Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Estudio de la aplicación de burbujas 5G en el ámbito de la seguridad y la defensa

Grado en Ingeniería Mecánica

ALUMNO: Pedro Palacios Guerrero

DIRECTORES: Milagros Fernández Gavilanes

Ramón Touza Gil

CURSO ACADÉMICO: 2023-2024

Universidade Vigo

Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Estudio de la aplicación de burbujas 5G en el ámbito de la seguridad y la defensa

Grado en Ingeniería Mecánica

ALUMNO: Pedro Palacios Guerrero

DIRECTORES: Milagros Fernández Gavilanes

Ramón Touza Gil

CURSO ACADÉMICO: 2023-2024

Universidade Vigo

RESUMEN

El término abreviado 5G hace referencia a la quinta generación de tecnologías de red móvil. Esta generación trae consigo avances significativos que van a hacer de ella un vector que permitirá la incorporación de novedosas tecnologías como el internet de las cosas (IoT, *Internet of things*), los vehículos autónomos o la inteligencia artificial aplicada a objetos a la vida cotidiana. Históricamente, el mundo militar llevaba la iniciativa tecnológica mundial. Este paradigma ha cambiado en los últimos tiempos. En el esfuerzo por lograr incorporar estas tecnologías al ámbito de la seguridad y la defensa, ha aparecido el concepto de "burbuja o nube táctica". Se trata desplegar una infraestructura 5G flexible y robusta que le permita adquirir la superioridad en el dominio de la información mediante capacidades de comunicación ampliamente mejoradas, así como la digitalización y la robotización del teatro de operaciones. Para ilustrarlo, este trabajo tiene como propósito arrojar luz sobre cómo aplicar esta tecnología mediante el desarrollo de una aplicación en un entorno de simulación: la geolocalización de unidades empleando señal 5G. Esta prueba de concepto mostrará cómo sería un despliegue de nube táctica, así como la adaptabilidad y facilidad que aporta una burbuja 5G la implementación de nuevas herramientas.

PALABRAS CLAVE

Redes móviles, Mando y control, Burbujas 5G, Nube táctica, Geoposicionamiento por señal 5G

AGRADECIMIENTOS

En primer lugar, quisiera dedicar mi más sincero agradecimiento a mis tutores. A Doña Milagros Fernández Gavilanes, por su inestimable apoyo sin la que este trabajo no hubiera podido siquiera materializarse, tratando de hacer posible toda idea que surgió a lo largo del trabajo. Agradecerle también su gran paciencia con el autor. Al Capitán de Fragata Don Ramón Touza como promotor principal de la idea y por introducirme al mundo del 5G y su desarrollo actual dentro de la Fuerzas Armadas.

Quisiera igualmente agradecer el apoyo de diversas personas pertenecientes a la Armada, por tratarme más como un compañero que como a un subordinado. Al Capitán de Corbeta Don Pablo Cartujo por su desinteresada ayuda y apoyo. Al Capitán de Corbeta Don Jesús Abraham por su interés en ayudarme y creer en mi como futuro "experto" en 5G a nivel Armada. A los Tenientes de Navío Doña Loreto Fontanals y Don Jaime Melgar por dedicar parte de su ocupado tiempo como profesores de la Escuela Naval encontrar información sobre el tema dentro del ámbito de la Armada.

Finalmente quisiera dar las gracias a mi familia por enseñarme la vocación militar de la que tanto disfruto. Y queriendo concluir como solían hacerlo los autores clásicos, *Laus Deo.*

CONTENIDO

ÍNDICE DE FIGURAS

LISTADO DE ACRÓNIMOS

1G: Primera Generación
2G: Segunda Generación
3G: Tercera Generación
3GPP: *Third Generation Parthnership Program*
5G: *Fifth generation* o quinta generación de redes de telefónia móvil
5G NSA: *5G Non Stand Alone*
5G SA: *5G Stand Alone*
ADSL: *Asymetric Digital Subscriber Line*
AF: *Application Function*
AMF: *Access and Mobility Management Function*
AoA: *Angle of Arrival*
ARIB: *Association of Radio Industries and Businesses*
AuC*: Authentication center*
AUSF: *Authentication Server Function*
B-PSK: *Binary Phase Shift Keying*
BS: *Base station*
BSC: *Base Station Controller*
BTS: *Base Transmission Station*
C2: *Command and Control*
CN: *Core Network*
CP: *Carrier Phase*
CS-Fallback: *Circuit Switched Fallback*
EDGE: *Enhanced GPRS*
ENB: *e-NodeB*
EPC: *Evolved Packet Core*
ETSI: *European Telecommunications Standards Institute*
GEO: *Geostationary Earth Orbit*
GMSC: *Gateway MSC*
GPRS: *General Packet Radio Service*
GSM: *Global System for Mobile Communications*
HLR: *Home Location Register*
HSDPA: *High-Speed Downlink Packet Access*
HSS: *Home Subscriber Server*
HSUPA: *High-Speed Uplink Packet Access*
IA: Inteligencia Artificial
IoT: *Internet of things*
IP: *Internet Protocol*
ISO: *Imagen de disco*
LEO: *Low Earth Orbit*
LTE: *Long Term Evolution*
LU: *Location update*
MEO: *Medium Earth Orbit*
MIMO: *Multiple Input Multiple Output*
MME: *Mobility Management Equipment*
MSC: *Mobile Switching Center*
NEF: *Network Exposure Function*
NF: *Network Funtion*
NFC: *Network Function Components*

NFV: *Network Function Virtualization*
NR: *New Radio*
NRF: *Network Function Repository Function*
NSSAI: *Network Slice Selection Assistance Information*
NSSF: *Network Slice Selection Function*
OFDMA: *Orthogonal Frequency Division Multiple Access*
PCF: *Policy and Control Function*
PCRF: *Policy and Charging Rules Function*
PCU: *Packet Control Unit*
PGW: *Packet Data Network Gateway*
Q-PSK: *Quadrature Phase Shift Keying*
QAM: *Quadrature Amplitude Modulation*
QoS: *Quality of Service*
RAE: Real Academia Española
RAN: *Radio Access Network*
SBA: *Service Based Architecture*
SC-OFDMA: *Single Carrier-OFDMA*
SDN: *Software Defined Networking*
SGSN: *Serving GPRS Support Node*
SGW: *Serving Gateway*
SMF: *Session Management Function*
SMS: *Short message service*
SNR: *Signal to Noise Ratio*
SVD: *Single Value Decomposition*
TCP: *Transmission Control Protocol*
TDoA: *Time Difference of Arrival*
TIA: *Telecommunications Industry Association*
ToA: *Time of Arrival*
UDM: *User Database Module*
UDP: *User Datagram Protocol*
UE: *User equipment*
ULCC: *Ultra Low Latency Communication*
UMTS: *Universal Mobile Telecommunications System*
UPF: *User Plane Function*
UTRAN: *Universal Terrestrial RAN*
VLAN: *Virtual Local Area Network*
VPN: *Virtual Private Network*
WCDMA: *Wideband Code Division Multiple Access*
WSL: *Windows Subsystem for Linux*

1. INTRODUCCIÓN Y OBJETIVOS

> "El campo de batalla es una escena de caos constante. El ganador será el que controla el caos tanto el propio como el de los enemigos"
>
> (Napoleón Bonaparte)

1.1 Introducción y motivación

Una de las características más determinantes en la evolución del ser humano y de la civilización fue el desarrollo del habla y, por ende, la capacidad de comunicación. Ya sea en los negocios, el deporte, las relaciones familiares o en la dirección de equipos, la capacidad de transmitir y recibir información, así como la calidad con lo que se realiza resulta crítica para lograr el éxito en todos estos ámbitos.

Desde antaño, las Fuerzas Armadas han creado distintos métodos para lograr la comunicación fluida de información y órdenes para lograr el éxito en sus operaciones. Aníbal, Alejandro Magno, Julio César, Carlomagno, Álvaro de Bazán, Napoleón, Rommel o el Almirante Woodward (jefe de las fuerzas británicas en la Guerra de Malvinas) han pasado a la historia como líderes exitosos. Sin embargo, su ingenio y capacidad para la dirección de tropas no hubiesen logrado nada sin un sistema de comunicaciones claro y con capacidad de llegar hasta el último de los soldados. Los medios empleados han variado desde banderas, tambores, cornetas, señales luminosas y ya en siglo XX medios que emplean las ondas electromagnéticas como la radio en sus distintas bandas y frecuencias, así como el enlace por satélite. Aparece también en este siglo un medio de transmisión de información alternativo tanto a la palabra hablada o escrita como a señales sónicas y visuales: los datos (imágenes, vídeo, datos de sensores, etc.) Pero tal y como recuerda la frase de Napoleón previamente citada, el teatro de operaciones es un lugar caótico y este caos no ha hecho sino aumentar si cabe en los últimos tiempos, precisando mayor robustez, cantidad de información, calidad de esta y velocidad en su transmisión.

El ámbito de las telecomunicaciones es uno de los que más velozmente evoluciona en la actualidad. El mundo civil ha pasado a ocupar la delantera en cuanto a innovaciones en este campo, puesto que tradicionalmente ocupaba el mundo militar. Las Fuerzas Armadas de los distintos países tratan ahora de "militarizar" tecnologías del mundo civil a fin de mejorar sus sistemas de mando y control (*Command and Control*, abreviado como C2). En medio de este contexto, ya hace más de 40 años que aparecen las redes móviles de comunicaciones las cuáles permiten a cualquier persona disponer de una tecnología de radiocomunicación portable y con facilidad para enlazar con otros dispositivos con facilidad para el usuario. En un principio solo era voz, más tarde breves mensajes de texto, luego conexión a internet y finalmente se llega al paradigma actual en el que prácticamente toda persona porta un teléfono móvil en

su bolsillo capaz de transmitir y recibir grandes volúmenes de datos en cualquier lugar que disponga de cobertura.

Sin embargo, regresando a los escenarios bélicos o de crisis, estos no suelen disponer de la infraestructura necesaria para establecer una red móvil con propósito militar. Quizá sea más preciso decir que está o no puede establecerse o no con las condiciones necesarias de seguridad y capacidad para cumplir su propósito militar. No solo se trata de transmitir la información sino de que esta no llegue al oponente situando a las fuerzas propias en una situación de desventaja frente este. Las soluciones tomadas ante estos problemas han variado desde el empleo de medios más clásicos pero robustos y probados en combate como la radio, el empleo de mecanismos de red como el uso de VPN (*Virtual Private Network*), tunelización de los datos a través de redes civiles o el establecimiento de infraestructuras propias.

En el año 2017 la 3GPP (*Third Generation Partnership Proyect*) desarrolla el estándar de la quinta generación de tecnologías de red móvil, más conocido como 5G [1]. La implantación de dicho estándar solo está completada parcialmente en muchos lugares ya que aún combina elementos nuevos con elementos previos de la cuarta generación (habitualmente denominadas como tecnologías *legacy*). Las capacidades superiores y oportunidades que aporta esta generación han provocado el interés de las Fuerzas Armadas por aplicar esta tecnología a su ámbito dentro del cual se pueden encontrar varios casos de uso como el de la logística, el adiestramiento o su uso en operaciones reales. Para este último caso se plantea el empleo de un concepto denominado burbuja o nube táctica 5G. Se trataría de desplegar una infraestructura 5G completa dando cobertura a todas las unidades presentes en el teatro de operaciones. Esto permitiría toda clase de servicios desde llamadas de voz, videoconferencias, telemedicina, manejo autónomo de drones y vehículos no tripulados, realidad virtual orientada al mando y control, la recopilación de datos de numerosos sensores basados en IoT o por último el empleo de algoritmos de inteligencia artificial (IA o AI, *Artificial intelligence*). La Armada en colaboración con Telefónica y otras empresas están desarrollando actualmente la tecnología necesaria para hacer realidad el proyecto y que este pueda ejecutarse en el terreno de operaciones [2, 3].

Este trabajo viene motivado por la oportunidad que supone el terreno del 5G como campo para la innovación y el desarrollo de tecnologías. Como se expondrá más adelante, esta tecnología no solo está en pleno desarrollo, sino que ha sido concebida de manera abierta para permitir la colaboración de distintos desarrolladores. Sin embargo, la motivación última es la de exponer los conceptos y usos del 5G a los miembros de la Armada y las Fuerzas Armadas en general para afrontar con éxito un futuro en el que esta tecnología suponga el día a día de las unidades. Lograr la superioridad y la ventaja en las operaciones supone aumentar las probabilidades de éxito lo que redunda en el beneficio de la seguridad de todos los ciudadanos.

1.2 Objetivos

La meta final del presente trabajo es poner de manifiesto las capacidades que puede suponer una red 5G para el ámbito de la seguridad y la defensa mediante el desarrollo de una aplicación integrable en un núcleo de red 5G que permita la localización de unidades. El desarrollo creado no busca en ningún caso mejorar el estado del arte actual, sino poner de manifiesto su utilidad en este ámbito. Para ello se pretenden lograr los siguientes objetivos:

- Realizar un estado del arte genérico la tecnología para luego explorar los últimos avances en el campo de las operaciones militares y de seguridad.
- Construir una simulación del estándar en el software “NS-3” empleando el módulo 5G LENA del Centro Tecnológico de Telecomunicaciones de Cataluña. El objetivo es simular una infraestructura de red lo más similar posible a una burbuja táctica de tal forma que el código empleado sirva de base para el desarrollo de otras aplicaciones pensadas para ser empleadas en una nube 5G.

- Desarrollar una aplicación de geolocalización mediante el empleo de paquetes de datos transmitidos por 5G.

1.3 Estructura

Respecto a la estructura del trabajo, este está dividido en cinco partes fundamentales. La primera que se acaba de relatar en el capítulo 1 no es más que una breve introducción acerca del problema a tratar, así como de los objetivos a conseguir.

La segunda parte está recogida en el capítulo 2 comprende a su vez 4 secciones en las cuáles: se explora la evolución de las redes móviles a nivel civil hasta llegar al estándar del 5G y sus características de mayor interés (sección 2.1); las necesidades de una red de comunicaciones militar y el estado actual de aplicación del 5G dentro de este ámbito (sección 2.2) las herramientas de simulación de redes más empleadas actualmente en el ámbito del 5G (sección 2.3) y por último el estado actual del posicionamiento de dispositivos mediante el empleo de señal 5G (sección 2.4).

La tercera parte está recogida en el capítulo 3 donde se explicará el desarrollo del código empleado para la simulación de una burbuja táctica 5G, así como el segmento específico desarrollado para lograr la localización de un dispositivo de usuario en este contexto.

La cuarta parte desarrollada en el capítulo 4 trata sobre el análisis de los resultados obtenidos en el apartado anterior. Finalmente, en la última parte 5, se expondrán las conclusiones obtenidas y las líneas futuras tanto en trabajos de índole similar como de la tecnología en general en el ámbito de la defensa.

