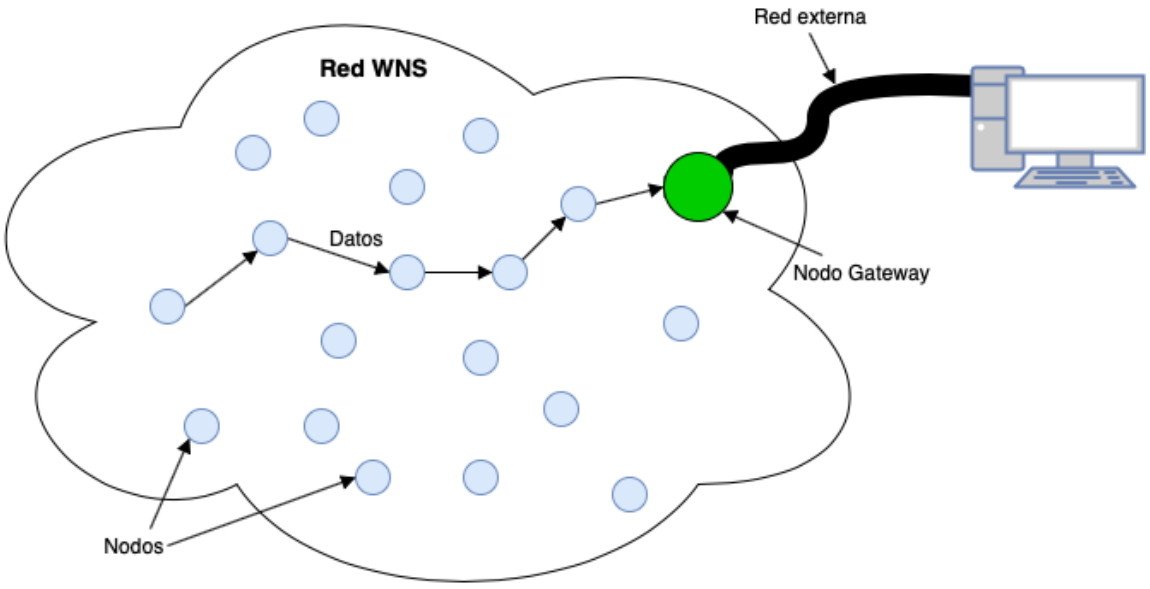

Figura 15: Red WSN

La arquitectura de un nodo de una red WSN tiene 4 componentes básicos (Figura 16):

- La unidad sensorial: que consta de dos partes:
  - El sensor: Recoge la información del entorno.
  - ADC (Analog to Digital Converter): Es un conversor de datos analógicos a datos digitales.
- La batería: necesaria para su autonomía.
- La unidad de procesamiento: procesa la información recogida por el sensor. Normalmente es un microprocesador o un microcontrolador con una memoria.
- Sistema de comunicación: permite la comunicación entre los nodos y la comunicación del nodo con el nodo Gateway.

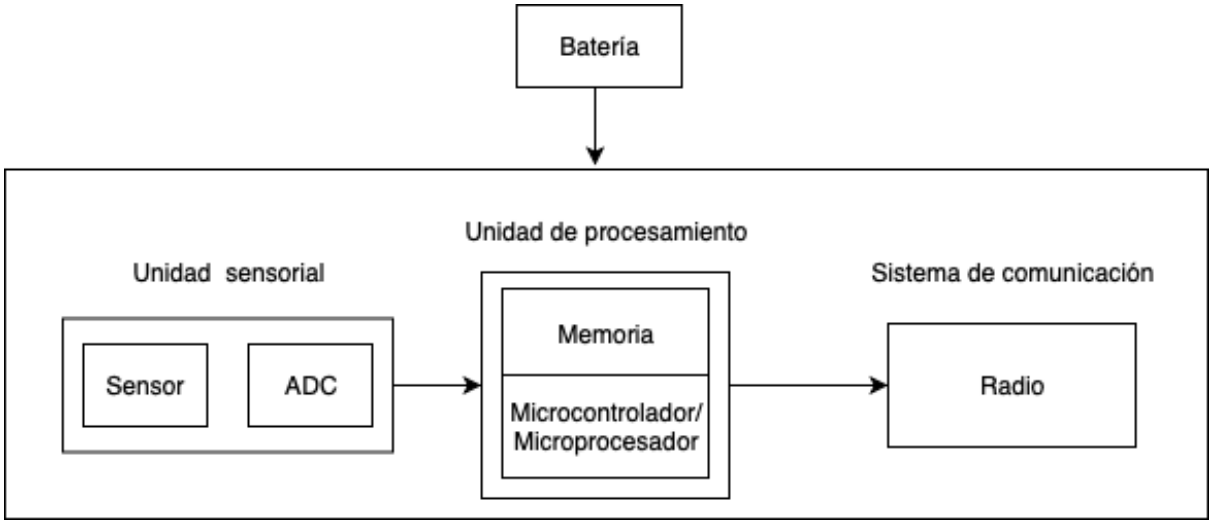

Figura 16: Nodo WSN

Las características de una red WSN:

- **No se utiliza una infraestructura de red:** no es necesario una infraestructura de red ya que los nodos pueden actuar como enrutadores, receptores o emisores de la información.
- **Topología dinámica:** la topología en una red de sensores no es fija, más bien tiene una naturaleza cambiante donde al añadir nuevos nodos, la red se adapta para poder comunicar la nueva información adquirida.
- **Comunicaciones *multihops* o *broadcast*:** las redes WSN es muy común utilizar protocolos de comunicación *multihops* donde la información pasa de un nodo a otro hasta llegar al nodo *gateway*, pero también es muy común utilizar un protocolo de comunicación broadcast donde la información se difunde por la red.
- **Bajos costes de producción:** para obtener datos con fiabilidad las redes WSN tiene un número elevado de nodos sensores por lo que, una vez definida su aplicación, son económicos al fabricarse en grandes cantidades.
- **Bajo consumo energético:** un nodo sensor debe estar compuesto por componentes de ultra bajo consumo energético para aumentar su autonomía. Sin embargo, esto es uno de los factores más sensibles ya que hay que equilibrar bien entre el consumo energético y la capacidad de procesamiento del nodo.
- **Limitaciones Hardware:** como necesitamos un nodo sensor de muy bajo consumo energético para su autonomía conlleva que los componentes hardware del nodo debe ser lo más sencillo posible, limitando la capacidad de procesamiento del nodo.

- **Variabilidad del canal:** el canal de radio es un canal propenso a errores debido a una lista de fenómenos tales como atenuación, desvanecimientos e interferencias que pueden provocar errores en los datos recogidos.
- **Tolerancia a errores:** un nodo dentro de una rede WSN puede seguir funcionando, aunque se haya generado errores en el sistema como por ejemplo la desconexión de algunos nodos de la red.

CLASIFICACIÓN DE LAS REDES WSN:

Las redes WSN tiene una amplia aplicabilidad, por lo tanto, una red WSN específica que ha sido desplegada de acuerdo con los requisitos de aplicación puede ser diferente a otra red WSN cuyos requisitos de aplicación son distintos. A continuación, se muestra una clasificación de las redes WSN en categorías:

- **Redes WSN estáticas y redes WSN móviles:** en muchas aplicaciones los nodos sensores son fijos sin movimiento y constituyen redes WSN estáticos. En cambio, en otras aplicaciones se necesitan que los nodos sensores se muevan en la red WSN para recoger información, dicha red se denomina redes WSN móviles.
- **Redes WSN deterministas y redes WSN no deterministas:** en una red de sensores determinista se calculan y se fijan las posiciones de los nodos. Sin embargo, existen otras redes en las que el entorno no permite determinar las posiciones de los nodos y requieren un sistema de control complejo. Estos últimos, forman una red no determinista.
- **Redes WSN con un único nodo *gateway* y redes WSN con múltiples nodos *gateway*:** en las redes WSN pequeñas, normalmente bastaría con un único nodo Gateway que recoge la información que mandan todos los nodos sensores. En redes WSN de gran tamaño esto no es posible y es necesario añadir más nodos Gateway en la cual un nodo sensor manda los datos generados al nodo Gateway más cercano. La información recogida por los nodos *Gateway* puede mandarse directamente a la red externa o mandarlas a un nodo *Gateway* responsable de mandar dicha información a la red externa.
- **Redes WSN con nodo *Gateway* estático y redes WSN con nodo *Gateway* móvil:** las redes sensores pueden tener un nodo *Gateway* estático fijado en una posición específica dentro de la región de sensores. También pueden tener un nodo *Gateway* móvil el cual es capaz de moverse a lo largo de la región de sensores para que la carga de los nodos sensores esté equilibrada.

- **Redes WSN *single-hop* y redes WSN *multi-hop*:** los nodos redes sensores de *single-hop* (salto simple) se comunican directamente con el nodo *Gateway* mientras que los datos generados por las redes de sensores de *multi-hops* (múltiples saltos) van de un nodo sensor a otro nodo sensor hasta llegar a un nodo Gateway.
- **Redes WSN auto configurables y redes WSN no auto configurables:** en las redes de sensores no auto configurables los nodos no pueden organizarse por sí mismos y dependen de una unidad de control para recolectar información. En cambio, en las redes auto configurables, los nodos sensores son capaces de organizarse y mantener la conexión y trabajar cooperando con otros nodos sensores para realizar la tarea.
- **Redes WSN homogéneas y redes WSN heterogéneas:** en las redes homogéneas todos los sensores tienen un consumo de energía, potencia computacional y capacidades de almacenamiento similares. Mientras que en las redes heterogéneas algunos nodos sensores tienen mayor potencia computacional y un mayor consumo energético por lo que las tareas de procesamiento y de comunicación se divide entre los nodos adecuadamente.

TOPOLOGÍA DE LAS REDES WSN:

Según las características de las redes WSN comentados anteriormente las redes sensores pueden formar diferentes topologías. A continuación, se describen algunas de las topologías más comunes:

- **Topología en estrella:** En dicha topología hay un solo nodo central llamado *Hub* o *Switch* y los demás nodos están conectados con el *Hub*. La topología en estrella es fácil de diseñar, implementar y ampliar, pero el problema es que todos los datos generados por los nodos pasan por el Hub por lo que juega un papel importante en la red y si falla el *Hub* puede resultar en un fallo de toda la red. Esta se puede observar en la Figura 17.

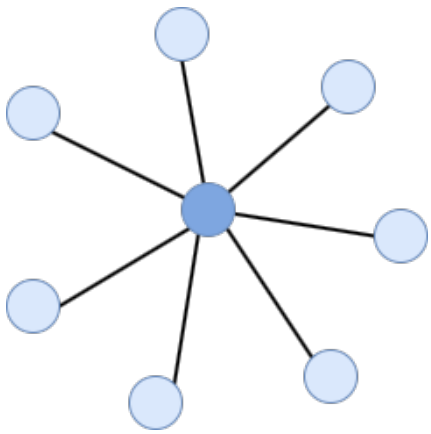

Figura 17: Topología en estrella.

- **Topología en árbol:** Una topología en árbol es una red jerárquica donde sólo hay un nodo raíz en la parte superior donde está conectado directamente con otros nodos que, a su vez, están conectados a otros nodos inferiores y así sucesivamente. La potencia de procesamiento y consumo energético es mayor en el nodo raíz y va disminuyendo a medida que bajamos por el orden jerárquico. Esta se puede observar en la Figura 18.

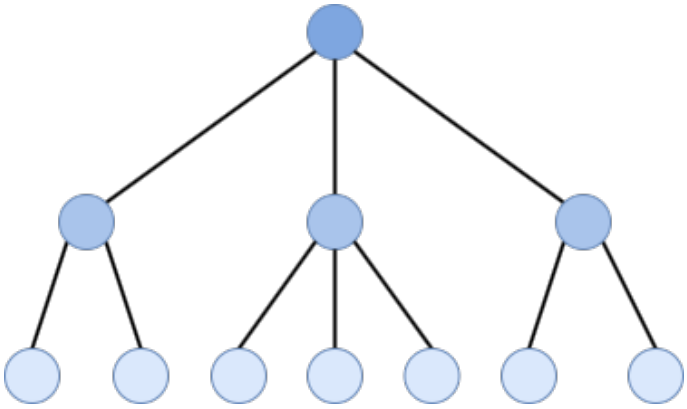

Figura 18: Topología en árbol.

- **Topología de malla:** En una topología de malla los nodos, además de transmitir sus propios datos, actúan como repetidores y retransmite los datos recibidos de otros nodos. Dicha topología puede ser de dos tipos, tal y como se representa en la Figura 19:
    - Topología de malla parcialmente conectada: Los nodos están conectados con uno o más vecinos.
    - Topología de malla totalmente conectada: Todos los nodos están conectados con los demás nodos.

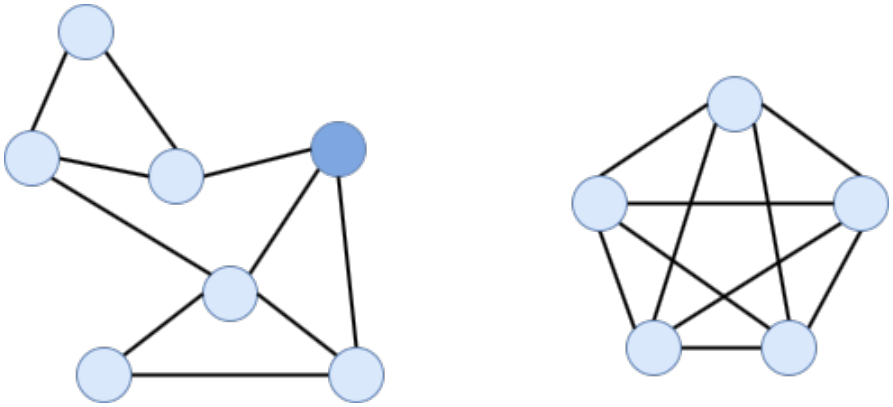

Figura 19: Topología de malla.

TÉCNICAS Y HERRAMIENTAS PARA EL DESARROLLO DEL SISTEMA Y SUS SERVICIOS

Este apartado tiene como objetivo mostrar cuales son las técnicas y herramientas escogidas para llevar a cabo la implementación del sistema, así como los servicios. Para ello, se hará un recorrido por estas técnicas y herramientas explicando brevemente en qué consisten y el motivo por el cual se utilizan. Posteriormente, se desarrollarán los distintos servicios que se implementarán por necesidad del sistema explicando cual va a ser el esquema a seguir.

TÉCNICAS Y HERRAMIENTAS A UTILIZAR

JAVASCRIPT

JavaScript es un lenguaje de programación muy utilizadas en la programación de páginas webs basado en el estándar ECMAScript. Aunque el lenguaje está diseñado principalmente para la programación del lado del cliente, existen frameworks que permiten crear programas de JavaScript en el lado del servidor. Características del lenguaje:

- **Multiparadigma:** es un lenguaje clasificado en múltiples paradigmas.
- **Lenguaje de programación interpretado:** es un lenguaje que no necesita ser preprocesado mediante un compilador para poder ser ejecutado.
- **Lenguaje orientado a objetos**.
- **Programación basada en prototipos:** es un estilo de programación orientado a objetos donde los objetos no se crean mediante la instanciación de clases si no por clonación de otros objetos.
- **Programación imperativa:** es un paradigma de programación que describe la programación en términos del estado del programa y sentencias que cambian dicho estado.
- **Débilmente tipado y dinámico:** las variables de JavaScript se declaran sin necesidad de indicarle el tipo de dato que se va a almacenar.
- **Programación dirigida por eventos:** es un paradigma de programación en que el flujo del programa está dirigido por eventos.

MONGODB

MongoDB es una base de datos NoSQL orientada a documentos, es decir, no guarda la información en registros sino en documentos. Los documentos son almacenados en formato BSON (representación binaria de JSON). Su principal característica es que no es necesario seguir un esquema de

almacenamiento dentro de una misma colección, es decir, se pueden almacenar documentos dentro de una misma colección que posean campos diferentes, así como menos campos o más campos.

Es una base de datos útil en aquellos proyectos en los cuales se necesitan almacenar datos semiestructurados lo que permite una mayor flexibilidad en la forma de almacenar los datos.

El objetivo que se busca con la incorporación de esta base de datos al proyecto que se ha desarrollado es de poder manejar datos contemplados en el sistema y las relaciones entre ellos de una forma más sencilla y comprensible, situación que sería más compleja en una base de datos relacional. Puesto que MongoDB permite libertad en los datos que son almacenados, lo hace una base de datos ideal para el proyecto, permitiendo el tratamiento de los datos en formato JSON lo cual hace que la funcionalidad del servidor sea más sencilla y clara.

MONGOOSE

Mongoose es un Object Document Mapper (ODM), es decir, permite definir objeto con un esquema tipado que se asigna a un documento de MongoDB. Proporciona una gran variedad de funcionalidades para trabajar con esquemas, actualmente contiene los siguientes SchemaTypes para especificar los tipos de los datos que conforman un esquema:

- String.
- Number.
- Date.
- Buffer.
- Boolean
- Mixed.
- ObjectId.
- Array.

Cada tipo de dato permite especificar:

- Valor predeterminado.
- Función de validación.
- Requerimiento del campo.
- Función get para manipular los datos antes de devolver el objeto.
- Función conjunto que permite manipular los datos antes de su almacenamiento.
- Crear índices para permitir que los datos se recuperen más rápido.

Mongoose se ha utilizado para la definición de los esquemas que se corresponden con los modelos definidos para el sistema, de forma que el manejo de la información almacenada en MongoDB sea más sencilla y clara.

JSON

JSON (JavaScript Object Notation) es un formato utilizado para guardar e intercambiar información de forma que cualquier persona lo pueda leer. Esto archivo tan solo contiene texto y se pueden distinguir por la extensión .json.

La información almacenada se encuentra estructurada y se utiliza principalmente para el intercambio de datos entre cliente y servidor.

En el caso que nos concierne, se ha utilizado como formato de intercambio de datos entre la aplicación y el servidor que realiza la gestión de los datos de la aplicación.

XML

XML (extensible Markup Language) es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos. Se entiende como lenguaje de marcado el conjunto de códigos que se pueden aplicar en el análisis de datos o la lectura de textos creados por computadoras o personas. El lenguaje XML proporciona una plataforma en la que poder definir una serie de elementos para crear un formato y generar un lenguaje personalizado.

Este lenguaje está formado por dos partes:

- **Prolog:** consiste en metadatos administrativos tales como declaración XML, declaración de tipo de documento y comentarios.
- **Body:** posee una parte estructural y otra de contenido.

Este lenguaje se basa en la simplicidad, generalidad y facilidad de uso, por tanto, ha terminado siendo utilizado en servicios web para el intercambio de información con APIs. Por esta razón, se enmarca dentro de la utilización de este proyecto, puesto que es un estándar que permite el intercambio de información entre servicio web y servidor.

HTML

El lenguaje de marcado de hipertextos (HTML, HyperText Markup Language) hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del

software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (código HTML) para la definición de contenido de una página web, como texto, videos, imágenes, etc… El estándar está a cargo del World Wide Web Consortium (W3C).

En el lenguaje HTML al añadir un elemento externo a la página no se incrusta directamente en el código de la página, sino que hace una referencia a la ubicación del elemento y el navegador del cliente es el encargado de recuperar dicho elemento. Por lo tanto, un código en HTML sólo incluye texto lo que permite que el código pese muy poco.

Este lenguaje se escribe en forma de etiquetas rodeadas por corchetes angulares y se compone de elementos y atributos.

Se utilizará para el Desarrollo de la web usada para la configuración del sistema junto a CSS y JavaScript.

CSS

La hoja de estilos en cascada (CSS, Cascading Style Sheets) es un lenguaje de diseño gráfico que permite definir y crear la presentación de un documento estructura en un lenguaje de marcado. Se utiliza sobre todo en el diseño visual de los documentos web e interfaces escritas en HTML aunque el lenguaje puede ser aplicado a cualquier documento XML (Extensible Markup Language).

APACHE

Apache es un servidor web HTTP de código abierto y multiplataforma (Unix, Windows, macOs). Es el servidor web más utilizado en el mundo. Se ha utilizado apache2 en el despliegue del subsistema web.

ANDROID

Sistema operativo utilizado por una gran cantidad de dispositivos móviles con pantalla táctil en la actualidad. Fue basado en Unix con el objetivo inicial de promover estándares abiertos en teléfonos y ordenadores. Este sistema operativo puede adaptarse a distintas resoluciones de pantalla y permite conexiones WiFi, Bluetooth, LTe y CDMA entre otras.

Por otra parte, permite el uso de navegador web, desarrollo de streaming y está capacitado para trabajar con distintos tipos de archivos como MP3, GIF, JPEG, etc.

Son numerosas las aplicaciones desarrolladas para este sistema operativo, sobre todo debido a su auge en su uso en dispositivos móviles. Para el desarrollo de cualquier aplicación en este sistema

operativo, gran parte de las ya desarrolladas se han llevado a cabo a través del IDE Android Studio. Este IDE permite el desarrollo de aplicaciones Android mediante el uso de los lenguajes XML, para la interfaz, y Java, para la programación de la funcionalidad.

JAVA

Java es un lenguaje de programación que nació con el objetivo de ser un lenguaje de estructura sencilla que pudiera ser ejecutado en diversos sistemas operativos, esto se consigue gracias a una máquina virtual que existe en cada sistema que es capaz de ejecutar Java y hacer de puente entre el lenguaje de programación y el dispositivo.

Este lenguaje permite crear aplicaciones y procesos en una gran variedad de dispositivos. Se trata de un lenguaje orientado a objetos que permite ejecutar un mismo programa en diversos sistemas operativos pudiendo ejecutar el código en sistemas remotos de forma segura.

Java se puede utilizar para el desarrollo de servicios web basados en SOAP o REST, aplicaciones de escritorio de consola o interfaz gráfica. Además, se utiliza para el desarrollo nativo de Android.

HTTP PROXY

Herramienta gratuita y fiable que ofrece a los usuarios un proxy para comunicaciones TCP y HTTP de alta disponibilidad con control de balanceo de carga para distribuir el tráfico de información entre varios servidores de un mismo servicio. Este balanceador de carga actúa como proxy inverso distribuyendo el tráfico de red entre varios servidores para incrementar la capacidad de procesamiento y confiabilidad.

Su uso es necesario para contemplar la distribución de peticiones del servicio API entre varios servidores para permitir su uso por parte de una gran cantidad de usuarios.

ALGORITMO DIJKSTRA

El algoritmo Dijkstra, también conocido como algoritmo de caminos mínimos, es un modelo clasificado dentro de los algoritmos de búsqueda cuyo objetivo es determinar la ruta más corta desde un nodo origen hasta un nodo destino. Su metodología está basada en iteraciones.

Este algoritmo presenta la capacidad de resolver problemas de la ruta más corta tanto en redes cíclicas como acíclicas de forma que los bucles presentes en una red no restringen el uso del algoritmo.

Para su uso define etiquetas a partir del nodo origen y para cada uno de los nodos subsiguientes. Estas etiquetas contienen información relativa al valor acumulado del tamaño de los arcos junto al nodo precedente más próximo de la red. Cabe destacar que la información es temporal, es decir, la información es susceptible de ser modificada siempre y cuando exista la posibilidad de que aparezca una ruta más corta.

Este algoritmo se utilizará para determinar cuál es la ruta más corta mediante la cual guiar al usuario teniendo en cuenta el lugar donde se encuentra y su destino final.

SERVICIOS A IMPLEMENTAR

API REST

Los servicios que ofrezca el sistema podrán ser accedidos a través de una API REST implementada para ello. Se trata de una arquitectura utilizada para diseñar aplicaciones en red. Una API REST es más efectiva gracias a HTTP debido a que este protocolo permite compartir información entre cliente y servidor. El intercambio de información se puede hacer en formatos como XML y JSON. Características:

- **Protocolo cliente-servidor:** el cliente y el servidor se mantienen débilmente acoplados, por tanto, el cliente no necesita conocer detalles de la implementación del servidor, así como el servidor no necesita tener conocimiento de cómo son usados los datos que envía al cliente.
- **Protocolo sin estado:** cada petición recibida por el servidor es independiente, es decir, no mantiene sesiones.
- **Peticiones cacheables:** el sistema de peticiones debe poseer un almacenamiento caché de varios niveles para evitar repetir varias conexiones entre servidor y cliente.
- **Interfaz uniforme:** define una interfaz genérica para administrar cada interacción que se realice entre cliente y servidor de manera uniforme simplificando así la arquitectura.
- **Sistema de capas:** permite al servidor tener varias capas para su implementación mejorando la escalabilidad, rendimiento y seguridad.