2 ESTADO DEL ARTE

2.1 Evolución de las redes móviles

2.1.1 Definición de red móvil

La Real Academia Española (RAE) define una red como "Conjunto de computadoras o de equipos informáticos conectados entre sí y que pueden intercambiar información. El término red móvil hace referencia a un tipo concreto de red en la cual sus extremos están conectados mediante un enlace por radiofrecuencia. El nombre habitual que se emplea para estos extremos de red es terminal de usuario (UE, *User equipment*). De la misma forma, al nodo que sirve de enlace y al que se conecta por señal radio se le conoce como estación base (BS, *base station*). Por último, este nodo se conecta a otro conocido como el núcleo de la red que constituye el principal centro de procesamiento de la información dentro de la red [4], tal y como se puede observar en la Figura 2-1.

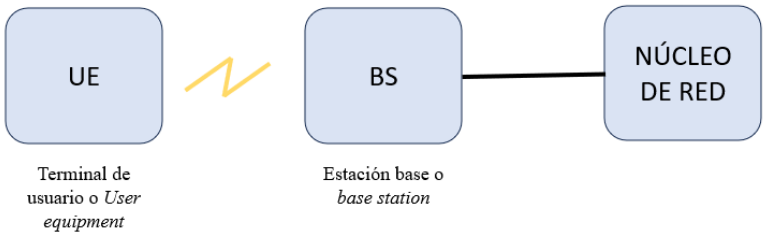

Figura 2-1: Esquema elementos básicos red móvil. Elaboración propia

Otra división de los componentes de una red móvil establece que existe un segmento de usuario, una red de acceso (RAN, *Radio access network*) y una red de núcleo (*Core Network*) así como la llamada interfaz de aire en referencia a la transmisión de ondas de radio, además de la modulación y otras características de las ondas.

La definición de red móvil estaría incompleta sin hacer alusión al adjetivo "móvil" que compone el concepto. Que una red ofrezca movilidad implica que el usuario pueda moverse con libertad sabiendo que siempre dispondrá de servicio en su dispositivo de manera automática ya que la red es capaz de adaptar su topología. Se plantean varios escenarios con diferentes variables que dan lugar a diversos protocolos para afrontarlos, aunque de manera resumida se pueden agrupar en los siguientes [5]:

- *Paging:* Estando un dispositivo en modo inactivo (encendido, pero sin transmisión de datos a nivel usuario) este ha de estar localizado. Para ello, la estación base más cercana a la que esté conectada enviará la información al núcleo de la red mediante un mensaje LU (*Location*

update) que servirá para la actualización de la base de datos del sistema a fin de poder encaminar las transmisiones.

- *Handover:* Se produce cuando el terminal está activo (por ejemplo, en medio de una llamada) y este cambia de célula de cobertura. La red ha de ser capaz de permitir una comunicación entre estaciones base que les permita ceder el testigo del servicio sin que esta sufra interrupciones.

Cada una de las partes de una red móvil ha experimentado mejoras muy sustanciales desde su nacimiento como pueden ser la aparición de los teléfonos inteligentes, la mejora en los métodos de modulación de la señal, así como de multiplexación para permitir un acceso a más dispositivos con mayores tasas de transmisión de datos o la reestructuración de los elementos del núcleo para ofrecer unas mejores prestaciones de servicio. Todas estas mejoras serán expuestas de manera resumida hasta llegar a la quinta generación de telefonía móvil.

2.1.2 El camino al 5G: desde los primeros móviles hasta el 4G

El 17 de octubre de 1973 aparece el primer teléfono móvil llamado "*Radio Telephone System*". Este dispositivo fue creado por el ingeniero Martin Cooper de la empresa Motorola [6]. La idea inicial es simple, el extremo de la red telefónica estaba enlazado al resto por un enlace radio que conectaba el dispositivo UE con una estación BS que enlazaba con el resto de la red por cable de telefonía.

La primera generación (1G) surge en los años 80. Pese a que ya existía el terminal de usuario no sería hasta estos años que se desarrollaría la infraestructura y los protocolos para dar soporte a este nuevo tipo de red. Aparece el concepto de celda, asociado al área de cobertura de cada una de las estaciones base, así como la comunicación entre las mismas a fin de que el UE fuese capaz de conectarse a la celda correspondiente a su localización. Este proceso llamado itinerancia (*roaming*) no era perfecto pues los servicios, prestaciones y tecnologías ofertados por las distintas compañías no garantizaban la compatibilidad con todos los dispositivos [2]. La transmisión de información seguía siendo analógica.

Habría que esperar hasta la década de los 90 en que se produjo un cambio de paradigma transcendental: la digitalización de las comunicaciones [2]. Se evolucionó del envío de señales analógicas al envío de secuencias de bits, al igual que en una red de ordenadores. Aparece un estándar común, el GSM (*Global System for mobile communication*), que al ser utilizado por todas las compañías permite una movilidad real a los usuarios. Además, se crea un nuevo tipo de servicio que permite el envío de otra información distinta de la voz: el SMS (*Short message service*). Está innovación era el comienzo del camino hacia el paradigma actual donde prima la transmisión de datos por encima de los servicios de llamada. Todas estas innovaciones suponían el nacimiento de la segunda generación [7].

Ahora, bien surge la pregunta de cómo era la infraestructura de la red en aquel momento. Trabajando sobre los elementos básicos definidos previamente en la sección 2.1.1, los principales cambios a nivel estructural se producirán en el núcleo. En el caso del 2G, este estaba compuesto por un centro de conmutación de móviles (MSC, *Mobile switching center* encargado de encaminar los paquetes a través de la red), unas bases de datos con los registros de localización de los dispositivos (HLR, *Home location register* y VLR, *Visitor location register*), el AuC (*Authentication center* encargado de proporcionar la autenticación y la cifra al sistema) y por el último el GMSC (*Gateway* MSC que permite la conmutación con otras redes). En cuanto a la parte referente a la estación base, esta tiene dos elementos, la antena propiamente (BTS, *Base transmisión station*) y una estación de control (BSC, *Base station controller*) para la gestión de recursos y la realización del procedimiento de *handover* [8]. Todos los elementos descritos se pueden ver representados de forma esquemática en la Figura 2-2.

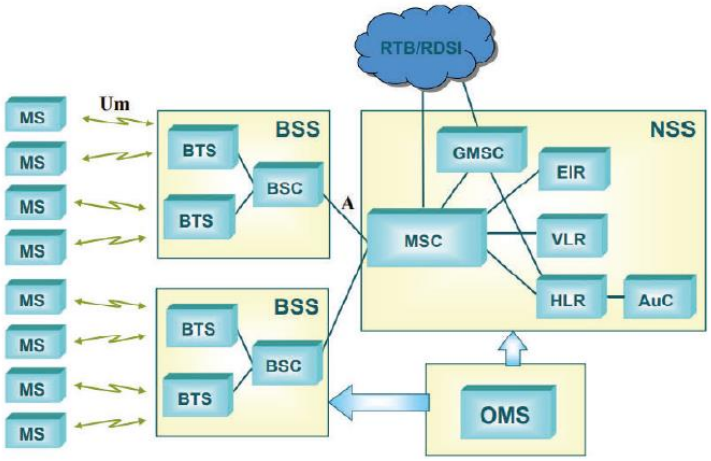

Figura 2-2: Estructura red GSM [7]

En la etapa de transición entre el 2G y el 3G aparecerían varias mejoras notables al sistema que pretendían imitar el avance que se estaba produciendo en la telefonía fija proveyendo internet a través de línea de teléfono. La primera de ellas es el GPRS (*General Packet Radio Service*) que permite la transmisión de datos implementando el protocolo IP (*Internet Protocol*). El núcleo de la red tuvo que evolucionar para albergar nuevos elementos encaminados a la gestión de estos paquetes siendo estos el SGSN (*Serving GPRS Support Node*), el GGSN (*Gateway GPRS Support Node*) y el PCU (*Packet Control Unit*) proporcionando a la red y las estaciones base la capacidad de conectarse a internet y de manejar paquetes IP [7]. La adición de estos elementos adicionales viene reflejada en la Figura 2-3. La segunda mejora se denominó como EDGE (*Enhanced GPRS*) que vino a incrementar la velocidad a 384 kbps [9].

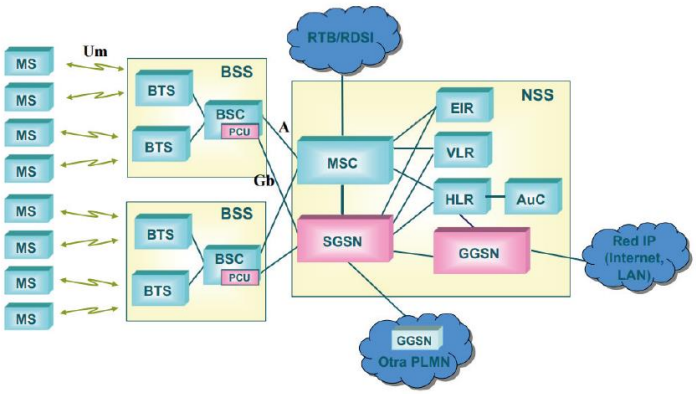

Figura 2-3: Estructura red GPRS [7]

Al comienzo del nuevo milenio se produjo la aparición de la tercera generación. Esta nace auspiciada por la *3rd Generation Partnership Proyect* (3GPP) una colaboración de diversos organismos dedicados a la estandarización de las telecomunicaciones [10]. Los tres miembros originales eran la ETSI (*European Telecommunications Standards Institute*), la TIA (*Telecommunications Industry Association*, cuyo origen es estadounidense) y la ARIB (*Association of Radio Industries and Businesses*, basada en Japón). Estos miembros son a su vez agrupaciones de empresas y organismos pertenecientes al ámbito de las telecomunicaciones como puede ser Telefónica, Huawei o Ericcson. Este grupo permanece en activo hoy día desarrollando las tecnologías móviles del futuro. En cualquier caso, retornando a la 3G, esta sería la generación donde se consolidaría el acceso a internet a través de los UE's, dispositivos que evolucionaron y se convirtieron en *smartphones* (destacando en este proceso la empresa canadiense "*Blackberry*" como el primer teléfono de esta clase y que sentó las bases de los actuales modelos)

permitiendo el manejo de aplicaciones de red en un principio básicas como el correo electrónico o un navegador web bastante rudimentario [11].

Esta nueva generación trataría de imitar, como ya ocurría con el GPRS, a las redes fijas que por aquel entonces habían incorporado la tecnología ADSL (*Asymetric Digital Subscriber Line*) que trajo velocidades de transmisión de datos de 2 Mbps a los hogares [12]. Para igualar esta velocidad se dispuso un cambio en la modulación empleada pasando a Q-PSK (*Quadrature Phase Shift Keying*, técnica de modulación que emplea el desfase para codificar sus símbolos) para el enlace descendente y para el ascendente una B-PSK (*Binary Phase Shift Keying*, similar a Q-PSK, pero con menos símbolos). De igual forma, el acceso al medio cambió a WCDMA (*Wideband Code Division Multiple Access*) sistema que emplea códigos ortogonales permitiendo la reutilización del espectro [13].

Otras mejoras implementadas en esta generación son el *soft-handover* (conexión simultánea a dos antenas para suavizar el relevo de servicio) y los receptores RAKE que permite sumar de manera constructiva las señales llegadas por multitrayecto aumentando la SNR (*Signal to Noise Ratio*) del enlace radio. El núcleo de la red no sufrió modificaciones sustanciales [7].

La RAN avanzó hacia la llamada UTRAN (*Universal Terrestrial Radio Access Network*). Las BS pasan a denominarse *Node-B*. El conjunto de toda la red se denominó UMTS (*Universal Mobile Telecommunications System*). La red 3G continuaría creciendo en velocidad con tecnologías como el HSDPA (*High-speed Downlink Packet* Access) y el HSUPA (*High-speed Uplink Packet Access*) con velocidades de hasta 14,4 Mbps [14, 15]. El pilar fundamental de estas mejoras es el mecanismo de *scheduling* que permite una distribución optimizada de recursos y de frecuencias de uso. Luego vendrían nuevas formas de modulación como el 16-QAM o incluso 64-QAM con constelaciones de símbolos mucho más grandes y por ello más velocidad. En la Figura 2-4 se pueden ver distintos tipos de modulación representados gráficamente y como cada vez se introducen más posibles símbolos para un dominio equivalente de señal portadora. Se incorporó un mecanismo de adaptación que permitía el empleo de modulaciones más robustas en ambientes en la que la SNR se veía comprometida. En esta línea aparecen las primeras antenas MIMO (*Multiple Input Multiple Output*), que son en realidad conjuntos de antenas con la capacidad de generar de manera coordinada mediante la suma de señales una única emisión resultante con mucha más direccionalidad. Poder elegir la dirección de radiación abrió la puerta a la multiplexación espacias pudiendo focalizar haces en distintos tipos de usuarios [16]. Todo ello mejoró aún más la velocidad. En resumen, el 3G trajo una mayor velocidad de conexión a unos dispositivos que ahora eran capaces de acceder a la red y de emplear aplicaciones básicas que no implicasen un consumo muy elevado de datos.

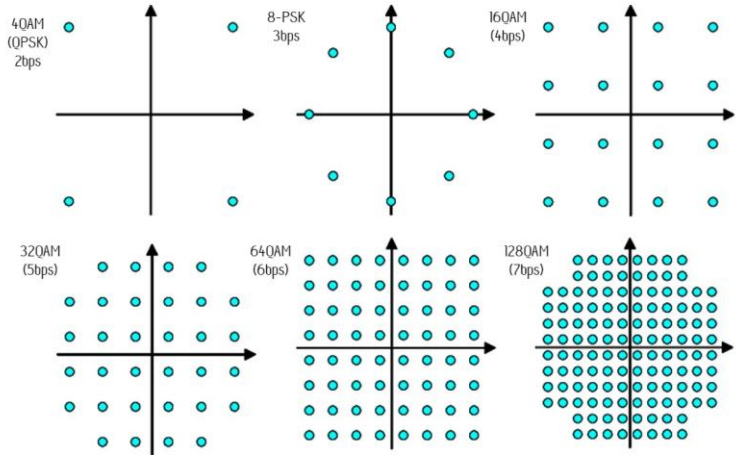

Figura 2-4: Constelaciones de símbolos asociadas a los distintos tipos de modulación y número de bits por símbolo (bps) [17]

En el año 2008 aparece la tecnología LTE (*Long Term Evolution*). Esta, en sus primeras versiones, no cumplía con los estándares fijados por 3GPP para ser una red 4G, aunque ya incorporaba mejoras notables tanto en ancho de banda como en latencia. Para ello se implementaron cambios tanto en la parte del enlace radio con mejoras también a las antenas tipo MIMO así como un aumento en la densidad de dispositivos que la red era capaz de sostener gracias a nuevas mejoras a la técnicas de acceso al medio [18]. Por último, se cambió por completo el núcleo de la red para dar paso a una nueva estructura: el *Evolved Packet Core* (EPC).