En este proyecto se llevará a cabo el desarrollo de una API REST que se utilizará para dar acceso a la información utilizada por el sistema. Para ello, se definirán una serie de recursos accesibles por estos escenarios.

BASE DE DATOS

Una base de datos es un conjunto de datos agrupados y almacenados sistemáticamente para su posterior uso. Las ventajas que aporta una base de datos son las siguientes:

- **Acceso rápido a los datos:** permite un acceso inmediato a la información alojada en ella de forma que se puede acceder a la misma, modificarla, eliminarla o agregar nueva información.
- **Almacenamiento de una gran cantidad de información:** el límite de la cantidad de datos a almacenar viene impuesto por la capacidad del disco dónde se almacene la información.
- **Centralización de la información:** la base de datos permite tener almacenados todos los datos en un servidor o varios servidores accesible una única forma.
- **Portable:** tan solo es necesario hacer una copia de seguridad de la información almacenada y trasladarla a otro servidor.
- **Dinámicas:** otorgan facilidad a la hora de gestionar la información almacenada o que se va a almacenar.

Dentro del proyecto, se incluirá una base de datos que se utilizará para el almacenamiento de datos de forma persistente con el objetivo de poder gestionar la información generada.

SISTEMA DESARROLLADO

En este apartado se detallarán las interacciones con el sistema de las principales funcionalidades que se implementaron, la arquitectura diseñada para el sistema y el funcionamiento del sistema mostrando imágenes del mismo.

DISEÑO DEL SISTEMA

En esta sección se describe la interacción del sistema por parte de cada una de las principales funcionalidades del sistema. Este diseño se trata de una aproximación cercana a la implementación y, por tanto, se utiliza como base para el desarrollo del sistema. Las principales funcionalidades diseñadas son las siguientes:

- **Configurar Mapa:** esta funcionalidad permite crear un nuevo mapa, lo que conlleva requerir datos como el tamaño del mismo, nombre y grupo al que pertenece. Con esta opción se pueden configurar distintas plantas para un mismo edificio con el objetivo de configurar posteriormente, los nodos y rutas pertenecientes al mismo. Esta interacción se puede observar en la Figura 20.
- **Asignar Beacon:** esta funcionalidad permite relacionar una baliza de posicionamiento con respecto a un nodo configurado de forma que, al configurar una ruta, al introducir un nodo se obtengan los datos de posición respecto a la baliza especificada. Esta interacción se puede observar en la Figura 21.
- **Configurar Ruta:** esta funcionalidad tiene como objetivo crear una ruta desde un punto inicial hasta un punto final donde se quiera llegar. Para ello, se deben especificar los distintos datos pertinentes junto a los nodos que conforman la ruta para poder posicionar al usuario y guiarle hasta su destino. Esta interacción se puede observar en la Figura 22.
- **Realizar Ruta:** esta funcionalidad es llevada a cabo por el usuario de la aplicación y es la que le permite escoger una ruta, desde un punto inicial hasta uno final, para que la aplicación le guíe posicionándolo a través de las balizas colocadas y especificadas para la ruta. Esta funcionalidad finaliza cuando el usuario ha llegado a su destino. Esto se puede observar en la Figura 23.

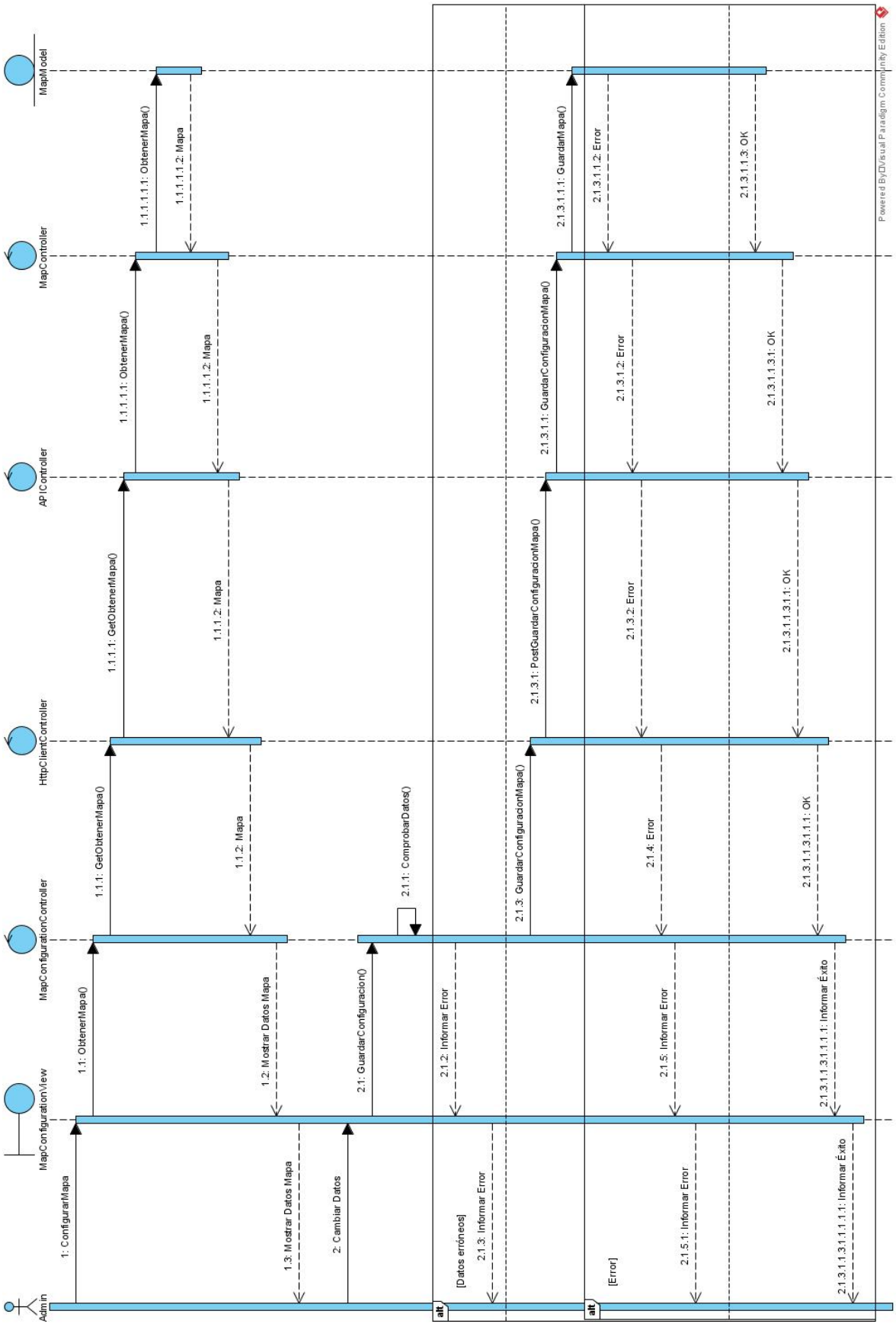

Figura 20: Diseño de la funcionalidad Configurar Mapa

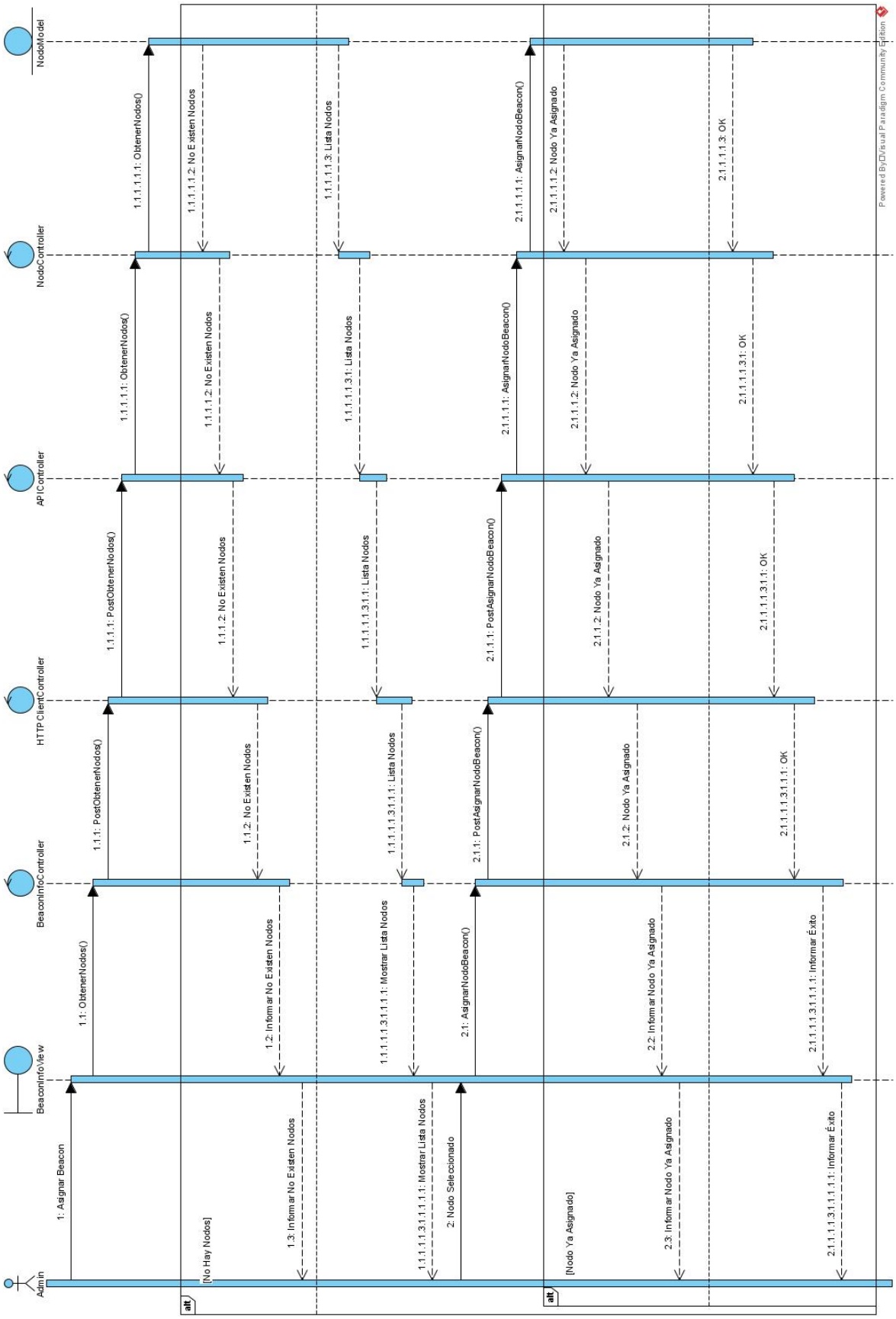

Figura 21: diseño de la funcionalidad Asignar Beacon

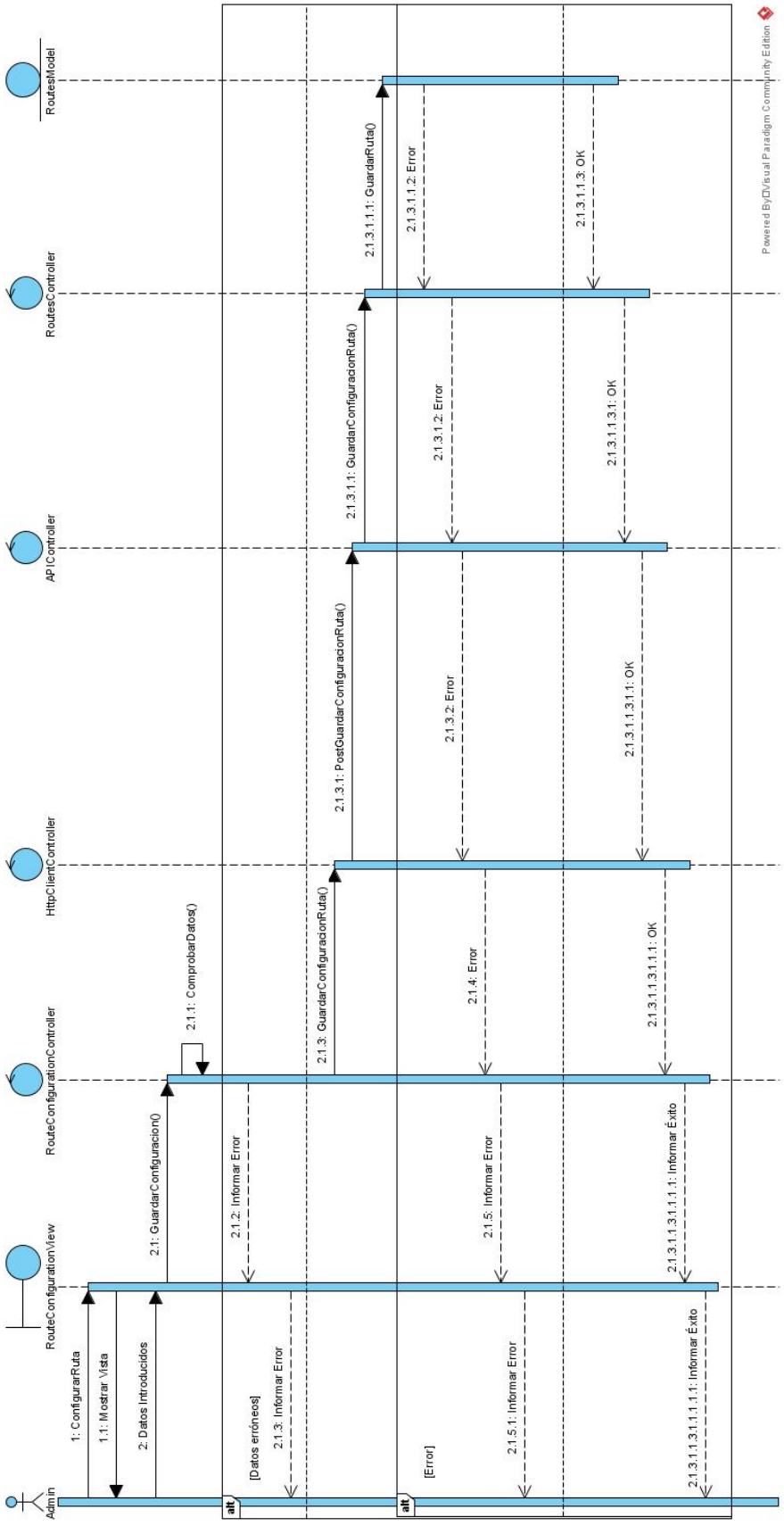

Figura 22: Diseño funcionalidad Configurar Ruta

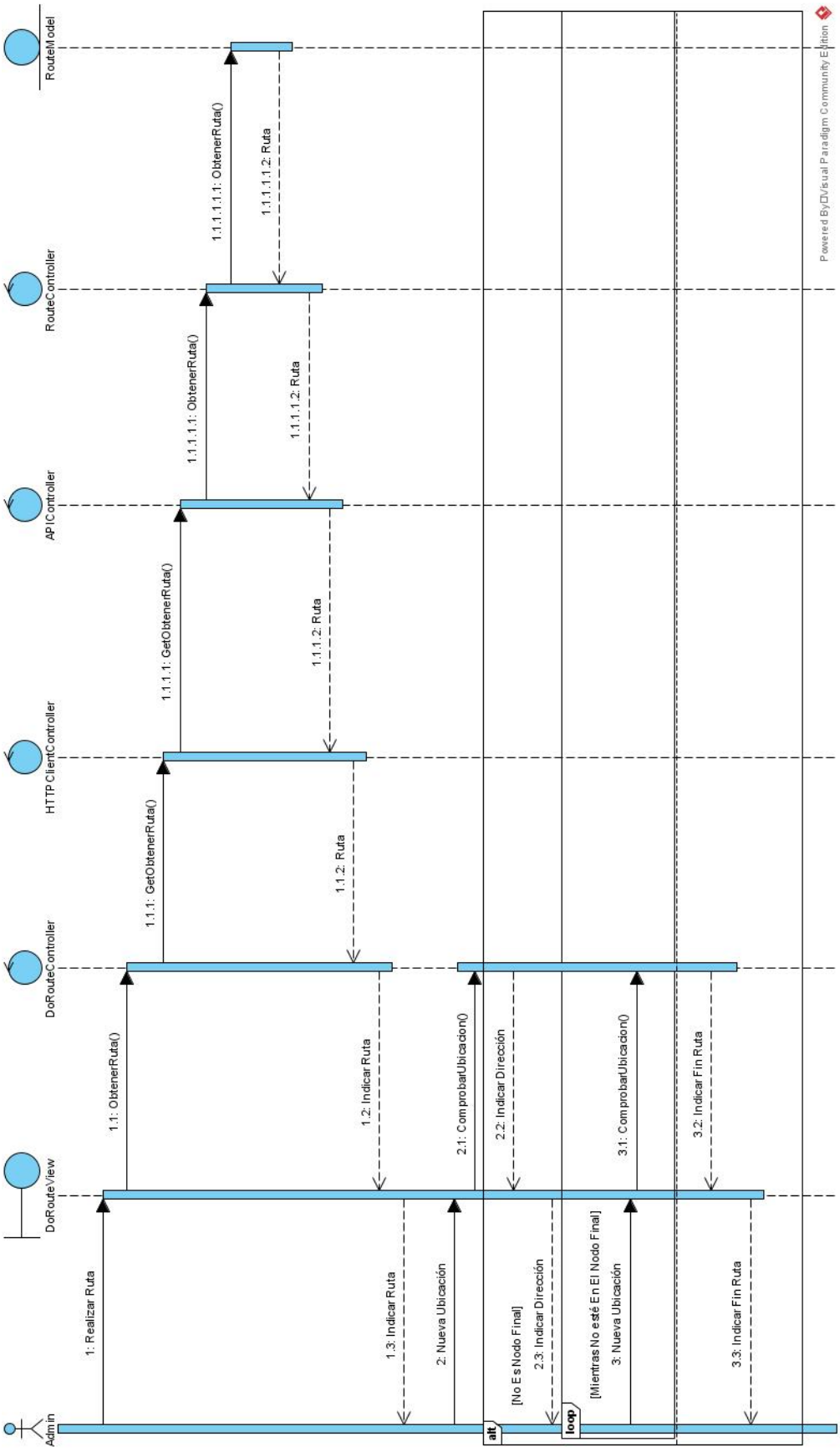

Figura 23: Diseño funcionalidad Realizar Ruta

ARQUITECTURA Y DESPLIEGUE DEL SISTEMA

En este apartado se va a detallar cual es la arquitectura del sistema con los elementos necesarios para el funcionamiento de este. Esta arquitectura se puede ver en la Figura 24.

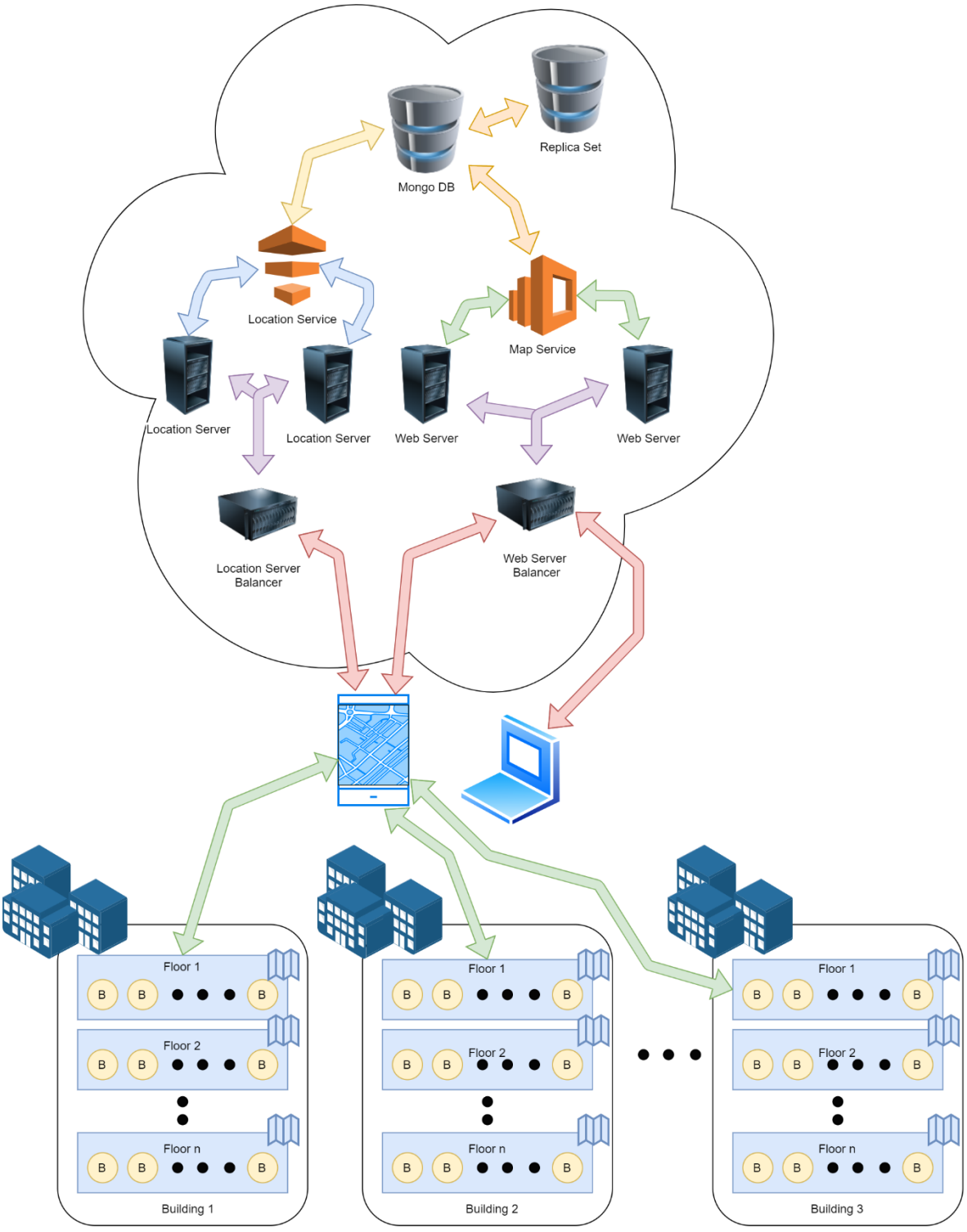

Figura 24: Topología en árbol.

La sensorización y ubicación del usuario en el sistema se realiza mediante la tecnología BLE. El uso de esta tecnología se basa en el despliegue de balizas de estas características por el entorno. Esto permite al dispositivo móvil que porta el usuario captar las señales enviadas por cada una de estas balizas y usarlas junto a los servicios de Map Service y Location Service de forma que tenga la capacidad de guiar a los usuarios por dentro de los edificios.

La sensorización en los edificios consiste en tener cada zona transitable por los usuarios balizada. Para ello, en cada una de las plantas de los edificios, se instalan un conjunto de balizas que dan lugar a un mapa de la planta. Estas balizas son instaladas de forma estratégica de modo que su posición coincida con los puntos de intersección de rutas y en las entradas/salidas de cada compartimiento del edificio, de manera que se pueda saber cuándo el usuario ha alcanzado el destino o conocer rápidamente el punto de partida. Un ejemplo de este despliegue de balizas se puede ver en la Figura 25. Las balizas este caso se encuentran instaladas en el centro del pasillo y en las entradas de las puertas de los despachos. Estas balizas también se colocan en puntos estratégicos para indicar informaciones de interés al usuario como puede ser al inicio y final de las escaleras.

Inicio y finalización de escaleras:

Entradas en despachos, aulas o baños:

Figura 25: Ejemplo de despliegue.

Los servicios que proporcionan la funcionalidad principal al sistema son Map Service y Location Service. Las principales características de estos servicios es su escalabilidad y su funcionamiento en la nube. Esto no solo permite de disponer de un centro de procesamiento central al cual se puede dotar

de unas mejores características técnicas, sino que otorga la posibilidad de ser usado en cualquier smartphone independientemente de los recursos hardware que disponga.

El servicio **LOCATION SERVICE** es el servicio encargado de la gestión de mapas y la neocartografía de interiores. Este servicio tiene la capacidad de asignar la creación y posición de planos de mapas de interiores en un sistema de cartografía global con su correspondiente cambio de coordenadas desde este plano al sistema de localización global. Este servicio también es el encargado de almacenar las posiciones y datos de todas las balizas desplegadas por el entorno, además, posee la capacidad de calcular.