El EPC (cuya infraestructura viene esquemáticamente representada en la Figura 2-5 permitía un control de los *Node-B* desde el mismo a través del MME (*Mobility Management Equipment*), nodos ahora dotados de un cierto nivel de inteligencia y rebautizados como *eNode-B*. Esto permitía una mejor gestión de los recursos y un *handover* más rápido y suave pues las nuevas estaciones base tenían capacidad de gestionar el cambio de célula sin pasar por el núcleo. Se pasa de una red basada en la conmutación de circuitos a una *all-IP* con funcionamiento similar a una red de ordenadores [5, 17].

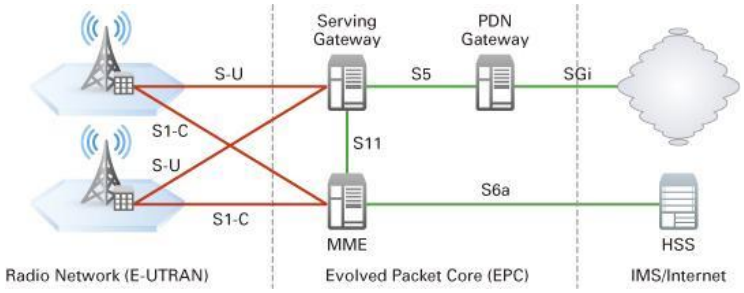

Figura 2-5: Estructura EPC [19]

Este nuevo núcleo de red (Figura 2-5) tiene una estructura renovada en la cual se distinguen como principales partes el MME (*Mobility Management Equipment*), el HSS (*Home Subscriber Server*), el SGW (*Serving Gateway*) unido al PGW (*Packet Data Network Gateway*) y por último el PCRF (*Policy and charging rules function*). El MME actúa como principal cerebro de la red al realizar el encaminamiento de paquetes en el nuevo paradigma en el que todo se basa en IP [18]. El HSS actúa de base de datos del sistema con las direcciones para encaminar los paquetes y por último PGW y SGW son los puertos de acceso a la red y de conexión a internet. Destacar que esta separación entre un puerto para el servicio y otro para los paquetes del usuario supuso también una innovación y un primer paso hacia el posterior paradigma del 5G en que se verá un diseño de red mucho más modular [18, 20].

El 4G no se consideraría como plenamente alcanzado hasta la entrada del LTE-Advanced en marzo del 2011, con un aumento del ancho de banda proporcionado, mejoras en la eficiencia espectral y antenas MIMO mejoradas. Además, incluyó una topología de red variable que permitía la existencia de celdas de diversos tamaños para satisfacer de manera más eficaz las necesidades de los usuarios. Es decir, en ambientes urbanos las celdas serían más pequeñas dado el mayor número de usuarios por unidad de área para así de esta manera conservar la ratio de densidad [21].

El ritmo de estas mejoras era cada vez más acelerado y no tardaría en aparecer una versión plus de la ya mejorada, el LTE-Advanced pro, apodado como el 4.5 G pues se aproximaba en muchas características técnicas al estándar 5G fijado por 3GPP en su *release 15* [22]. Latencia ultra baja (*Ultra Low latency communications* o ULCC), fiabilidad mejorada y velocidades de hasta 8 Gbps [23]. Todas estas características eran los prolegómenos de algo mayor pues como se describirá en el siguiente epígrafe el 5G no supone solamente una nueva evolución técnica si no un cambio sustancial en las redes móviles actuales [24].

Cabe mencionar como última mejora antes de dar paso al 5G algo que influye en la experiencia diaria de muchos usuarios, el *CS-Fallback* (*Circuit Switched*). Este sistema permite el empleo de los llamados *legacy systems* que no son más que las infraestructuras físicas y virtuales previas al 4G. Dado

el cada vez mayor aumento de la frecuencia empleada en la interfaz de radio, las pérdidas por propagación y por ello el alcance son menores. Es por ello que haciendo un cambio automático a generaciones anteriores se garantiza la comunicación con el usuario a pesar de la pérdida en calidad del servicio [23, 24].

2.1.3 La tecnología 5G

Como se mencionó en la sección 1.1 el 5G nace en el año 2017. Cada nueva generación ha traído cambios lo suficientemente relevantes como para constituirse como generación en sí y no como mera mejora. Tomando el ejemplo de las revoluciones industriales, la aparición de la máquina de vapor en las fábricas aumentó enormemente las unidades producidas. Sin embargo, la revolución no viene dada por este aumento en los números sino por los cambios internos que lo han permitido. De manera análoga, el 5G no es solo un aumento en el volumen de datos que los usuarios pueden recibir y transmitir.

No son pocos los analistas, expertos y empresas que afirman que la quinta generación de redes móviles constituye una auténtica revolución para la industria y el mundo. No obstante, hay que entender que no es la red móvil en sí lo que es una innovación sin precedentes sino lo que esta va a permitir hacer a los usuarios finales y a las industrias. En palabras del gerente de Telefónica en el área de seguridad y defensa, el 5G es un vector tecnológico, esto es, un medio que va a permitir que avances tan notables como la inteligencia artificial, los vehículos autónomos, la domótica avanzada, la gestión logística inteligente y ultra optimizada o las *Smart Cities* sean una realidad al alcance de la industria y los usuarios [24].

Comenzando por las prestaciones que ofrece el 5G, esta generación destaca por:

- Ancho de banda mejorado con velocidades de 20 GBps en bajada de datos y de 10 GBps en subida [25].
- Un tiempo de latencia de en torno a un milisegundo, conocido en la industria como latencia ultra baja.
- Mayor “densidad” pudiendo prestar servicio a un número mayor de terminales por unidad de área.
- Mayor eficiencia energética logrando que los dispositivos dependientes de baterías tengan una mayor autonomía.

Expertos del sector consideran la latencia como la mejora más relevante pues permitirá intercambios de información casi en tiempo real [24]. Su empleo en aplicaciones como el manejo de vehículos autónomos quizá sea el uso más evidente, pero existen otras. Volviendo a la idea del 5G como vector y utilizando un caso actual, supóngase un dron al que queremos incorporar una tecnología de reconocimiento de imagen en tiempo real. Sin el empleo de 5G existen dos posibilidades, la primera sería incorporar en el dron el hardware necesario para realizar esta función. La segunda sería establecer una conexión en remoto con un nodo que incorpore la infraestructura adecuada para dicho reconocimiento de imagen. Ninguna de las dos soluciones resulta óptima, pues, por una parte, la instalación a bordo puede implicar un aumento excesivo del peso del dron y con ello del consumo de energía; y, por otra parte, la segunda alternativa es muy dependiente de la calidad de la conexión y la velocidad que ofrezca, y aunque pueda ofrecer reconocimiento de imágenes difícilmente será en tiempo real.

Si se aplicase el 5G a este caso de uso, la segunda opción descrita sería viable gracias a las condiciones de conexión ofrecidas. Es más, otros componentes hardware del dron podrían ser “deslocalizados” logrando una reducción del peso, así como del consumo de energía por parte del ordenador de este. Este caso de uso evidencia las capacidades y el potencial tecnológico de esta nueva generación. Dicho potencial no es solo fruto de una mejora técnica o material, sino que nace de la

conjunción de estos avances y un nuevo diseño de infraestructura basada en software en lugar de elementos físicos.

Si se preguntase a un usuario medio que adquiere un teléfono nuevo, lo más probable es que este diga que su dispositivo recién comprado dispone de conexión 5G. Esta respuesta no es incorrecta pero no da fe del estado actual de las redes de telefonía. Actualmente el mundo se encuentra en una etapa de transición entre el 5G NSA ("*Non Stand Alone*") y el 5G SA ("*Stand Alone*"), siendo el primero una versión preliminar del segundo. El 5G NSA aprovecha las últimas mejoras técnicas en cuanto a estaciones base empleando antenas MIMO mejoradas que si se corresponden con los nuevos estándares fijados por 3GPP [22]. Sin embargo, el núcleo de la red sigue siendo el correspondiente a la tecnología LTE. Las principales compañías de telecomunicaciones están inmersas en el proceso de sustitución dichos núcleos de red, así como de la totalidad de las estaciones base a fin de alcanzar el 5G SA [26]. En las siguientes subsecciones se desarrollarán las características de este nuevo núcleo de red y del 5G SA a fin de exponer las razones que pugnan por convertir a esta generación en el hito más importante desde la aparición de las redes móviles en general.

2.1.3.1 Arquitectura de red 5G SA

Aunque se profundizará más en el siguiente apartado respecto a este hecho, para entender la arquitectura de una red 5G SA es necesario saber que esta forma parte de un nuevo paradigma pues es una SDN (*Software Defined Networking*) [27], es decir, una infraestructura basada en contenedores existentes dentro de un servidor. Lo que antes constituían distintos elementos de hardware cada uno realizando una función, en este caso es un único elemento de hardware conteniendo todas las funciones. El esquema general de la infraestructura viene representado en la Figura 2-6:

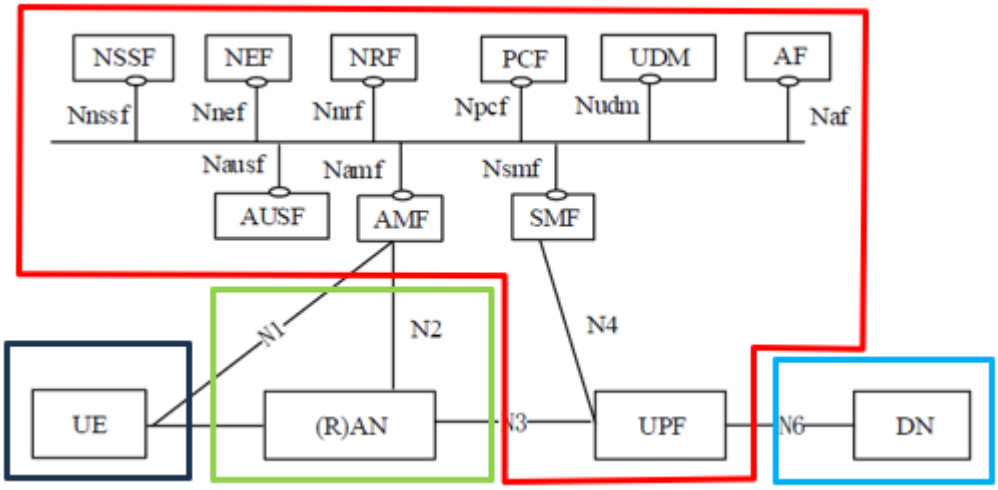

Figura 2-6: Arquitectura 5G SA [22]

A fin de facilitar la comprensión de la nueva estructura se han señalado una serie de divisiones en el esquema de la Figura 2-6. Las cuatro partes fundamentales son: el segmento de usuario (señalado en negro), la estación base (verde), el núcleo de red (en rojo) y finalmente el puerto de acceso a la red (en azul). Existe una segunda diferenciación en la cual todos los cuadros de la parte inferior (UE, RAN, UPF y DN) conforman el plano de usuario (mostrado en la Figura 2-7 pues son los que emplea de forma directa el usuario y a través de los cuales transita su información; y todos los cuadros de la parte superior que se corresponden con el plano de control de la red móvil. Destacar, que la función (R)AN no se corresponde con una única estación base sino con todas las de una zona geográfica determinada por la compañía de telefonía que proporcione soporte en esa zona [27, 28].

Figura 2-7: Esquema abreviado de la infraestructura de red [1]

Al igual que ocurre con el núcleo de red en otras generaciones, en 5G se puede dividir el mismo en distintos bloques funcionales. Más conocidos por el nombre de funciones de red (por razones que se expondrán más adelante), cada uno de los mismos vienen a cumplir las funciones básicas de cualquier red móvil (autenticación, control de tráfico, seguridad y asignación de recursos) junto con otros añadidos a fin de mejorar el servicio global. Estas funciones son:

- AMF: *Access and Mobility management function*, gestiona la movilidad y autenticación de los usuarios que acceden a la red.
- UPF: *User Plane function,* manejo de los datos y mantenimiento de la calidad de servicio de los usuarios. Realiza además funciones adicionales de seguridad
- SMF: *Session Management function*, gestión de las funciones de la capa de sesión.
- PCF: *Policy Control function,* establece las condiciones para cada usuario acorde al contrato que este tenga.
- UDM: *User Database Module,* base de datos con el registro de usuarios que pueden acceder a la red
- NSSF: *Network Slice Selection function*, asigna *slices* (concepto que será desarrollado adecuadamente en la Sección 2.1.3.4) según el tipo de UE que se conecte a la red.
- NEF: *Network Exposure function,* función que muestra la información de red necesaria para que terceras personas puedan generar productos y aplicaciones compatibles con la misma.
- NRF: *Network function repository function*, sirve como repositorio de información para el resto de las funciones de red del núcleo.
- AUSF: *Authentication Server function,* función que junto con AMF realiza la autenticación de los UE's que acceden a la red.
- AF: *Application function*, sirve de soporte para otras aplicaciones desarrolladas dentro del núcleo 5G, así como asegurar la calidad de servicio acorde a las necesidades de la aplicación y las condiciones del usuario.

Las reseñadas son las funciones básicas acorde a las funcionalidades descritas en los *release* 15 y 16 de la 3GPP [20, 29]. Faltan por definir los bloques ajenos al núcleo. En primer lugar, UE es el término que en otras generaciones se emplea para referirse al teléfono móvil, más en esta generación también se incorporan como otros tipos de UE los drones, los dispositivos IoT o los vehículos autónomos. La función de (R)AN es la encargada de la transmisión de ondas, así como de otros aspectos que se desarrollarán más adelante en la parte de interfaz radio del 5G.

2.1.3.2 Virtualización de los servicios de red: NFV y SBA

Como ya se ha mencionado en la subsección anterior, la nueva arquitectura de red está basada en software. Definida como una red SDN (*Software Defined Network)*, existe un único servidor en el cual se alojan todas las funciones de red (NF, *Network functions*). Es por ello por lo que se habla de que estas funciones han sido virtualizadas (*Network function virtualization*). Los beneficios de este cambio de paradigma son muchos. Por una parte, constituyen una liberalización del sector pues de esta forma

empresas más enfocadas al desarrollo de software, pero sin capacidad para el desarrollo o compra del hardware propio a escala industrial o comercial pueden crear servicios y productos ya sean estos para el usuario o para las empresas de telecomunicaciones mantenedoras de la arquitectura de red 5G. Es un modelo que favorece la innovación [30]. Por otra, para las compañías del sector es un cambio a un sistema más flexible, escalable y modular. El mantenimiento es más sencillo, así como la capacidad para recibir actualizaciones y mejoras pues estas no están atadas a elementos materiales [27].