El segundo servicio más importante es el **MAP SERVICE**, este servicio dispone de la capacidad de creación de rutas, así como realizar la navegación en interiores. Por otra parte, indica a los usuarios cuales son los destinos alcanzables, la ruta adaptada a sus necesidades y realiza las indicaciones de voz.

En el diseño de la arquitectura se ha tenido en cuenta la necesidad de escalabilidad de modo que estos servidores pueden ser replicados horizontalmente tantas veces como sea necesario para poder mantener la carga de los servidores estable. Para el balanceo de cargas de estos servidores, se usan balanceadores de carga de tipo Haproxi. En cuanto a base de datos del sistema se usa la base de datos no relacional MongoDB con su réplica set que permite también la escalabilidad del sistema.

SOFTWARE DE GESTIÓN

CONFIGURACIÓN DE MAPAS

Como ya se ha indicado anteriormente, el primer paso en el sistema es realizar la neocartografía de interiores, para ello se usa el servicio de LOCATION SERVICE. En este servicio, lo primero que se configura son los planos de interiores indicando la planta a la que pertenece. Para este proceso, resulta más sencillo usar los planos del edificio, sin embargo, si no se adecúa a las necesidades que se quieran cubrir, se puede realizar un plano nuevo realizado a medida. En la Figura 26 se pueden ver los planos, con el número de balizas correspondientes a cada planta de un edificio.

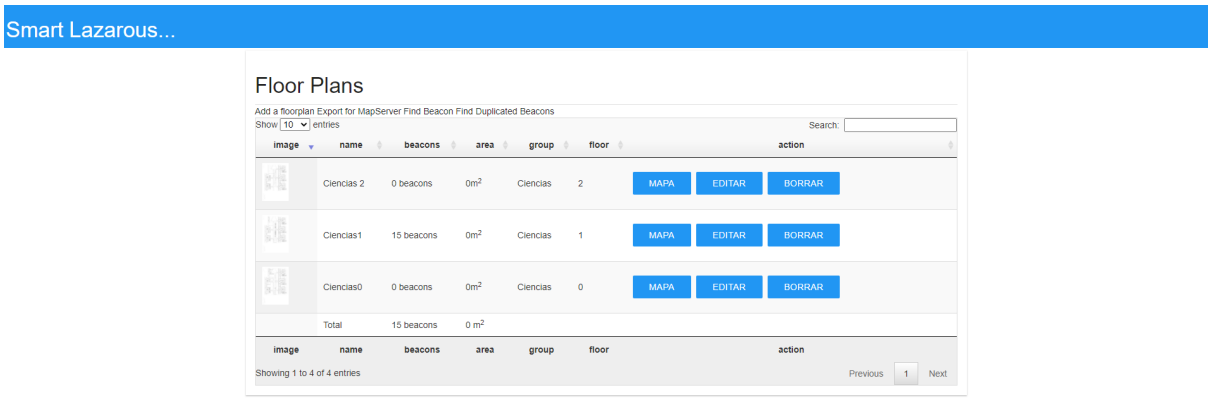

Figura 26: Ejemplo de despliegue.

Por cada uno de los planos que se introduce en el sistema, como se observa en la Figura 27, se indican las coordenadas centrales del plano correspondientes con la posición del sistema de cartografía global, el ángulo de rotación, opacidad, los puntos de origen x e y dentro de la imagen y la escala de la imagen que se le indica con los PPM (Puntos Por Metro) tanto para X como para Y.

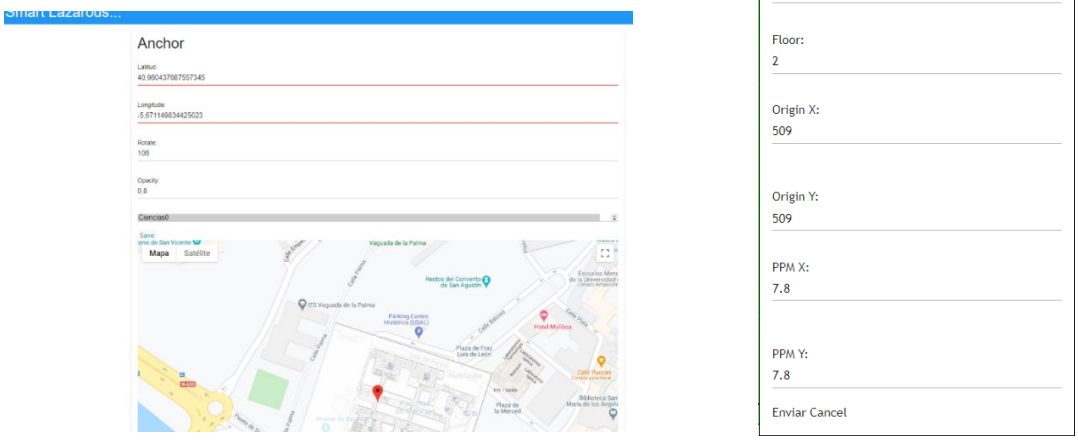

Figura 27: Posicionamiento de plano.

El siguiente apartado de este panel de configuración es el servicio LOCATION SERVICE mediante el cual se realiza la asignación posiciones de balizas en el mapa. Estas posiciones se pueden ver en la Figura 28.

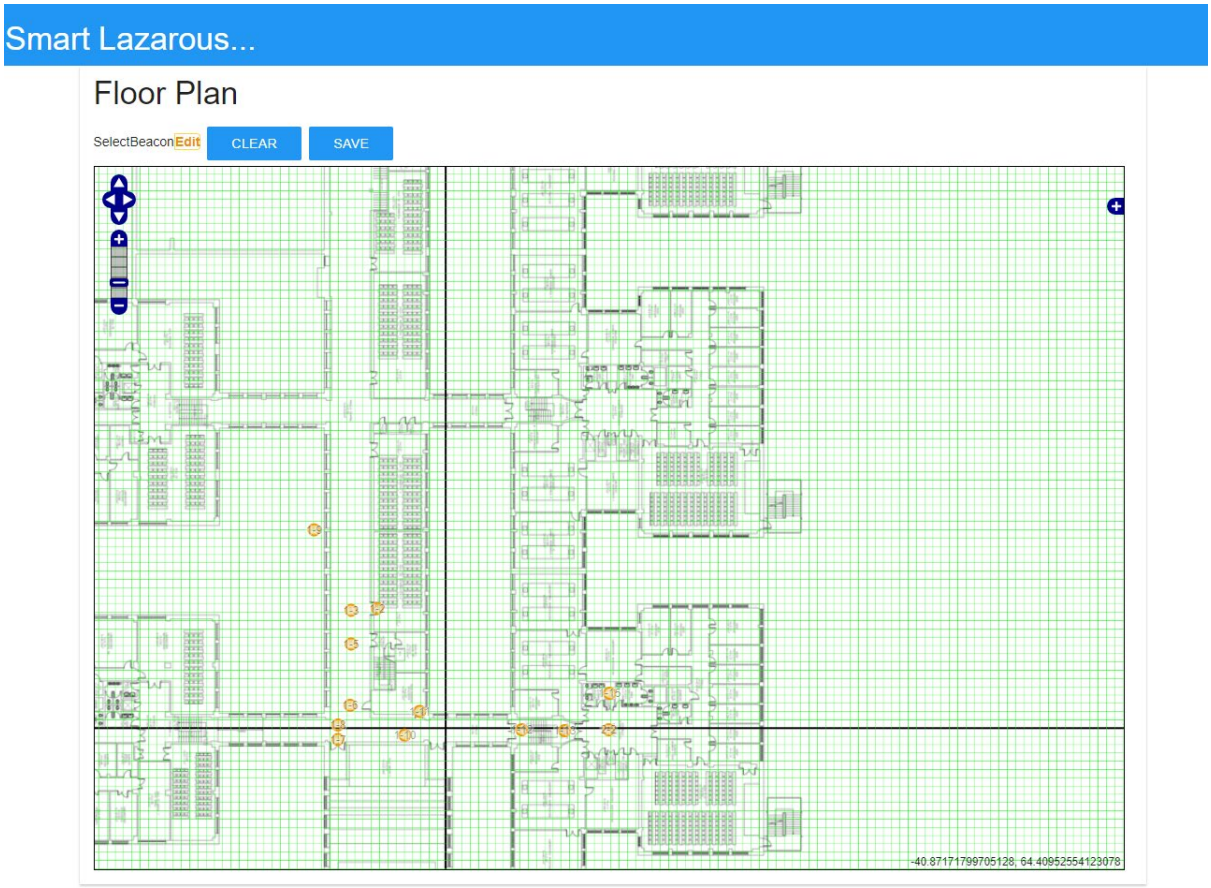

Figura 28: Asignación de balizas al mapa.

Para cada una de las balizas, como se puede observar en la Figura 29, disponemos de una serie de parámetros configurables necesarias para el funcionamiento del sistema.

- **UUID:** es el identificador único del servicio que dispone la baliza. Este identificador es el que usa en el dispositivo móvil para realizar filtrado de dispositivos en el momento de escaneo. Esto permite realizar filtrado de dispositivos que se encuentren únicamente registrados en el sistema.
- **MAC** (Media Access Control): es el identificador único de baliza dentro de la red Bluetooth.
- **Major:** es un parámetro configurable dentro del sistema que se usa para realizar agrupaciones por planta, es decir, para la planta baja el valor del parámetro mayor de todas las balizas de esa planta sería “0” para la primera planta sería el valor “1” y así sucesivamente. Esta

asignación permite de una manera muy rápida conocer con un simples escaneo para conocer cuál es la planta en la que se encuentra el usuario.

- **Minnor:** este parámetro se usa para indicar cual es el número de baliza dentro de la planta, es decir, si la planta posee *n* balizas, a cada baliza se le va a asignar un número empezando desde el 0 hasta llegar al *n* para identificarlas dentro de la planta. Habitualmente se indican como número consecutivos en el momento de la instalación, es decir, si la última incorporada ha sido *n*, entonces la siguiente que se incorpore será *n+1* y así consecutivamente.

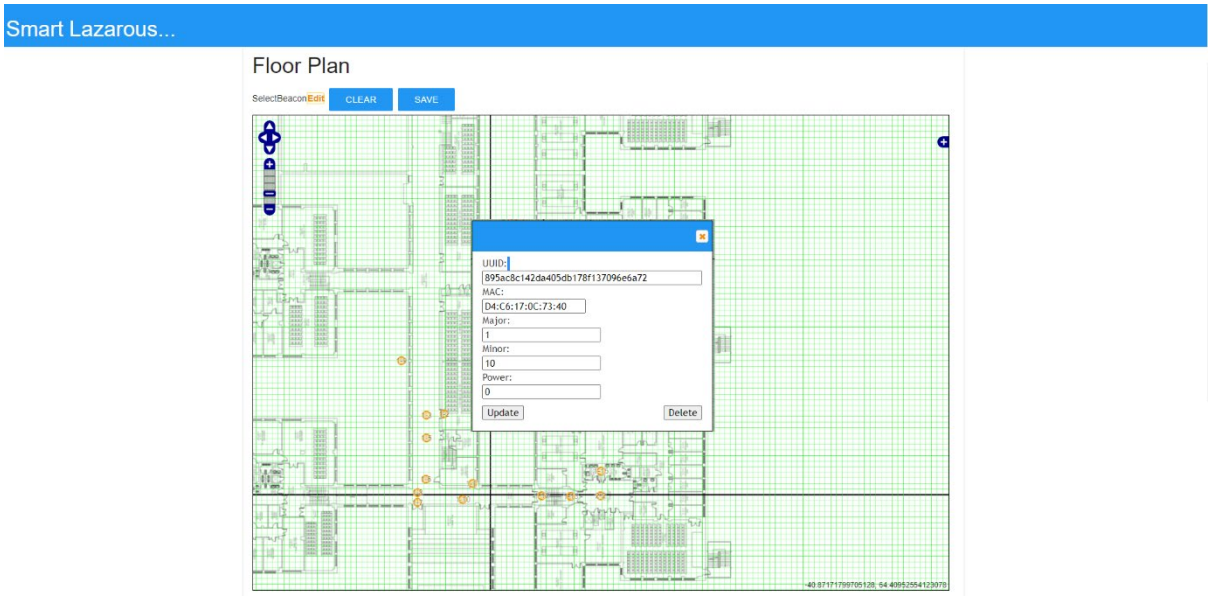

Figura 29 Asignación de balizas al mapa.

CONFIGURACIÓN DE RUTAS

El paso previo para realizar el guiado de interiores es la configuración de las rutas. Para ello, la utilidad presente en la Figura 30 permite establecer los nodos, intercepciones, puntos finales de las rutas, escaleras, ascensores y notificaciones enviadas al usuario.

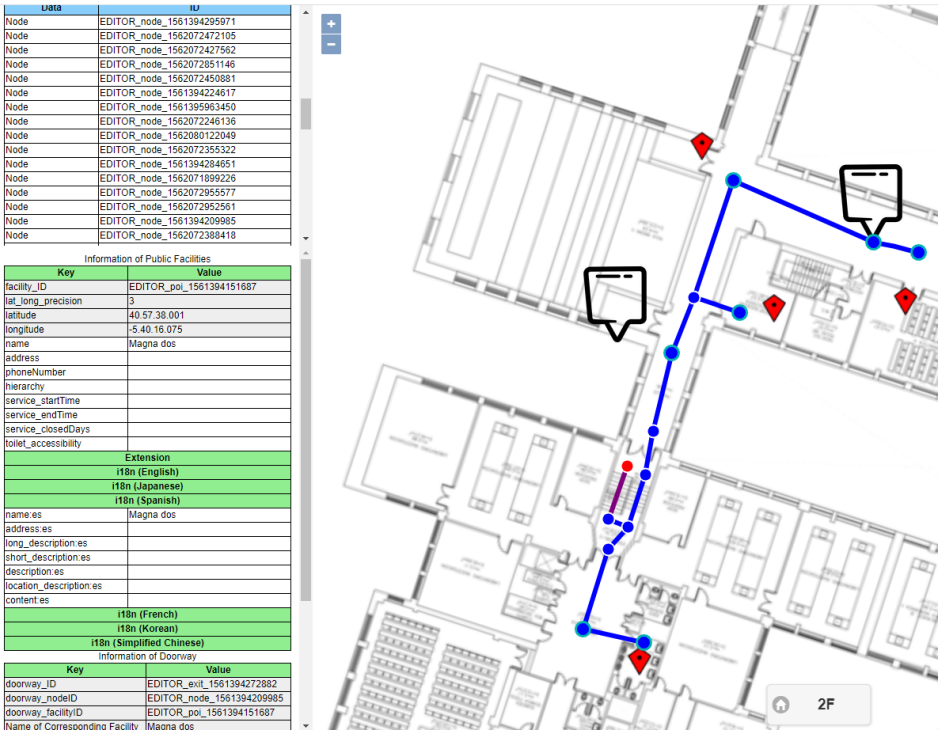

Figura 30: Herramienta de configuración de rutas.

Esta herramienta permite la interconexión entre plantas, indicando el tipo nivel de accesibilidad, es decir, indicando si esa ruta es viable físicamente para personas de muletas o de sillas ruedas de modo que el sistema pueda sugerir a estos usuarios una ruta alternativa que cumpla con sus requisitos, tal y como se puede observar en la Figura 31.

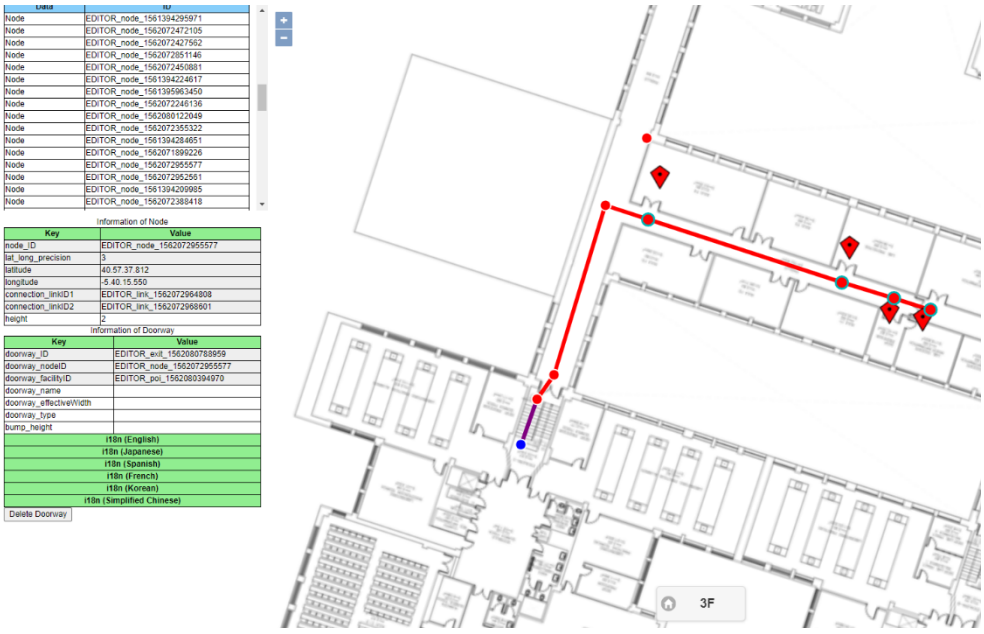

Figura 31: Herramienta configuración de rutas - interconexión entre plantas.

La herramienta de configuración de rutas se encuentra preparada para multidiomas con los que se pueden introducir todos los parámetros en múltiples idiomas. Esto permite ahorrar tiempo de configuración de las rutas ya que se puede traducir fácilmente los nombres de los destinos y las notificaciones.

APLICACIÓN MÓVIL

Mientras que las dos aplicaciones descritas anteriormente son usadas por los administradores del sistema, la aplicación móvil es la usada por los usuarios finales. En esta aplicación, dentro de los usuarios posibles, se encuentra contemplado el perfil de persona invidente. Esto ha conllevado el desarrollo de una parte de la funcionalidad que permita la **aceptación de comandos** de voz para el control de la aplicación, ya sea para introducir destino, cambios de configuración o finalización de guiado. El sistema, aparte del guiado visual dentro de la aplicación, también permite el guiado mediante órdenes acústicas o vibratorias por el móvil.

Esta aplicación captura las señales emitidas por las balizas y, mediante uso del servicio LOCATION SERVICE, es capaz de obtener las coordenadas y la planta en la que se encuentra dentro del edificio. La localización en el interior y las plantas se pueden observar en la Figura 32.

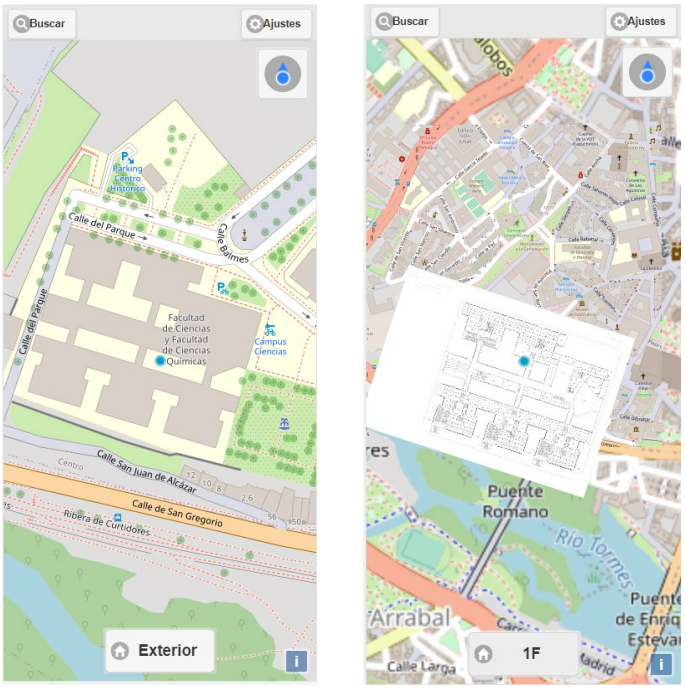

Figura 32: Herramienta de interconexión entre plantas.

En la pantalla de configuración, como se observa en la Figura 33, el usuario puede ver indicar si tiene necesidades especiales en su desplazamiento como puede ser el uso de muletas o el uso de una silla

de ruedas, esto conlleva a que el sistema haga una búsqueda alternativa evitando escaleras o algún pasillo más estrecho, optando por rutas con ascensores o rampas.

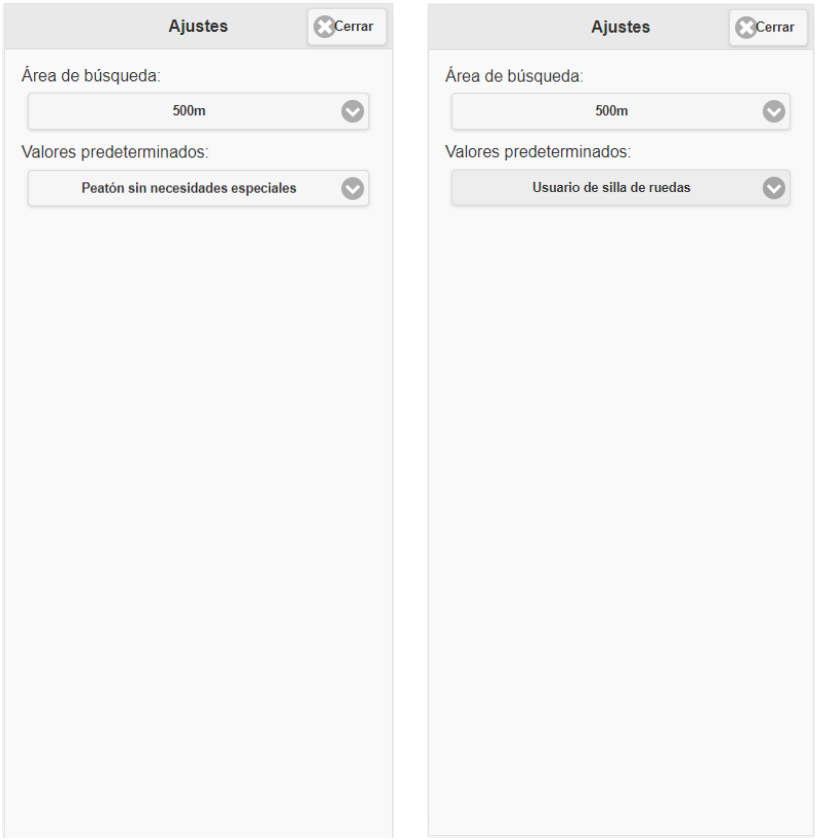

Figura 33: Ajuste de necesidades especiales.

La ultima funcionalidad disponible en la aplicación es la de guiado. Para ello, el usuario introduce una dirección de destino. La aplicación, haciendo uso del servicio MAP SERVICE, es capaz de calcular la ruta desde la posición en la que encuenta hasta destino mediante la implementación de un algoritmo Dijsktra, indicando las instruciones y la distancia para alcanzar el destino. Esto se puede observar en la Figura 34.

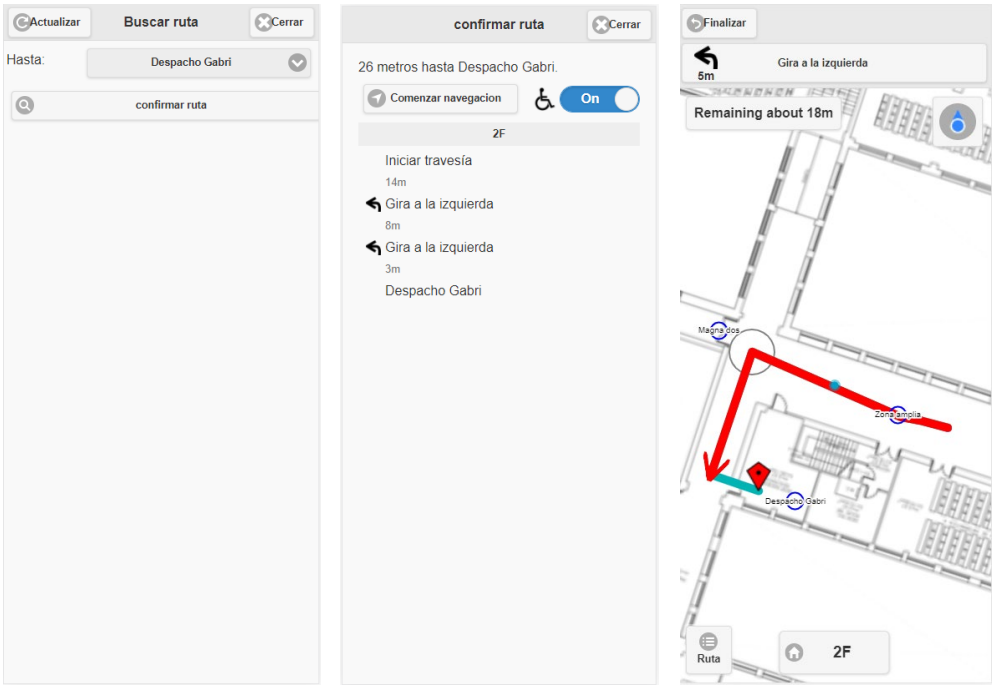

Figura 34: Ejemplo de ruta.