El 5G es en definitiva una tecnología *Cloud Native*. Está pensada para funcionar y desarrollarse con una estructura de contenedores y orquestadores de los mismos (con aplicaciones como *Dockers* o *Kubernetes*). Un contenedor es una aplicación desarrollada dentro de una máquina virtual cuyo único cometido es soportar dicha aplicación. Esta forma de diseño permite optimizar los recursos, la modularidad y la compatibilidad con cualquier sistema que pretenda usarla [31]. En la Figura 2-8 el contexto en que se encuentra un contenedor dentro de los sistemas de una computadora. En este contexto, un orquestador permite la coordinación y comunicación entre los distintos contenedores a fin de mejorar el servicio final al usuario.

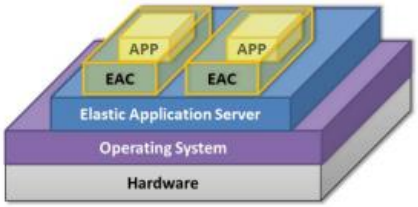

Figura 2-8: Diseño de contenedor elástico de aplicación [31]

El fin último de todo es crear una arquitectura que sea basada en el servicio (SBA, *Service based architecture*) la cual permita un grado alto de compatibilidad entre los distintos UE's, tipos de usuario y las aplicaciones de red. Es por ello que las NF's en su mayoría están desarrolladas en contenedores con la AMF actuando de orquestador de las mismas [32].

2.1.3.3 La interfaz radio: beamforming y conformado de haz

El nuevo estándar de interfaz de radio conocido como *New Radio* es probablemente la característica más conocida para el público en general. El uso de frecuencias más elevadas llegando incluso al uso de ondas milimétricas va a conllevar un incremento en el ancho de banda de la señal y por ende de la velocidad de transmisión. Sin embargo, no se trata simplemente de aumentar la frecuencia *per se* sino de optimizarla al caso de uso concreto [33].

	5G Features	Benefit	Use Case
	Enhanced mobile broadband (eMBB)	Fast exchange of high amounts of data	AR, VR, Telemedicine, Drones
	Ultra-reliable and low latency communications (URLLC)	Low response time	Manufacturing Automation, Robotic Control, Autonomous driving
	Massive machine type communications (mMTC)	Connectivity of a large number of devices	Infrastructure Management and Maintenance, Connected Machinery, Smart Sensors

Figura 2-9: Casos de uso acorde a los tres grupos de frecuencia empleables según el estándar NR [33]

Existen tres casos principales, los cuales se recogen en la Figura 2-9: el primero con un ancho de banda mejorado, otro con comunicaciones ultra confiables de baja latencia y por último uno con comunicaciones masivas entre máquinas. Sumados a estos, existen tres escenarios habituales, un entorno rural con edificaciones dispersas, un entorno urbano medio y un entorno urbano con gran densidad de usuarios y de obstáculos. Para enfrentar las diversas situaciones existen tres grupos de frecuencias principales, por debajo de 7 GHz cuyo uso es el más habitual logrando los mayores alcances, entre 7 y 24 GHz siendo el término medio y por último por encima de 24 GHz, región de las ondas milimétricas, logrando muy buen ancho de banda y una baja latencia [7].

En cuanto a las técnicas de acceso múltiple al medio empleadas se usa OFDMA (*Orthogonal Frequency Division Multiple Access*) para el enlace de bajada y se emplea SC-OFDMA (*Single Carrier-OFDMA*) para el de subida del dispositivo a la estación base. Ambas técnicas basadas en la asignación de códigos ortogonales solo se diferencian en el número de portadoras empleadas que en el caso del enlace de subida es una única portadora lo que simplifica el hardware del dispositivo [34]. Las modulaciones utilizadas emplean mecanismos adaptativos como los ya explicados en el caso del 4G, es decir, desde QPSK hasta 256-QAM tanto en enlace de subida (*uplink*) como de bajada (*downlink*) siendo esta última la modulación óptima de trabajo [35].

La última mejora sustancial al enlace radio es el empleo de MIMO masivo. Como ya se ha expuesto previamente, MIMO son las siglas de *Multiple Input Multiple Output* en referencia al hecho de una antena 5G está en realidad compuesta por múltiples antenas formando todas ellas un panel con distintas celdas siendo cada una de ellas una de estas antenas (estos paneles son también conocidos con el nombre de *arrays*). De esta forma la señal emitida es la composición de la generada de manera coordinada por todas las antenas individuales. A mayor número de antenas, mayor número de posibles combinaciones. En la Figura 2-10 se muestra la evolución de las antenas MIMO 2x2 con cuatro celdas hasta llegar al MIMO masivo que pretende ser el estándar para cualquier estación base 5G. La composición de las señales permite conformar y direccionar haces cuya potencia e intensidad de señal es superior logrando con ello mayores alcances y eficiencia energética. Esta capacidad se conoce como *beamforming* [35].

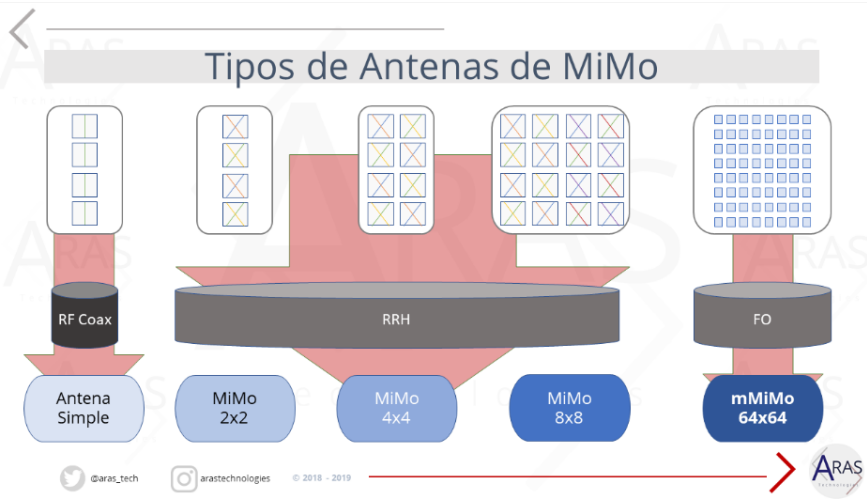

Figura 2-10: Evolución y modelos de MIMO (2x2, 4x4 y Masivo) [36]

Sería inútil conformar el haz en una dirección en la cual no exista un dispositivo, es por ello por lo que el *beamforming* sigue un protocolo de pasos que le permite detectar las direcciones en las cuáles ha de concentrar el haz. En primer lugar, se realiza un barrido denominado como *Beam Sweeping* con un haz determinado por defecto en cual se localizan las direcciones de interés, es decir, aquellas en las que se localizan dispositivos. Realizado el primer barrido se realiza un segundo focalizado en las zonas determinadas de interés con un haz mucho más estrecho capaz de definir mejor la situación general de dispositivos en torno a la estación base. El proceso completo viene ilustrado en la Figura 2-11 y la Figura

2-12. La precisión dependerá del tipo de MIMO empleado ya que a menor número de antenas la direccionalidad y conformado de haz será menor [35, 37].

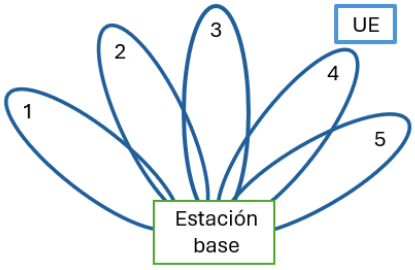

Figura 2-11: Proceso de *Beam sweeping,* elaboración propia

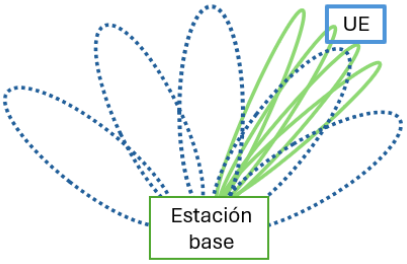

Figura 2-12: Proceso de *Beam refinement,* elaboración propia

Por otra parte, se diferencian dos tipos de *beamforming*: *Switched beamforming* y *Adaptative beamforming*. El primer método se limita a encontrar las direcciones de interés con un único barrido y genera haces iguales distribuidos de la forma más equitativa posible. El *beamforming* adaptativo si realiza el proceso completo generando haces optimizados en cada dirección. El segundo tipo, siendo técnicamente superior, presenta los inconvenientes de tener mayor dificultad de implementación y coste en cuanto a infraestructura [37]. Los esquemas de radiación resultantes se representan en la Figura 2-13.

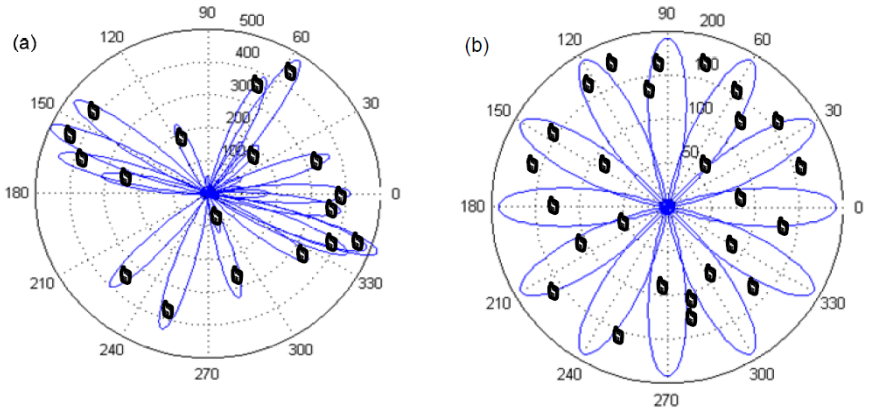

Figura 2-13: Diagramas de radiación de los distintos tipos de *beamforming*: *Adaptative beamforming* (a) y *Switched beamforming* (b) [37]

2.1.3.4 Network slicing: segmentación de la red acorde a las necesidades

En la sección anterior referente a la interfaz de radio, se mencionaron diversos casos de uso representados en la Figura 2-9. Se mencionó como se puede variar la interfaz de radio para adaptarse mejor a cada tipo de uso de la red. Sin embargo, lo más habitual será que en un mismo contexto coincidan diversos dispositivos teniendo cada uno de ellos unas necesidades distintas. Por ejemplo, pueden

coincidir en el mismo contexto un móvil reproduciendo un contenido audiovisual que necesite priorizar el ancho de banda disponible a la vez que hay un vehículo autónomo que precise de latencias bajas. La solución ideal pasaría por conseguir que las antenas de la red ofrezcan prestaciones distintas a cada clase de UE's. El *Network slicing* es la solución a nivel software para lograr esta diferenciación de servicios [38]. La idea en síntesis es crear distintas subredes a nivel lógico con un comportamiento similar a una estructura de redes virtuales (VLAN) [39] pero con el añadido de ser capaces de asignar recursos de red, capacidades y características de conectividad únicas a cada uno de los *slices* o segmentos de red generados todo ellos en con una infraestructura física común [38]. Esta idea se manifiesta en la Figura 2-14.

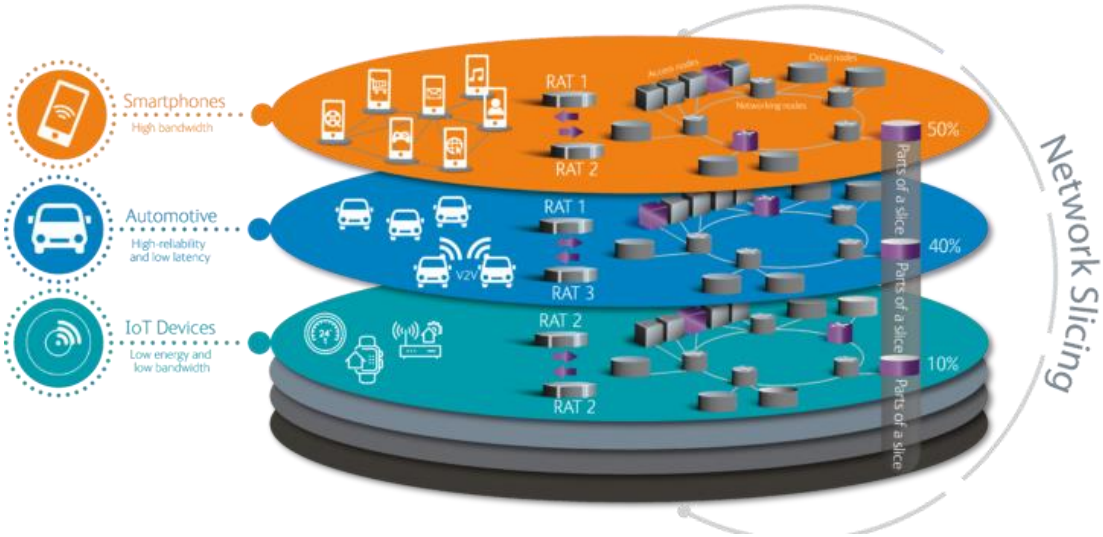

Figura 2-14: Esquema de Network Slicing [40]

Los conceptos previamente explicados en el aparto de virtualización de servicios son ahora los elementos posibilitadores del *Network Slicing*. Al trabajar con un paradigma SDN y con funciones de red virtualizadas con componentes modulares (Conocidos como *Network function components* o NFC's) se pueden generar de manera sencilla particiones o copias de dichas funciones y servicios a fin de generar los distintos segmentos de red. Los eNode-B recibirán solicitudes de los distintos núcleos de red virtuales generando de facto la compartición de recursos físicos a nivel enlace radio. Lo más habitual es realizar un reparto mediante un mecanismo de *scheduling* que reparte el tiempo de uso entre las distintas particiones de la red. La otra opción es realizar un *slice* que también incluya a la estación base y sus recursos, aunque esta opción no es viable en todas las clases de RAN [41, 42].

La última pieza que falta para completar cada subred virtual es la de los UE's. Una de las funciones de red básicas de componentes del núcleo de red 5G es la NSSF. Su forma de proceder será asignar un NSSAI (*Network Slice Selection Assistance Information*) a cada dispositivo en su primera interacción con la red. De esta forma la información proveniente de ese terminal de usuario vendrá "etiquetada" de forma que se transmita y gestione en el segmento de red que se le haya asignado. Inicialmente la NSSAI será genérica dando al dispositivo información sobre las características de los segmentos disponibles, lo cual permitirá decidir cuál será más conveniente para el tipo de servicio que requiere ese usuario [42].