PÁGINA WEB DEL PROYECTO

La página web a la cual hace referencia este entregable se encuentra disponible en la url: http://smartlazarus.esalab.es/ Para su desarrollo se ha utilizado el HTML5, Bootstrap y CSS. La página tiene como objetivo presentar públicamente los aspectos relevantes de proyecto. El primer apartado de la página hace referencia a un resumen del proyecto donde se exhiben las necesidades del proyecto, el objetivo general del proyecto, el reto tecnológico y los aspectos innovadores del proyecto. La página principal presenta el aspecto que se puede observar en la Figura 35.

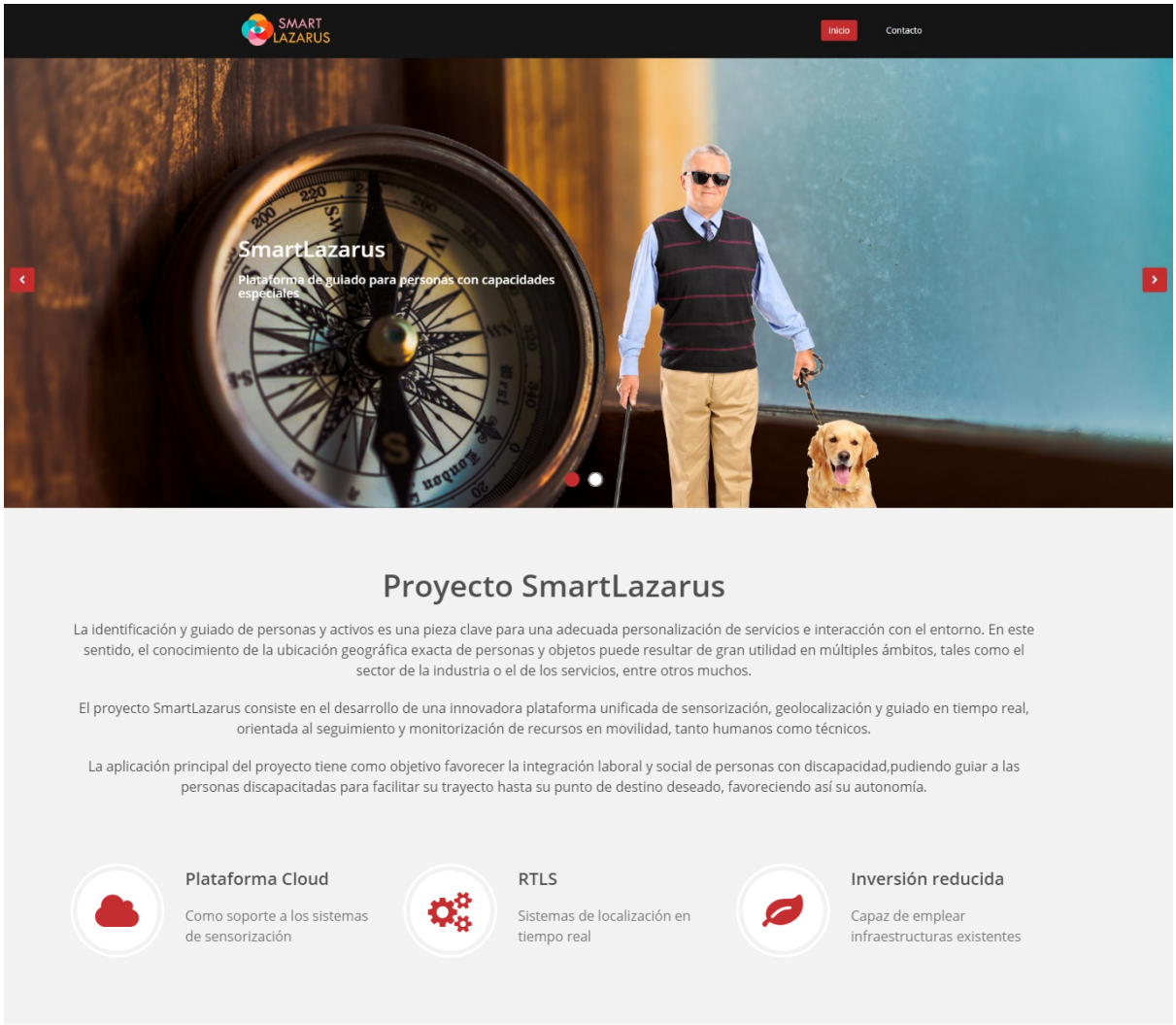

Figura 35: Sección resumen del proyecto

Las siguientes secciones hacen referencia a los objetivos y líneas de investigación del proyecto tal y como se puede observar en la Figura 36.

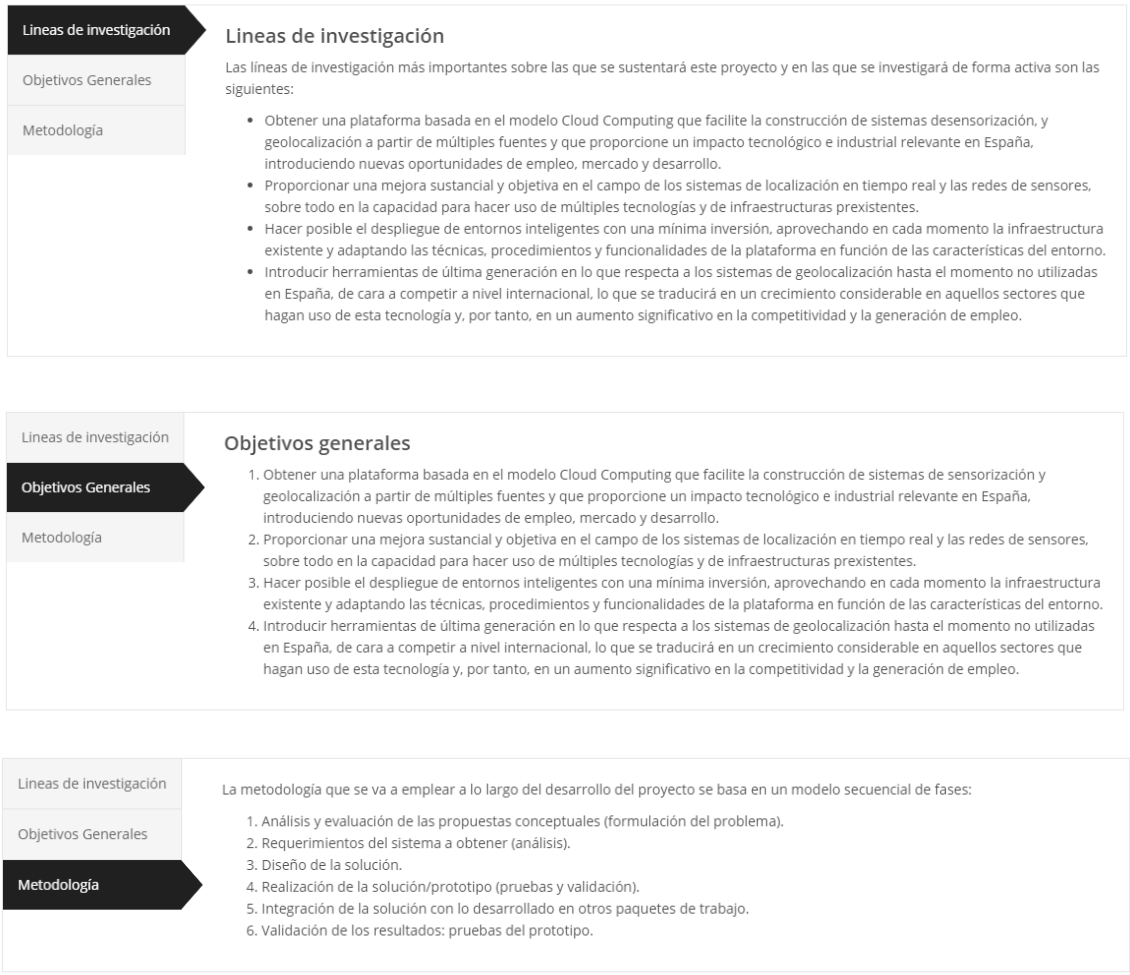

Figura 36: Sección objetivos y líneas de investigación.

Con objetivo de difundir la colaboración entre los participantes se ha añadido cada uno de los logos de los participantes en la sección de a continuación, como se puede observar en la Figura 37.

Figura 37: Participantes.

CONCLUSIONES

A lo largo de este documento se ha mostrado cual ha sido el desarrollo del proyecto llevado a cabo por el equipo encargado, explicando parte por parte los análisis realizados, así como las consideraciones tomadas y el desarrollo llevado a cabo.

Como se ha podido observar en la documentación, el análisis realizado respecto a tecnologías de localización existentes, así como técnicas y herramientas necesarias han girado respecto a un eje central tan importante como son los objetivos establecidos para el proyecto. Como se ha podido comprobar, todo el análisis realizado ha tenido como finalidad escoger aquellas tecnologías, herramientas y técnicas que supusieran un impulso en el desarrollo de la plataforma para lograr dichos objetivos.

Como se ha ido explicando en este documento, se ha desarrollado una plataforma cuya infraestructura permite poder llevar a cabo la localización de personas, tanto en interiores como en exteriores, con un coste de despliegue y mantenimiento mínimo con la finalidad de no tener que realizar una inversión grande. Esta plataforma ha sido desarrollada con herramientas de última generación que han sido integradas en un servidor en la nube diseñado y desarrollado por el equipo de proyecto, lo que permite una gestión más personalizada del sistema, así como asegurar la privacidad de los datos de los usuarios, además, este servidor ha sido desarrollado previendo su uso por una gran cantidad de usuarios. Por otra parte, el sistema ha sido desarrollado con la inclusión de un motor en tiempo real de alta precisión reduciendo el consumo de los dispositivos utilizados y mejorando el funcionamiento de otros sistemas de localización ya existentes.

El sistema ha sido probado con éxito por varios usuarios obteniendo unos resultados satisfactorios. Por tanto, se puede concluir que se ha logrado cumplir con los objetivos establecidos para este proyecto de forma satisfactoria, siendo probado con éxito por usuarios demostrando así la eficacia de la plataforma desarrollada.

REFERENCIAS

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, *6*(1), 37–66. https://doi.org/10.1007/BF00153759

Anderson, T., Peterson, L., Shenker, S., & Turner, J. (2005). Overcoming the Internet impasse through virtualization. *Computer*, *38*(4), 34–41. https://doi.org/10.1109/MC.2005.136

Andrzejak, A., Kondo, D., & Yi, S. (2010). Decision Model for Cloud Computing under SLA Constraints. Retrieved from https://hal.archives-ouvertes.fr/inria-00474849/

Armbrust, M., Stoica, I., Zaharia, M., Fox, A., Griffith, R., Joseph, A. D., … Rabkin, A. (2010). A view of cloud computing. *Communications of the ACM*, *53*(4), 50. https://doi.org/10.1145/1721654.1721672

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., … Warfield, A. (2003). Xen and the art of virtualization. In *Proceedings of the nineteenth ACM symposium on Operating systems principles - SOSP '03* (Vol. 37, p. 164). New York, New York, USA: ACM Press. https://doi.org/10.1145/945445.945462

Bezemer, C.-P., Zaidman, A., Platzbeecker, B., Hurkmans, T., & Hart, A. 't. (2010). Enabling multi-tenancy: An industrial experience report. In *2010 IEEE International Conference on Software Maintenance* (pp. 1–8). IEEE. https://doi.org/10.1109/ICSM.2010.5609735

Bhadauria, R., & Sanyal, S. (2012). Survey on Security Issues in Cloud Computing and Associated Mitigation Techniques. https://doi.org/10.5120/7292-0578

Bote-Lorenzo, M. L., Dimitriadis, Y. A., & Gómez-Sánchez, E. (2004). Grid Characteristics and Uses: A Grid Definition (pp. 291–298). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-24689-3_36

Buyya, R., Yeo, C. S., & Venugopal, S. (2008). Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. In *2008 10th IEEE International Conference on High Performance Computing and Communications* (pp. 5–13). IEEE. https://doi.org/10.1109/HPCC.2008.172

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, *25*(6), 599–616. https://doi.org/10.1016/J.FUTURE.2008.12.001

Capera, D., Georgé, J.-P., & Gleizes, M.-P. (n.d.). *Emergence of organisations, emergence of functions Co-Initialization with Multi-Agent Systems View project Mission Planning at IRT Saint-Exupery View project*. Retrieved from https://www.researchgate.net/publication/267195528

Carr, N. G. (2005). *The End of Corporate Computing SPRING 2005 VOL.46 NO.3 REPRINT NUMBER 46313*. Retrieved from https://pdfs.semanticscholar.org/6043/4c9a20d7276f61710abcb061ac1f556d87d9.pdf

Cestnik, B. (n.d.). Estimating probabilities: a crucial task in Machine Learning. Retrieved from http://www.temida.si/~bojan/_resources/Cestnik_Estimating_probabilities.pdf

Chahal, S., Hahn-Steichen, J., Kambhout, D., Kraemer, R., Li, H., & Peters, C. (2010). *An Enterprise Private Cloud Architecture and Implementation Roadmap*. *Intel Information Technology*.