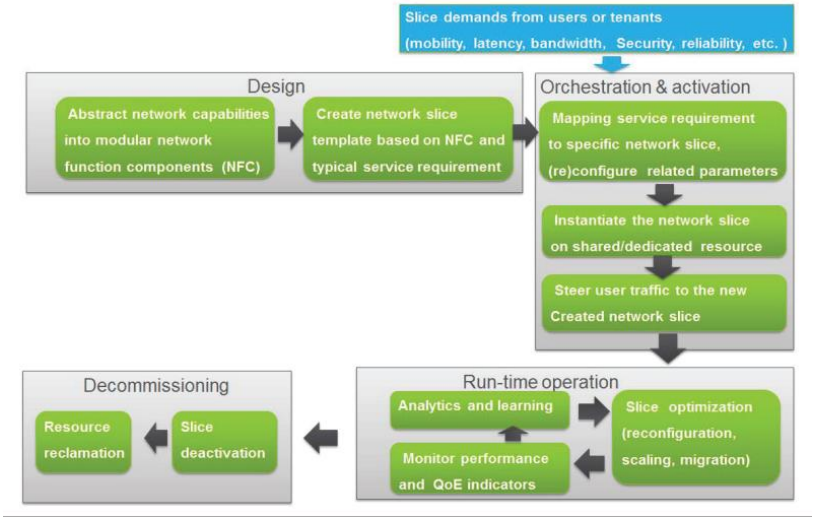

Figura 2-15: Ciclo de vida de un segmento de red [41]

La gestión de una red segmentada es más compleja pese a las características de modularidad y flexibilidad que presenta. La Figura 2-15 muestra el ciclo de vida de un segmento de red. Este comienza con una fase de diseño donde los administradores de la red móvil exploran los posibles usos de la red en un determinado contexto para después crear y asignar los recursos adecuados a cada uno. En segundo lugar, cada uno de los segmentos ha de ser activado para lo cual es necesaria la existencia de un orquestador que introduzca el nuevo segmento de red reasignando los recursos disponibles y reasignando los dispositivos ya presentes en la red al segmento más adecuado. El orquestador mantendrá una vigilancia activa del desempeño de los *slices* de tal forma que sea capaz de optimizar su funcionamiento ajustando sus parámetros o cambiando la asignación inicial de recursos. Por último, en caso de desactivar un segmento, sus recursos pasan de nuevo a ser comunes y entrarán a trabajar donde su empleo sea óptimo.

Es necesaria por tanto la existencia de una función de red que ejerza de orquestador para el diseño y despliegue de la segmentación de red que, permitiendo la mencionada asignación dinámica de recursos, así como la coordinación entre segmentos. La complejidad de diseño y mantenimiento que puede acarrear la segmentación de red tiene sin embargo beneficios notables. Aparte de la ya mencionada adaptabilidad a los casos de uso, otras ventajas son el aumento de la seguridad pues esta también se modificará acorde al escenario de empleo, la mejora en la QoS (*Quality of Service*) tratando de alejarse del modelo *best effort*; y por último es catalizador de innovación y experimentación al permitir crear *slices* de prueba sin comprometer la red móvil al completo [42, 43]

2.1.3.5 Edge computing y cloud computing

Dos de las principales características de cualquier dispositivo inteligente son la capacidad de computación, así como la capacidad de almacenamiento y manejo de datos. Aunque es cierto que la ley de Moore parece cumplirse [44] y cada vez dispositivos con menor tamaño físico son capaces de tener mayores capacidades de cálculo y memoria; la consumición de energía que estas capacidades conllevan así las posibles limitaciones a la hora de manejar los volúmenes de datos que se prevén en las aplicaciones 5G pueden suponer un factor limitante para los usuarios. Pruebas de ello en es que en 2017 había 7,5 billones de dispositivos creciendo exponencialmente en los años posteriores. En 2020 la cantidad estimada de datos generada por estos dispositivos fue de en torno a 500 exabytes [45]. Aplicaciones como las de realidad aumentada, de realidad virtual o aplicaciones de telemedicina y respuesta ante emergencias no harán sino incrementar más aún estos datos de tráfico de datos.

El concepto de *cloud computing* nace previamente de la llegada del 5G. La idea es aprovechar el acceso a la red de los dispositivos para deslocalizar volúmenes de datos y procesos computacionales que por su tamaño excedan la capacidad de dicho dispositivo [46]. De esta forma los terminales de usuario pueden extender sus capacidades más allá mediante la conexión a servidores en la nube y cubrir estas dos necesidades [47]. El siguiente paso en el proceso evolutivo de esta tecnología es el denominado *edge computing*, cuyo propósito principal es acercar de manera física los servidores de servicios en la nube a los usuarios. De esta forma el número de nodos a recorrer y la distancia son menores disminuyendo la latencia y por tanto el tiempo de "computación" que percibe el usuario. Es probable que estos servidores *on the Edge* no sean equivalentes en potencia y memoria a un servidor tradicional en la nube, pero de cara a las necesidades del usuario promedio serán suficientes [48]. Ambas disposiciones de red vienen reflejadas en el esquema de la Figura 2-16.

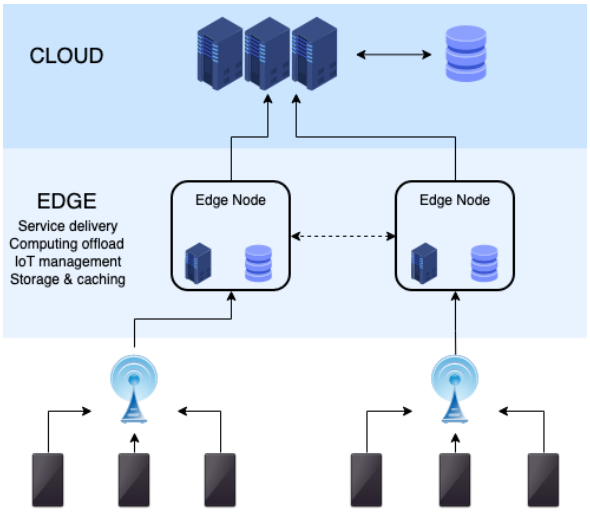

Figura 2-16: Esquema de *Cloud* y *Edge computing* [49]

En el caso del 5G, el servidor en el borde está localizado en el núcleo de la red. Sus funciones no se limitan al plano de usuario, sino que también toma una parte activa en las funciones del plano de control mejorando su funcionamiento. La latencia ultra baja del 5G es fruto tanto del RAN como del *Edge Computing* logrando tiempos de en torno a 1 ms los cuales son necesarios para el correcto funcionamiento de aplicaciones en las que el factor temporal es crítico como en el caso de la Realidad Aumentada o de la Realidad Virtual [45]. Estos servidores también son capaces de mejorar la QoS global puesto que con su capacidad de comprensión del entorno en cuanto a red y dispositivos conectados a la misma en esa región geográfica permiten realizar una optimización del tráfico [50]. Por último, esta tecnología también aporta mejoras en el RAN, apoyando en el proceso de *beamforming*; así como en el ámbito de la seguridad proveyendo de otra capa más de protección [45, 51].

2.2 Concepto de burbuja táctica 5G

En esta sección se describirán las necesidades de una red de comunicaciones de propósito militar, aunque también aplicable al campo de las fuerzas de seguridad de un país. A continuación, se pasará a definir y exponer el estado del arte de las burbujas o nubes tácticas 5G, así como sus aplicaciones potenciales.

2.2.1 Redes de comunicaciones militares

En la reflexión sobre la motivación para la realización del presente trabajo (Sección 1.1) se comentaron algunas de las características necesarias para establecer una red de comunicaciones de

propósito militar. También se mencionó el nombre con el que habitualmente se las suele nombrar, redes C2. Sin pretender ser redundante ni entrar en una gran digresión conceptual sobre cómo ha de ser la red ideal sí que se han de comentar las virtudes fundamentales de la misma [52, 53]:

- Rapidez: velocidad en la transmisión de la información entre unidades para permitir al correcta coordinación y ejecución de las acciones a realizar por las fuerzas propias.
- Seguridad: los mensajes transmitidos solo han de ser recibidos por las personas designadas como receptores evitando de todas las formas posibles que la fuerza oponente logre interceptar y acceder al contenido de esos mensajes. De igual forma habrá que tratar de evitar la modificación o degradación de la información.
- Confianza: certeza sobre el correcto funcionamiento de la red sin que existan fallo de conexión que impidan la transmisión. Va unido a la precisión en la transmisión pudiendo producirse fallos de tipo humano o de origen aleatorio.
- Densidad de la información: el contenido de los mensajes transmitidos ha de contener el mayor grado de información tanto de manera ascendente (proporcionar al mando el mayor número de datos relevantes posibles) como descendente (órdenes claras a los escalones subordinados). Habitualmente la forma de lograr esto es mediante la concisión especialmente en las comunicaciones por voz. Sin embargo, cuando se trata de transmitir datos estos han de ser de la mayor calidad posible o son pesados *per se* con lo que ello puede implicar de cara al volumen de datos a transmitir.
- Alcance: las partes de la comunicación se capacitan a grandes distancias o en ambientes con gran cantidad de obstáculos.

En la actualidad, y pese a existencia de métodos alternativos y complementarios, el sistema por excelencia para establecer una red, son las radiocomunicaciones. Cabe aclarar que esto es en los escenarios de operaciones o adiestramiento para la realización de acciones de carácter táctico. A nivel administrativo lo más habitual es encontrarse con redes de conmutación por cable de fibra constituyendo una infraestructura privada con posibles salidas a la red pública. En ciertos casos se pueden establecer nodos con ubicaciones externas a esa red privada mediante un procedimiento de tunelización de datos como por ejemplo la tunelización por protocolo IP. Este procedimiento ha de cumplir con los servicios de confidencialidad y seguridad exigidos a nivel gubernamental de cada país [54, 55].

Las ondas de radio son un sistema rápido en lo que se refiere a propagación por el medio (de hecho, es el más rápido al viajar a la velocidad de la luz). Dentro de la llamada onda portadora es donde se codifica la información de diversas formas y empleando distintas técnicas de composición de ondas (un ejemplo de ello se puede encontrar en la sección 2.1.2 cuando se mencionaron diversas técnicas de modulación) El detrimento a la velocidad se encuentra en este plano ya que se ha de generar e interpretar la información. La seguridad que se quiera implementar en una red de comunicaciones constituye una capa más con la que se envuelve el mensaje y por ello puede afectar a la velocidad con la que se interpreta dicho mensaje. La principal de forma de lograr la seguridad es mediante mecanismos de cifrado, usualmente mediante clave privada, la cual es distribuida a todas las unidades participantes en la red. Otra forma de lograr esquivar la acción de interceptación es mediante cambios de frecuencia, llegando a haber dispositivos que efectúan saltos automáticos de una frecuencia a otra logrando un patrón impredecible para un tercero que no posea la clave que crea esos saltos.

El alcance en gran medida está condicionado por la frecuencia empleada. Existen diversas formas de propagación, pero de forma reduccionista a más frecuencia menos alcance, pero más ancho de banda y con ello más cantidad de información por segundo transmitida. En comunicaciones por voz esto se refleja en la nitidez de esta al recibirse. La alternativa para lograr grandes alcances es recurrir a las comunicaciones por satélite. Aunque existen grandes limitaciones especialmente en cuanto a ancho de banda y velocidad (la mayoría de los satélites de comunicaciones militares se encuentran en órbita GEO

(*Geostationary Earth Orbit*) al lograr una posición relativa respecto a la tierra estática. La altura de esa órbita es de 35786 km por lo que la latencia mínima asociada a la propagación será de 0,238 s, existen compañías que están logrando grandes avances en este terreno [56]. En concreto la compañía Starlink[1] ha aumentado de forma considerable el ancho de bando y la velocidad ofrecida a los usuarios mediante constelaciones de satélites en órbita LEO (*Low Earth Orbit*, con una altura de unos 500 km de altura [57]). En los rangos de frecuencia de interés para el trabajo la propagación es por la línea de visión directa, aunque la refracción causada por la atmósfera y la potencia de transmisión pueden aumentar el alcance más allá del horizonte en ciertas ocasiones. La Figura 2-17 concentra en una sola imagen todos los tipos de comunicación y redes que pueden existir vinculadas al teatro de operaciones.

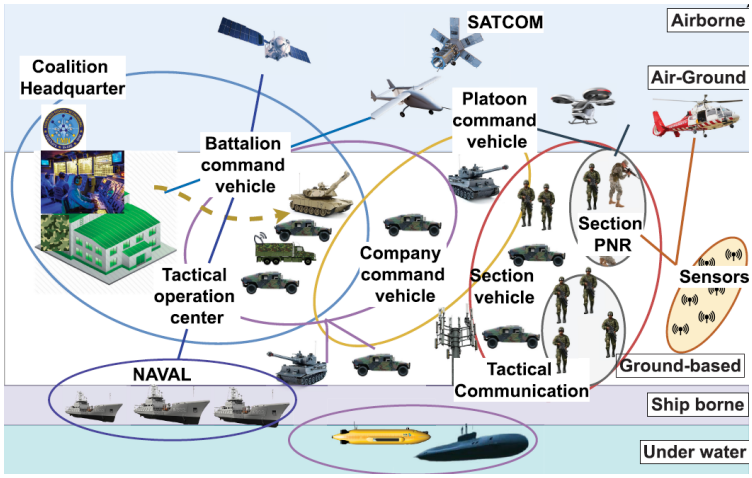

Figura 2-17: Esquema con las diferentes redes de comunicaciones y los enlaces entre ellas [52]

Para concluir, cabe reseñar que en los teatros de operaciones actuales existen también actores no humanos que participan y necesitan de las redes de comunicación. Los vehículos no tripulados y sensores son los ejemplos más habituales. Estos elementos incorporan en muchos casos armas con las que ejercer el poder ofensivo con letalidad. El nivel de control de sobre estos elementos ha de ser alto no solo por la necesidad técnica de manejo si no por sus implicaciones éticas [58] (implicaciones que serán más exploradas en el Anexo II: Reflexiones Éticas y Sociales). Es por ello que las comunicaciones con estos nodos son críticas a todos los niveles pues aparte de rapidez para asegurar el manejo en tiempo real la seguridad ha de ser lo suficientemente robusta para asegurar que el dron no sea secuestrado por el enemigo.

2.2.2 Definición burbuja táctica 5G

El concepto de burbuja táctica 5G se refiere al uso y establecimiento de una red móvil 5G SA independiente de la red a fin de establecer una red de mando y control. Esta red es privada y en principio aislada del resto de la red, aunque esto puede modificarse a fin de prestar ciertos servicios que se consideren. Por ende, a continuación, se repasarán los dos casos de uso que se prevén como más habituales [59].

El primero estaría más enfocado a la seguridad o la logística y es la instalación de una infraestructura fija en una base, edificio, arsenal o almacén de material. Esta clase de escenario es ideal para el uso de las bandas de frecuencia milimétricas, las más altas de las que dispone el 5G. El objetivo es dotar de

[1] https://www.starlink.com/

sensores y automatizar los procesos (seguridad y logísticos principalmente) mediante un despliegue masivo de dispositivos IoT [52, 60].

El segundo y en cual pretende centrarse el presente trabajo en su desarrollo es el establecimiento de una infraestructura que preste una red de mando y control a unidades de combate que estén realizando una operación militar o cívico militar del tipo que sea. Un ejemplo de ello podría ser la operación Dédalo-23 realizada por la Armada que contó con el despliegue de una burbuja 5G experimental [61]. En este caso se despliega un núcleo portátil conectado a una serie de antenas que prestan servicio a los distintos UE's dentro de los cuáles encontramos una gran variedad de servicios. Estos van desde elementos más habituales en una red móvil como pueden ser teléfonos móviles o *tablets* (debidamente adaptadas para ser más resistentes físicamente y a nivel ciber), drones, equipos de telemedicina hasta algunos muy novedosos como gafas de realidad virtual. El núcleo de la red y las antenas están pensadas para ser rápidamente desplegables o venir preinstaladas a bordo de una plataforma como puede ser un buque de guerra o un vehículo de combate, siendo la primera especialmente idónea por los alcances logrados [59] en el mar y resultar el 5G un complemento ideal a las capacidades de proyección de fuerza de las que dispone una fuerza naval [62].

Profundizando en la segunda aplicación de la burbuja 5G, se ha estudiado su uso en tres contextos geográficos: uno plenamente marítimo, uno terrestre y otro litoral.

La tecnología se haya actualmente en pleno desarrollo y a nivel internacional es incierto el grado de desarrollo de posibles proyectos de adaptación en países del entorno OTAN o europeo.

2.3 Herramientas de simulación de redes

En la actualidad el despliegue de cualquier estructura de red está sometido a un proceso de validación previa a fin de reducir costes finales en la fase de pruebas y optimizar el diseño. Por ello y antes de cualquier elaboración y testeo de algún prototipo físico se suele recurrir a *softwares* de simulación. A continuación, se mencionan algunas de las herramientas de simulación de redes 5G más destacadas [63, 64, 65]:

2.3.1 NS-3 (Network Simulator 3)

NS-3 es una herramienta de simulación de código abierto ampliamente utilizada para la investigación en redes (su logo está representado en la Figura 2-18). Permite simular diversos protocolos de red y escenarios en entornos realistas. Basada en el uso de C++, incorpora así mismo gran cantidad de módulos desarrollados por personal propio o terceros entre los cuáles encontramos el módulo 5G LENA NR de Centro Tecnológico de Telecomunicaciones de Cataluña que será el que se emplee en la fase de desarrollo.

Figura 2-18: Logo NS-3

2.3.2 OMNeT++ (Objective Modular Network Testbed in C++)

OMNeT++ es una plataforma modular para simulación de sistemas de comunicación. Proporciona una arquitectura extensible y es popular en la investigación de redes y sistemas distribuidos. De las extensiones que permiten la simulación de 5G destaca INET Framework (Cuyo logo se muestra en la Figura 2-18**¡Error! No se encuentra el origen de la referencia.**). Este módulo es particularmente útil a la hora de probar nuevos protocolos de red y así como escenarios novedosos para el despliegue de una red móvil.

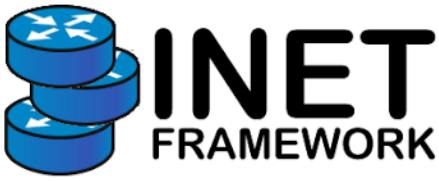

Figura 2-19: Logo de la extensión 5G para OMNeT++ [66]

2.3.3 QualNet

Qualnet es un simulador de eventos discretos paralelos que emula los diversos tipos de protocolos, interfaces de radio y dispositivos existentes hoy en día. Gracias a su capacidad para ejecutar varios procesos en paralelo logra unos tiempos de simulación menores. Es además capaz de combinar escenarios con elementos heterogéneos y con diversos tipos de conexiones entre ellos. Su logo y el del grupo francés al que pertenece se representan en la Figura 2-20**¡Error! No se encuentra el origen de la referencia.**.

Figura 2-20: Logo de QualNet perteneciente al grupo francés QualNet [67]

2.3.4 OPNET (Riverbed SteelCentral Modeler)

OPNET es parte de *Riverbed SteelCentral Modeler* un software de simulación más extenso. Ambas herramientas en conjunto son capaces de simular y obtener el rendimiento de una red 5G. *Riverbed* (cuyo logo queda recogido en la Figura 2-21**¡Error! No se encuentra el origen de la referencia.**) de igual forma otros productos propios que aportan capacidades adicionales para desarrollar distintas simulaciones.

Figura 2-21: Logo de la empresa *Riverbed* propietaria de OPNE y de *Riverbed SteelCentral Modeler* [68]

2.3.5 NetSim

NetSim es una herramienta de simulación de pago desarrollada por la empresa india Tetcos. Es capaz de simular una amplia variedad de redes entre las cuales se incluye una red 5G. Permite asimismo la simulación de todas las capas del modelo de internet incluyendo la capa de aplicación. Las simulaciones desarrolladas permiten un gran nivel de detalle pudiéndose ajustar la arquitectura de red o el modelo de movilidad de los usuarios. Su logo se adjunta en la Figura 2-22.

Figura 2-22: Logo del software de simulación NetSim perteneciente a la empresa *Tetcos* [69]

2.3.6 MATLAB/Simulink

MATLAB/Simulink es un programa de modelado y simulación ampliamente utilizado en diversos campos, incluidos los de las comunicaciones y las redes. Se puede utilizar para simular sistemas de red complejos además de manejar grandes volúmenes de datos y su representación (como en la mostrada en Figura 2-23) dado su funcionamiento básico basado en matrices. Su extensa comunidad de usuarios y popularidad en el mundo académico son las causas de que disponga de una gran cantidad de módulos específicos tanto de simulación de redes, QoS y 5G.

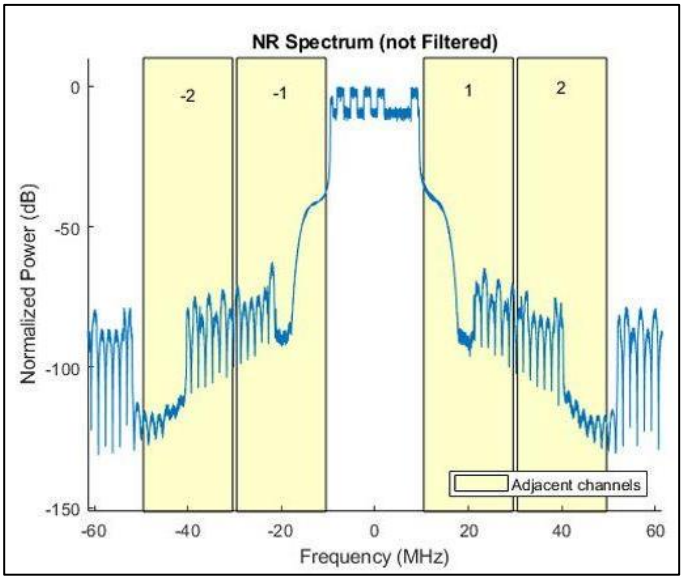

Figura 2-23: Gráfica ejemplo del módulo 5G toolbox de Matlab [70]

Estas herramientas ofrecen una variedad de capacidades y enfoques para la simulación de redes, desde simulaciones detalladas de bajo nivel hasta entornos virtuales de alto nivel. En muchos casos estas simulaciones se centran en la interfaz física de las conexiones y los parámetros de transmisión alcanzados en las distintas conmutaciones de paquetes sin entrar en el funcionamiento interno de los servidores.

2.4 Geoposicionamiento en 5G

En la presente sección se desarrollará la teoría general detrás del posicionamiento de dispositivos de usuario mediante el empleo de señales 5G, los tres principales métodos o formas de extraer los datos necesarios para llevar a cabo dicho proceso y por último se analizarán las novedades que implementa el 5G para mejorar su precisión respecto a generaciones anteriores.

2.4.1 Aspectos generales del posicionamiento por señal de radiofrecuencia

Uno de los usos de la tecnología 5G es emplearla como herramienta de geolocalización ya sea de forma sustitutiva o complementaria a la señal GPS. Las técnicas empleadas pueden variar, pero en síntesis se basan en un mecanismo similar al de los satélites GPS, los cuales transmiten su posición y los datos necesarios para determinar un subespacio de posibles posiciones geográficas. El proceso se realiza con varias estaciones base a fin de obtener al menos 3 subespacios tridimensionales cuya intersección nos da un punto que es la localización final. En la Figura 2-24 se puede ver una representación gráfica del proceso sobre el plano. Esta figura se corresponde con uno de los dos grupos en los que se pueden agrupar los métodos de posicionamiento, la trilateración. El otro método es la triangulación. Basados ambos en la intersección de subespacios uno emplea distancias para generar subespacios tipo esfera y el segundo usa ángulos para generar subespacios tipo recta [71, 72]. La Figura 2-25 representa de forma esquemática ambas clases de posicionamiento para el caso del posicionamiento mediante el empleo de satélites.

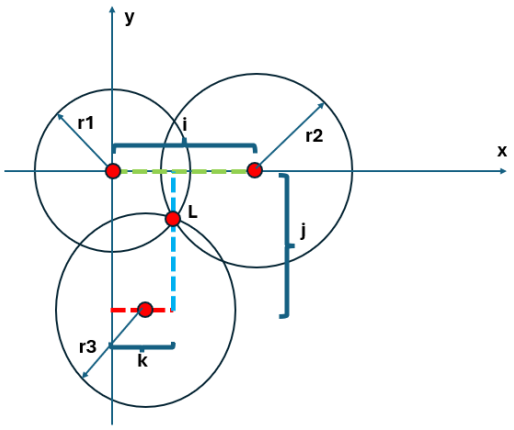

Figura 2-24: Esquema algoritmo de trilateración en 2 dimensiones. Autoría propia.

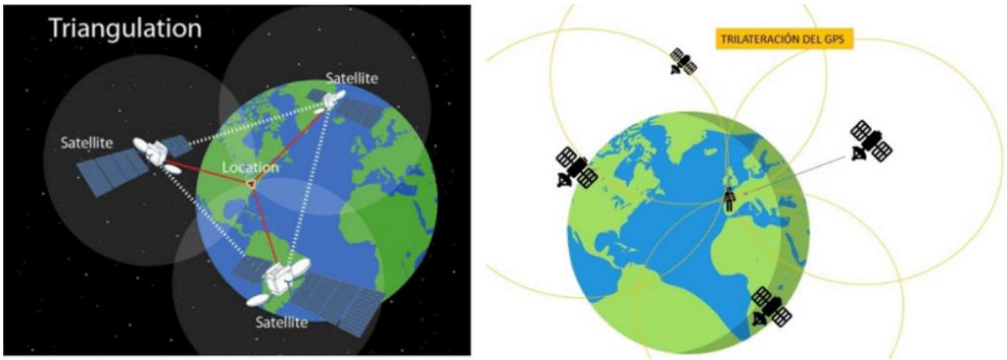

Figura 2-25: Triangulación y trilateración aplicadas al caso de posicionamiento por señal GPS [71]

Cualquier aplicación de posicionamiento basada en este tipo de métodos tendrá dos partes: la primera es la parte de medición y obtención de los datos que permitirán obtener la posición y la segunda es el procesamiento de dichos datos mediante un algoritmo matemático adecuado que permita obtener las soluciones lo más precisas posibles. Suponiendo que el algoritmo sea perfecto y no introduzca errores propios en los resultados emitidos, este es igualmente capaz de producir datos de salida incorrectos si los datos de entrada recibidos presentan errores. Dada la existencia innata de un error aleatorio por pequeño que sea, el método matemático elegido ha de presentar un cierto grado de robustez ante leves variaciones en las mediciones [73].

El método matemático que se pretende emplear es la resolución del sistema de ecuaciones planteado mediante el método de mínimos cuadrados. Este es sencillo de implementar, pero con la suficiente

robustez para soportar variaciones surgidas de errores de tipo aleatorio. Una posible descripción gráfica, aunque no la única, de este método es que este permite la proyección del vector entrada sobre el subespacio de posibles soluciones de tal forma que proyección del vector solución obtenido sea solución del sistema y por ello dicho vector sea la solución aproximada más cercana [74]. Otros posibles métodos de posicionamiento pasan por el uso de determinantes de Caley-Menger u otros métodos basados en el uso de geometría analítica. De igual forma el tratamiento estadístico de los errores obtenidos puede ayudar en su reducción [75].

A continuación, se desarrollan los principales métodos para obtener los datos de entrada necesarios para realizar el posicionamiento de un dispositivo.

2.4.2 Tiempo de llegada

El tiempo de llegada, más conocido como ToA (*Time of Arrival*) consiste en la medición del tiempo que ha empleado la señal para ir desde la estación base hasta el dispositivo móvil. Una de las formas más habituales de obtener este dato es enviar un paquete que permita realizar una sincronización de relojes entre la estación base y el dispositivo móvil para a continuación enviar un paquete con un sello temporal que indique el instante en el que se ha transmitido el mismo. Al recibirse el paquete se compara el tiempo de recepción con el dato de tiempo registrado en el sello realizando la resta se obtiene el ToA. Conocida la velocidad de propagación de una onda electromagnética, se multiplica la velocidad por el tiempo para obtener de esa forma la distancia [76]. En la Figura 2-26 se puede ver una representación gráfica del resultado final.

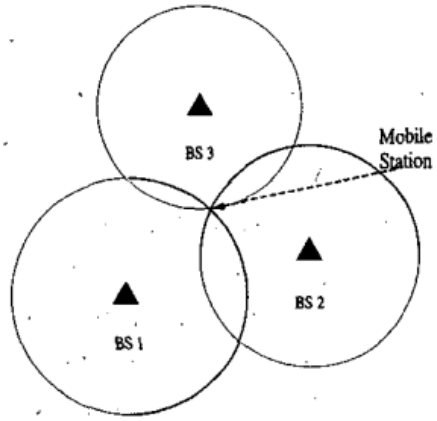

Figura 2-26: Determinación de la posición mediante ToA [77]

2.4.3 Ángulo de llegada

Conocido por sus siglas en inglés como AoA (*Angle of Arrival*) este método emplea la capacidad goniométrica de la antena para determinar la dirección de la cual proviene la señal recibida. Dado que en la información transmitida también se recibe la posición de las estaciones base, realizando el cálculo de la intersección de las señales se logra determinar la posición del dispositivo de usuario. Este método se vio beneficiado con la aparición de las antenas MIMO, las cuales al ser en realidad varias antenas, permiten obtener el AoA con mayor exactitud y menor grado de error [76]. En la Figura 2-27 se puede ver una representación gráfica del resultado final.