Retrieved from http://download.intel.com/it/pdf/Entrprse_Priv_Cloud_Arch_final.pdf

Chellappa, R. (1997). Intermediaries in cloud-computing: A new computing paradigm. In *INFORMS Annual Meeting, Dallas* (pp. 26–29).

Cohen, W. W. (1995). Fast effective rule induction. In *Machine Learning Proceedings 1995* (pp. 115–123). Elsevier.

CSA. (2011). *SECURITY GUIDANCE FOR CRITICAL AREAS OF FOCUS IN CLOUD COMPUTING V3.0*. Retrieved from http://www.cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf

Cusumano, M., & Michael. (2010). Cloud computing and SaaS as new computing platforms. *Communications of the ACM*, *53*(4), 27. https://doi.org/10.1145/1721654.1721667

De Clercq, J. (2002). Single Sign-On Architectures (pp. 40–58). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45831-X_4

Docker. (n.d.). Enterprise Container Platform | Docker. Retrieved February 15, 2019, from https://www.docker.com/

Duda, R. O., & Hart, P. E. (1973a). Pattern classification and scene analysis. *A Wiley-Interscience Publication, New York: Wiley, 1973*. Retrieved from http://adsabs.harvard.edu/abs/1973pcsa.book.....D␀Ü3□ڶ␀ÉZ␀ȟ

Duda, R. O., & Hart, P. E. (1973b). Pattern classification and scene analysis. *A Wiley-Interscience Publication, New York: Wiley, 1973*.

Foster, I., Zhao, Y., Raicu, I., & Lu, S. (2008). Cloud Computing and Grid Computing 360-Degree Compared. In *2008 Grid Computing Environments Workshop* (pp. 1–10). IEEE. https://doi.org/10.1109/GCE.2008.4738445

Frank, E., & Witten, I. H. (1998). *Generating accurate rule sets without global optimization* (Computer Science Working Papers). *Computer Science Working Papers*. University of Waikato, Department of Computer Science. Retrieved from https://hdl.handle.net/10289/1047

Fraternali, P., Rossi, G., & Sánchez-Figueroa, F. (2010). Rich Internet Applications. *IEEE Internet Computing*, *14*(3), 9–12. https://doi.org/10.1109/MIC.2010.76

Glassner, A. S. (1995). *Principles of digital image synthesis. Vol. 1*. Morgan Kaufmann Publ.

Grance, T., Patt-Corner, R., Voas, J. B., & Badger, J. (2012). Cloud Computing Synopsis and Recommendations. *NIST Special Publication*, 146–800.

Gray, J., & Siewiorek, D. P. (1991). High-availability computer systems. *Computer*, *24*(9), 39–48. https://doi.org/10.1109/2.84898

Grobauer, B., Walloschek, T., & Stocker, E. (2011). Understanding Cloud Computing Vulnerabilities. *IEEE Security & Privacy Magazine*, *9*(2), 50–57. https://doi.org/10.1109/MSP.2010.115

Hall Eibe Frank, M., Holmes, G., Pfahringer Peter Reutemann, B., & Witten, I. H. (2009). The WEKA Data Mining Software : An Update. *ACM SIGKDD Explorations Newsletter*, *11*(1), 10–18.

Hamdi, M. (2012). Security of cloud computing, storage, and networking. In *2012 International Conference on Collaboration Technologies and Systems (CTS)* (pp. 1–5). IEEE.

https://doi.org/10.1109/CTS.2012.6261019

Holte, R. C. (1993). Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning*, *11*(1), 63–90. https://doi.org/10.1023/A:1022631118932

Horn, P. (2001). Autonomic computing: IBM's Perspective on the State of Information Technology.

Kamara, S., & Lauter, K. (2010). Cryptographic Cloud Storage (pp. 136–149). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-14992-4_13

Kubernetes. (n.d.). Production-Grade Container Orchestration - Kubernetes. Retrieved February 15, 2019, from https://kubernetes.io/

Lenk, A., Klems, M., Nimis, J., Tai, S., & Sandholm, T. (2009). What's inside the Cloud? An architectural map of the Cloud landscape. In *2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing* (pp. 23–31). IEEE. https://doi.org/10.1109/CLOUD.2009.5071529

Leo, B., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression trees. *Wadsworth International Group*.

Lipkowitz, K. B. (2007). *Reviews in computational chemistry. Vol. 23*. Wiley.

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., & Leaf, D. (2011). *NIST Cloud Computing Reference Architecture Recommendations of the National Institute of Standards and Technology*. Retrieved from https://ws680.nist.gov/publication/get_pdf.cfm?pub_id=909505

Lohr, S. (2007). *Google and I.B.M. Join in "Cloud Computing" Research*. Retrieved from http://www.nytimes.com/2007/10/08/technology/08cloud.html?_r=1&or...

McDougall, R., & Anderson, J. (2010). Virtualization performance. *ACM SIGOPS Operating Systems Review*, *44*(4), 40. https://doi.org/10.1145/1899928.1899933

Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology*. Retrieved from http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf

Mesos. (n.d.). Apache Mesos. Retrieved February 15, 2019, from http://mesos.apache.org/

Milojičić, D., Llorente, I. M., & Montero, R. S. (2011). OpenNebula: A Cloud Management Tool. *IEEE Internet Computing*, *15*(2), 11–14. https://doi.org/10.1109/MIC.2011.44

Newcomer, E., & Lomow, G. (2004). Understanding SOA with Web Services (Independent Technology Guides) 2004. Addison-Wesley Professional.

Newman, N. (2014). Apple iBeacon technology briefing. *Journal of Direct, Data and Digital Marketing Practice*, *15*(3), 222–225. https://doi.org/10.1057/dddmp.2014.7

Ogrizović, D., Sviličić, B., & Tijan, E. (2010). Open source science clouds. Retrieved from https://ieeexplore.ieee.org/abstract/document/5533642

Ortiz, S. (2011). The Problem with Cloud-Computing Standardization. *Computer*, *44*(7), 13–16. https://doi.org/10.1109/MC.2011.220

Pal, S., Khatua, S., Chaki, N., & Sanyal, S. (2011). A New Trusted and Collaborative Agent Based Approach for Ensuring Cloud Security. Retrieved from http://arxiv.org/abs/1108.4100

Parashar, M., & Hariri, S. (2005). Autonomic Computing: An Overview (pp. 257–269). Springer, Berlin, Heidelberg. https://doi.org/10.1007/11527800_20

Peltzman, S. (1976). Toward a More General Theory of Regulation. *The Journal of Law and Economics*, *19*(2), 211–240. https://doi.org/10.1086/466865

Petrucci, V., Loques, O., & Mossé, D. (2010). Dynamic optimization of power and performance for virtualized server clusters. In *Proceedings of the 2010 ACM Symposium on Applied Computing - SAC '10* (p. 263). New York, New York, USA: ACM Press. https://doi.org/10.1145/1774088.1774144

Platt, J. C. (1999). *Fast Training of support Vector Machines using Sequential Minimal Optimization*. Retrieved from http://www.cs.utsa.edu/~bylander/cs6243/smo-book.pdf

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Quinlan, J. R. (1993). *C4. 5: programs for machine learning*. Elsevier.

Ross, J. W., & Westerman, G. (2004). Preparing for utility computing: The role of IT architecture and relationship management. *IBM Systems Journal*, *43*(1), 5–19. https://doi.org/10.1147/sj.431.0005

Sefraoui, O., Aissaoui, M., & Eleuldj, M. (2012). OpenStack: Toward an Open-source Solution for Cloud Computing. *International Journal of Computer Applications*, *55*(3), 38–42. https://doi.org/10.5120/8738-2991

Sosinsky, B. A. (2011). *Cloud computing bible*. Wiley Pub.

Staab, S., van der Aalst, W., Benjamins, V. R., Sheth, A., Miller, J. A., Bussler, C., … Gannon, D. (2003). Web services: been there, done that? *IEEE Intelligent Systems*, *18*(1), 72–85. https://doi.org/10.1109/MIS.2003.1179197

Strategy, C. C. C.-D. C. (2009). Architecture, and Solutions. *Point of View White Paper for US Public Sector,*.

Subashini, S., & Kavitha, V. (2011). A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications*, *34*(1), 1–11. https://doi.org/10.1016/J.JNCA.2010.07.006

Swarm. (n.d.). Swarm mode overview. Retrieved February 15, 2019, from https://docs.docker.com/engine/swarm/

Tapia, D. I., Rodríguez, S., Bajo, J., & Corchado Rodríguez, J. M. (2008). *Multi-agent system for security control on industrial environments*. Retrieved from https://www.researchgate.net/publication/228865290

Tianfield, H. (2011). Cloud computing architectures. In *2011 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 1394–1399). IEEE. https://doi.org/10.1109/ICSMC.2011.6083853

Timofeev, R. (2004). *Classification and Regression Trees (CART) Theory and Applications*. Retrieved from https://s3.amazonaws.com/academia.edu.documents/38106508/timofeev.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1550149461&Signature=%2FTfZF%2B2nWnagTlYfgfpI2vHk%2FL8%3D&response-content-disposition=inline%3B

filename%3DClassification_and_Regression_T

Vapnik, V. (1963). Pattern recognition using generalized portrait method. *Automation and Remote Control*, *24*, 774–780.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, *10*(5), 988–999. https://doi.org/10.1109/72.788640

Vaquero, L. M., Rodero-Merino, L., Caceres, J., & Lindner, M. (2008). A break in the clouds. *ACM SIGCOMM Computer Communication Review*, *39*(1), 50. https://doi.org/10.1145/1496091.1496100

von Laszewski, G., Diaz, J., Wang, F., & Fox, G. C. (2012). Comparison of Multiple Cloud Frameworks. In *2012 IEEE Fifth International Conference on Cloud Computing* (pp. 734–741). IEEE. https://doi.org/10.1109/CLOUD.2012.104

Wang, L., von Laszewski, G., Younge, A., He, X., Kunze, M., Tao, J., & Fu, C. (2010). Cloud Computing: a Perspective Study. *New Generation Computing*, *28*(2), 137–146. https://doi.org/10.1007/s00354-008-0081-5

Weissman, C. D., & Bobrowski, S. (2009). *The Design of the Force.com Multitenant Internet Application Development Platform*. Retrieved from http://cloud.pubs.dbs.uni-leipzig.de/sites/cloud.pubs.dbs.uni-leipzig.de/files/p889-weissman-1.pdf

Wen, X., Gu, G., Li, Q., Gao, Y., & Zhang, X. (2012). Comparison of open-source cloud management platforms: OpenStack and OpenNebula. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery* (pp. 2457–2461). IEEE. https://doi.org/10.1109/FSKD.2012.6234218

Zhang, F., Chen, J., Chen, H., & Zang, B. (2011). CloudVisor. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles - SOSP '11* (p. 203). New York, New York, USA: ACM Press. https://doi.org/10.1145/2043556.2043576

Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications*, *1*(1), 7–18. https://doi.org/10.1007/s13174-010-0007-6