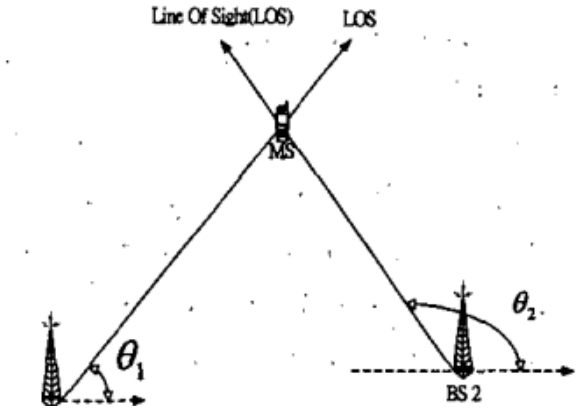

Figura 2-27: Determinación de la posición mediante AoA [77]

2.4.4 Desfase de la onda portadora

El desfase en la onda portadora o CP (*Carrier* Phase) es otro de los métodos posibles para la determinación de la distancia. Toda onda tiene asociado un cierto desfase cuando se emite. Al recibirse la señal dicho desfase varia acorde al momento en el cual llega al receptor. La señal contiene la información del desfase de salida lo que permite por comparación con el desfase obtenido determinar en qué punto del ciclo se ha recibido la señal. Contando el número de ciclos que han tenido lugar desde el emisor al receptor se puede obtener la distancia. No obstante, es necesario resolver el problema de ambigüedad que surge pues en un principio los ciclos son iguales entre sí. Existen diversos métodos, como la combinación con la obtención del ToA, o la modulación en frecuencia de la señal de forma que cada ciclo tenga una forma única que permita distinguirlo [78]. La Figura 2-28 ilustra de manera conceptual el proceso.

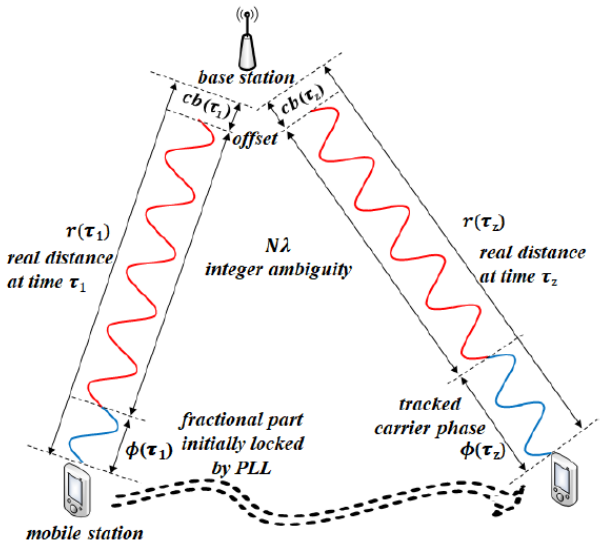

Figura 2-28: Empleo de CP para la determinación de la distancia [79]

Cabe destacar como conclusión a las tres secciones anteriores que lo habitual en la actualidad, como forma de reducir los errores y aumentar la precisión de manera notable, es la combinación de varios métodos de posicionamiento descritos. En lugar de emplear solo AoA, ToA o CP, se emplean varios de los métodos para obtener una posición media de las soluciones obtenida [79]. La Figura 2-29 es un ejemplo gráfico de este proceso.

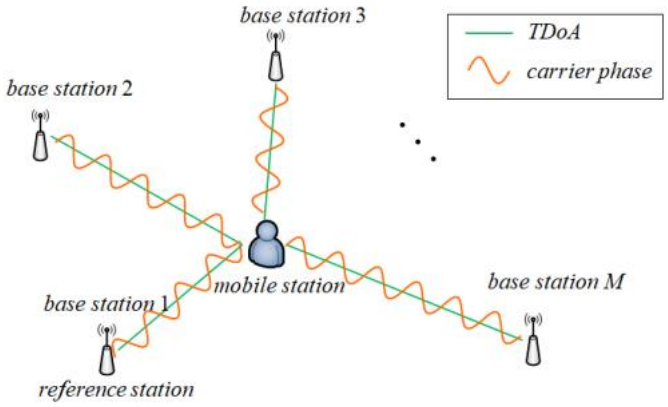

Figura 2-29: Combinación de dos métodos de posicionamiento de forma simultánea [79]

2.4.5. Ventajas del posicionamiento empleando redes 5G

Algunos aspectos clave en los que el 5G aporta mejoras significativas en los métodos de posicionamiento con respecto a otras generaciones son [80]:

- Empleo de antenas MIMO: Estas proporciona dos soluciones, siendo una utilizar *beamforming* (visto en la sección 2.1.3.3) de gran precisión para determinar el ángulo al UE o la obtención del desfase en la portadora. La implementación de múltiples frecuencias y bandas de frecuencia también contribuye a una mayor precisión en la geolocalización.
- Capacidad de computación en el borde: Apoyándose en el uso de los servidores localizados en el núcleo de la red, la capacidad de cálculo global del sistema aumenta significativamente.
- Integración con otras tecnologías: La geolocalización 5G puede integrarse como ya se ha mencionado con tecnologías como el GPS, pero además con otras como Wi-Fi y Bluetooth para mejorar la precisión especialmente en aplicaciones en la que esta resulta crítica como los vehículos autónomos o el empleo de drones en ambientes urbanos densos [76].

3. DESARROLLO DEL TRABAJO

3.1 Metodología

3.1.1 *Propósito*

Se proyecta diseñar una simulación de una burbuja 5G con las características más similares posibles a las de un caso de uso real. Se pretende que el bloque principal de esta sea lo más genérico posible abriendo la posibilidad de realizar pruebas y predicciones de otro tipo al propuesto en este trabajo.

Dentro de esta simulación se diseñará una aplicación de prueba del sistema que permita localizar dispositivos. Esto se realizará mediante el empleo de la señal emitida por tres estaciones base. Para ello se empleará el método ToA nombrado en la Sección 2.4.2 Este consiste en medir las diferencias de tiempo en las que una señal ha tardado en transmitirse entre las estaciones base y el dispositivo de usuario. Conocida la velocidad de propagación a través de la red se puede calcular la distancia de cada estación al UE (tal y como se ver en la ecuación (1)) y emplear las tres distancias en un algoritmo de trilateración basado en mínimos cuadrados que calcule la posición relativa respecto a cualquiera de las estaciones base (resolviendo el sistema de ecuaciones (2) donde x, z e y son las incógnitas del vector solución). Conocida la posición geográfica de dicha BS, bastaría con componer las coordenadas absolutas de la antena con las relativas para obtener la posición absoluta del dispositivo.

$$d_{estación\ base} = ToA \cdot v_{propagación} \quad \textbf{(1)}$$

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 = {d_{estación\ base\ 1}}^2 \\ (x - x_2)^2 + (y - y_2)^2 + (z - z_3)^2 = {d_{estación\ base\ 2}}^2 \\ (x - x_3)^2 + (y - y_3)^2 + (z - z_3) = {d_{estación\ base\ 3}}^2 \end{cases} \quad \textbf{(2)}$$

Para los cálculos de posición se considera la curvatura de la tierra como nula ya que las distancias son los suficientemente bajas para aproximar el escenario a un plano geométrico. De igual forma, se considera la señal de tres estaciones base. En un caso de uso real se emplearían 4 o más como mínimo para obtener la coordenada en altura y lograr reducir o evitar posibles errores. Se asume que todos los puntos se hallan a la misma altura geográfica. Para el objetivo propuesto de realizar una prueba de concepto, estos errores son asumibles, pero para una aplicación de uso real sí sería necesario tener en cuenta sus efectos a la precisión en la obtención del resultado.

El caso inicial plantea un escenario "vacío" en el que solamente influye la distancia entre nodos de la simulación y más tarde se plantearan escenarios en los que se simulará algún obstáculo. Estos elementos de modificación de escenarios servirán de igual forma al propósito más general de construir un "simulador raíz" que pueda servir para otros proyectos.

3.1.2 Fases del desarrollo

La Figura 3-1 recoge de manera esquemática las fases planificadas de desarrollo, describiendo de manera breve los hitos principales de cada una de ellas:

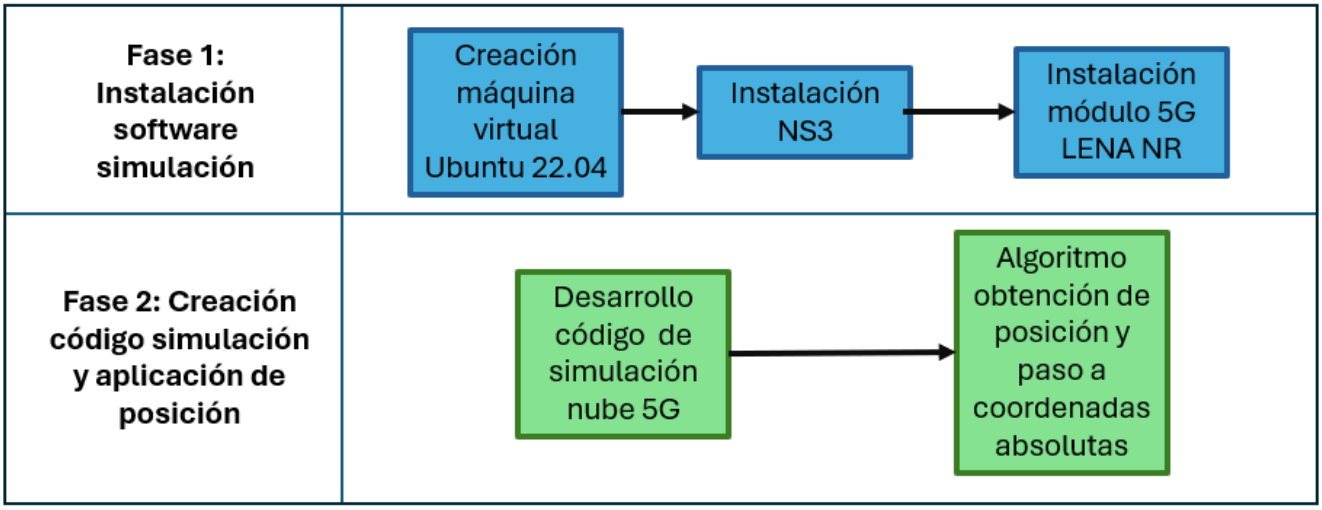

Figura 3-1: Esquema de las fases del desarrollo de la simulación. Elaboración propia

En las siguientes secciones se pasa a detallar los pasos seguidos en cada una de estas fases.

3.2 Fase 1: Instalación del software de simulación

Todo el software empleado es de uso libre y se puede acceder al mismo a través de la red. El sistema operativo de partida era *Windows 11 pro*. Dentro del mismo se instaló una máquina virtual *Linux* con el sistema operativo *Ubuntu 22.04.* A continuación, se instalaron los archivos base sobre los que trabaja NS-3 a los cuáles se añadió el módulo 5G LENA. Estos se obtuvieron de las webs de NSAM y la página de la CTTC en *GitHub*. Se podría haber utilizado cualquier otro sistema operativo como *MacOS*, haber empleado el subsistema de Linux para Windows (WSL). Para más detalles sobre otros tipos de instalaciones o sobre los requisitos para la descrita se recomienda acudir a [81].

3.2.1 Creación de la máquina virtual Ubuntu 22.04

NS-3 tiene como sistema operativo base el sistema *Linux.* Por ello es necesario tener este sistema operativo instalado en el equipo donde se vaya a instalar el software. En caso de partir de un sistema operativo Windows, existen dos posibilidades. La primera es realizar una partición del disco para instalar Linux en la misma. La segunda es emplear una máquina virtual.

Esta segunda opción suele resultar más recomendable en el caso de querer realizar un trabajo de investigación temporal como es el caso puesto que es más sencillo liberar los recursos de computadora empleados una vez acabada la vida útil de las simulaciones desarrolladas. Además, el volumen de archivos temporales generados, así como los programas de apoyo requeridos para la instalación de NS-3 hacen muy recomendable el empleo de una máquina virtual pues estas permiten la asignación dinámica

de recursos de memoria, recurso que se agotará rápidamente si no actúa con la debida prudencia y previsión. Para crear la misma se recurrió al programa *Oracle VM VirtualBox*[2] que permite tanto la creación como la administración de máquinas virtuales. De forma complementaria a este proceso, se descargó desde la página oficial de *Ubuntu* la versión 22.04 para usuarios, la cual a la fecha de la realización del trabajo era la versión estable probada más moderna.

Con el paquete de archivos ya descargado en formato archivo de imagen de disco (ISO), se procedió a crear la máquina virtual como se puede ver en la Figura 3-2. Se recomienda asignar todos los recursos de memoria y procesador posibles dentro de los niveles recomendados por *Virtual Box* a fin de que las simulaciones puedan ejecutarse de la manera más fluida posible.

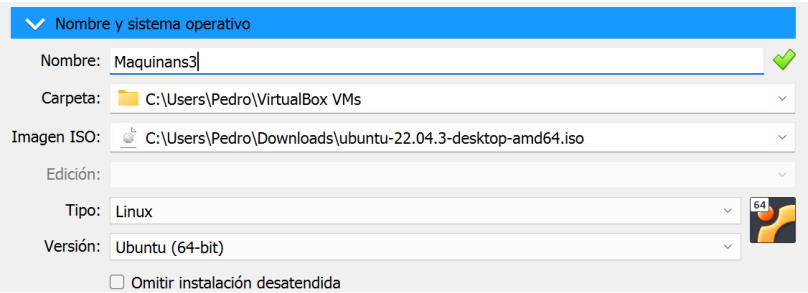

Figura 3-2: Proceso de creación máquina virtual (Autoría propia)

3.2.2 Instalación y contenidos por defecto de NS-3

Instalada la máquina virtual, se procedió a realizar la instalación de los archivos de NS-3 en concreto de la versión 3.38. Esta herramienta de simulación de redes usa de base el lenguaje de programación C++, tal y como ya se mencionó en la sección 2.3.1. Por tanto, permite el desarrollo de código en C++ e incluir numerosas bibliotecas adicionales. Estas bibliotecas constituyen el núcleo de NS-3 y contienen gran cantidad de objetos y funciones predefinidos (por ejemplo, un nodo o una antena) que facilitan la labor del desarrollador.

El otro cimiento lo constituyen todos los programas que se enumeran en el capítulo 3 de [81]. Estos colaboran tanto en funciones básicas como la compilación del código como en otras adicionales como pueden ser el seguimiento de paquetes o la visualización por pantalla de la información. La obtención de todos estos programas se puede lograr a través del empleo del comando `apt` con permisos de administrador en el terminal de *Ubuntu*. La Figura 3-3, la Figura 3-4 y la Figura 3-5 contienen cada una los programas necesarios, recomendables y opcionales respectivamente para la instalación de NS-3. Cabe destacar que tanto la guía de instalación como las versiones de los programas requeridos como prerrequisitos pueden haber variado su nombre o versión desde la fecha de publicación del trabajo.

ns-3 Version	apt Packages
3.36 and later	`g++ python3 cmake ninja-build git`
3.30-3.35	`g++ python3 git`
3.29 and earlier	`g++ python2`

Figura 3-3: Programas necesarios para la instalación de NS-3 [81]

[2] https://www.virtualbox.org/

Feature	apt Packages
Compiler cache optimization	`ccache`
Code linting	`clang-format clang-tidy`
Debugging	`gdb valgrind`

Figura 3-4: Programas recomendados para la instalación de NS-3 [81]

Feature	apt Packages
Reading pcap traces	`tcpdump wireshark`
Database support	`sqlite sqlite3 libsqlite3-dev`
NetAnim animator	`qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools`
MPI-based distributed simulation	`openmpi-bin openmpi-common openmpi-doc libopenmpi-dev`
Building Doxygen	`doxygen graphviz imagemagick`
Sphinx documentation	`python3-sphinx dia imagemagick texlive dvipng latexmk texlive-extra-utils texlive-latex-extra texlive-font-utils`
Eigen3	`libeigen3-dev`
GNU Scientific Library	`gsl-bin libgsl-dev libgslcblas0`
XML config store	`libxml2 libxml2-dev`
GTK-based config store	`libgtk-3-dev`
Emulation with virtual machines and tap bridge	`lxc-utils lxc-templates vtun uml-utilities ebtables bridge-utils`
Support for openflow	`libxml2 libxml2-dev libboost-all-dev`

Figura 3-5: Programas opcionales para la instalación [81]

Se procedió en primer lugar a descargar todos programas necesarios y opcionales para la instalación. Para ello se ha de emplear la siguiente instrucción:

```
sudo apt-get install g++
```

Donde `g++` es el programa que deseamos instalar. En algunas versiones de Linux se permite la instalación simultánea, pero se recomienda hacerlo uno a uno a fin de mantener el control sobre los programas instalados y el espacio de disco que ocupa cada uno.

Posteriormente se descargaron los archivos de instalación de NS-3. Para hacerlo existen de nuevo varias opciones, pero la elegida fue empleando el comando `wget` para acceder a la página oficial de NSAM descargándose la versión 3.38[3].

```
wget https://www.nsnam.org/releases/ns-allinone-3.41.tar.bz2
```

Dentro de los archivos descargados se distinguen dos carpetas principales, una con los programas de NS-3 y otra con los de *NetAnim*, un programa complementario que permite la visualización de la topología de red creada en una simulación.

[3] https://www.nsnam.org/releases/ns-allinone-3.41.tar.bz2

Finalizado el paso anterior se procedió a la compilación de todas las bibliotecas básicas mediante los comandos siguientes ejecutados desde el directorio de NS-3:

```
./ns3 configure –enable-test –enable-examples
./ns3 build
```

En la primera parte de la ejecución se mostrarán los módulos que se van a instalar y los que no debido a la ausencia de algún prerrequisito o algún otro error. La Figura 3-6 muestra un ejemplo por pantalla del comando `build`. Acabado el proceso anterior ya se puede comenzar a trabajar con NS-3. Para ello se ha de escribir el código con la simulación a realizar deseada, guardarlo en el directorio *scratch* contenido dentro del de NS-3 y ejecutar el comando:

```
./ns3 run nombre_del_archivo
```

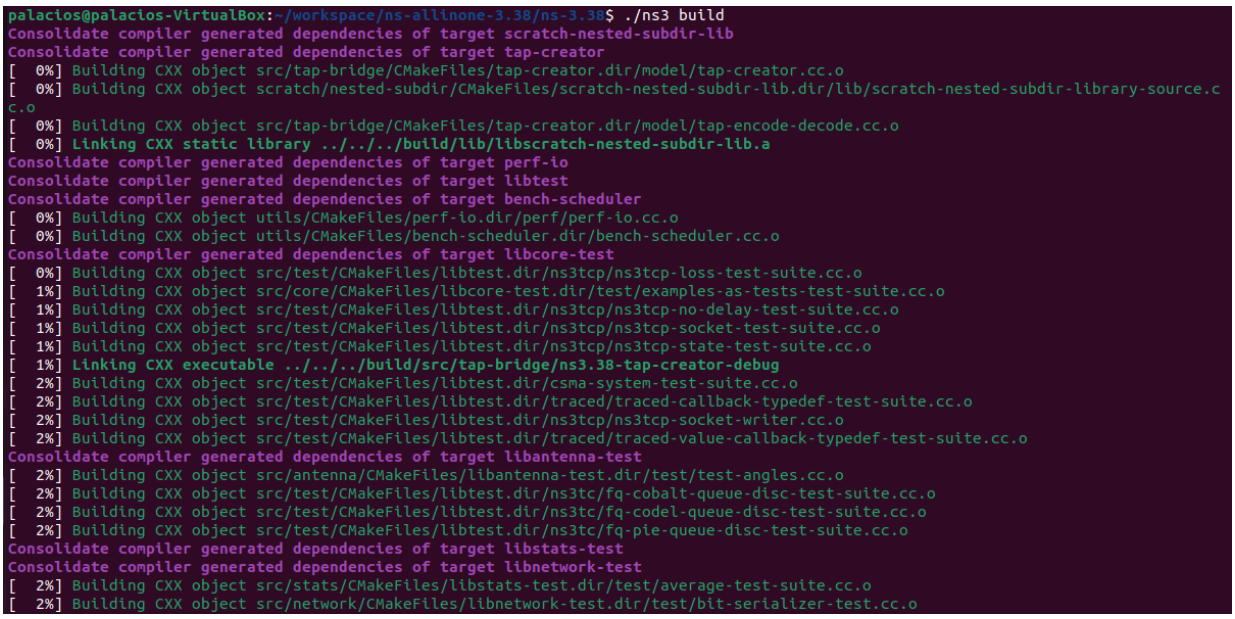

Figura 3-6: Ejecución por pantalla del comando `build`

3.2.3 Instalación del módulo 5G LENA NR

La instalación del módulo 5G LENA se puede realizar de forma paralela o secuencial con la de NS-3. En este caso se empleó el comando `git` en colaboración con la página *Gitlab* de la cual procede. La versión descargada ha de ser compatible con la versión de NS-3 descargada. Para el caso del trabajo, a la versión 3.38 le corresponde la versión 2.4. La secuencia completa de comandos es:

```
cd contrib
git clone https://gitlab.com/cttc-lena/nr.git
cd nr
git checkout -b 5g-lena-v2.4
```

Tras esto se volvió a usar el comando `build`. Es importante cerciorarse que en los módulos a instalar se muestre uno llamado `nr` (como se muestra en la Figura 3-7) que se corresponden con el software 5G LENA. En caso de no aparecer significa que la instalación no se ha podido realizar correctamente y habrá que comprobar si se tienen todos los programas requeridos como prerrequisitos.

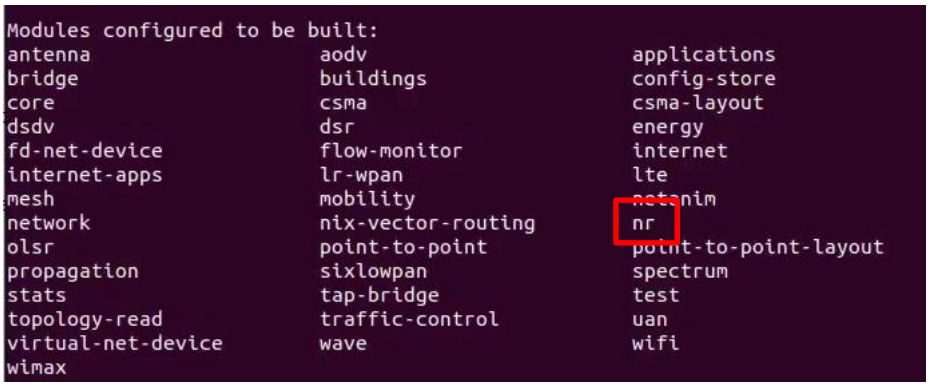

Figura 3-7: Módulos a compilar en el proceso de instalación de NS-3 con 5G LENA (Autoría propia)

3.3 Fase 2: Desarrollo del código de simulación nube 5G y aplicación de posición

Una vez instalado el software necesario, se procede a elaborar el código. Este código tiene como propósito simular una red móvil con interfaz de radio NR que sirva de base para la simulación de nubes tácticas 5G. Una red estándar de estas características debe tener al menos una estación base. Sin embargo, el caso planteado tendrá 3 estaciones base (por las razones que se expusieron en la sección 2.4) a fin de lograr el posicionamiento de un terminal de usuario. La estructura propuesta para un programa estándar es la siguiente:

1. Declaración de las bibliotecas a importar.
2. Creación del escenario, tanto del medio como de los nodos que van a participar en la simulación.
3. Configuración de las características del medio.
4. Configuración de las características de los dispositivos.
5. Creación de elementos de red externos.
6. Enrutamiento.
7. Declaración de la capa de aplicación (se decide que nodos enviarán y recibirán paquetes y en qué tiempos lo harán).
8. Establecimiento de herramientas de seguimiento y ejecución de la simulación.

De estos elementos que han de ser creados en la secuencia asignada hay algunos que de no establecerse se fijaran a unos valores genéricos por defecto como, por ejemplo, el medio de propagación de la señal. Aparte de estos bloques básicos en los que se puede clasificar el código faltaría solamente la parte de declaración de las bibliotecas empleadas. NS-3 tiene en su página web abundantes recursos sobre el contenido de cada una de sus bibliotecas. De igual forma, el CTTC tiene también recursos en línea sobre sus bibliotecas adicionales que constituyen el módulo 5G LENA. Ambas fuentes de información son accesibles a través de las siguientes referencias [82, 83].

Con el fin de facilitar la explicación el código se ha segmentado en distintas figuras. El código completo se incluye en el Anexo III: .

3.3.1 Declaración de bibliotecas

En la Figura 3-8 se pueden ver las bibliotecas que se importaron para la simulación desarrollada. Destacar que en [82] se pueden encontrar más detalles sobre el contenido de las bibliotecas correspondiente a NS-3. También se incluyeron bibliotecas básicas de C++ distinguibles por no incorporar la cabecera "`ns3/`".

```
#include "ns3/applications-module.h"
#include "ns3/config-store.h"
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/log.h"
#include "ns3/mobility-module.h"
#include "ns3/network-module.h"
#include "ns3/nr-helper.h"
#include "ns3/nr-mac-scheduler-tdma-rr.h"
#include "ns3/nr-module.h"
#include "ns3/nr-point-to-point-epc-helper.h"
#include "ns3/point-to-point-helper.h"
#include <ns3/antenna-module.h>
#include <ns3/buildings-helper.h>
#include <cmath>
#include "math.h"
#include "ns3/netanim-module.h"
#include "ns3/flow-monitor.h"
#include "ns3/flow-monitor-helper.h"
#include "ns3/node-list.h"
#include "ns3/ipv4-flow-classifier.h"
#include "iostream"
```

Figura 3-8: Declaración de bibliotecas (Autoría propia)

3.3.2 Creación del escenario

En la Figura 3-9 se puede ver el comienzo de la función principal y la declaración de las variables fundamentales para la simulación. El código incorpora comentarios en verde oscuro cuyo propósito es inhabilitar un comando o esclarecer el comando en cuestión. En la línea 117 se configura el escenario entre los posibles:

- rural "`Rma`",
- urbano "`Uma`",
- urbano con gran número de edificaciones "`Umi StreetCanyon`",
- interior mixto "`InH-OfficeMixed`" e
- interior abierto "`InH-OfficeOpen`.

En la sección 2.1.3.3 ya se describieron los tres escenarios que se contemplan dentro del 5G. En este caso el simulador añade los dos posibles escenarios más que en realidad se tratan de versiones del escenario urbano de alta densidad, pero adaptados para su uso en espacios interiores. En las líneas siguientes se declaran la frecuencia, ancho de banda, movilidad de los nodos, tiempo de simulación, velocidad de un UE en movimiento, activador de los servicios de análisis por defecto, alturas de las antenas y potencia de transmisión. Por último, se declaran tres variables que servirán para medir los tiempos de transmisión de cara a la localización del UE y dos variables tipo cadena (`string`) que servirán para la presentación por pantalla de la información.

```
90    //Bloque de configuración inicial del escenario
91    std::string scenario = "UMa"; // escenario general de la simulación a elegir entre
92    |   |   |   |   |   |   |   | //'RMa', 'UMa', 'UMi-StreetCanyon', 'InH-OfficeMixed' y 'InH-OfficeOpen'
93    double frequency = 28e9;      // frecuencia central
94    double bandwidth = 100e6;     // ancho de banda
95    double mobility = false;      // mobilidad habilitada si/no
96    double simTime = 1.5;           // tiempo de simulación en segundos. Este ha de tener la sufieciente duración para
97    |   |   |   |   |   |   |   |   // albergar todos los eventos que queramos simular
98    double speed = 1;             // velocidad en m/s para un UT caminando
99    bool logging = true; // habilitación de los servicios de análisis de
100   |   |   |   |   |   | // resultados básicos
101   double hBS;          // altura de la estación base en metros
102   double hUT;          // altura de la antena de usuario en metros
103   double txPower = 50; // Potencia de tx
104   double t1,t2,t3;
105   std::string simTag = "default";
106   std::string outputDir = "./";
```

Figura 3-9: Declaración de variables básicas (Autoría propia)

En la Figura 3-10 se puede ver la ejecución de dos sentencias que contienen dos funciones importantes para el correcto desarrollo de la simulación. La primera de ellas configura el modelo de canal y el modelo de propagación acorde al escenario elegido. Esto determina la manera en que las ondas se propagaran de forma simulada. De igual forma otro de los comandos que se ejecuta internamente al llamar a esta función es la inicialización de una semilla de números pseudoaleatorios. Esta semilla permite a NS-3 añadir una pequeña variación aleatoria a los cálculos realizados para crear mayor realismo en la simulación. La segunda sentencia permite la compatibilidad con protocolos y dispositivos de otras generaciones a nivel interno. Es importante incluirla ya que, aunque la mayoría de los nodos se configuraran usando funciones provenientes de 5G-LENA, si se incluye un elemento que utilice funciones estándar de NS-3 puede que no sea reconocido como parte de la red 5G.

```
110     //Configurado de las opciones seleccionadas en la simulación
111     enum BandwidthPartInfo::Scenario scenarioEnum = BandwidthPartInfo::UMa;
112     Config::SetDefault("ns3::LteRlcUm::MaxTxBufferSize", UintegerValue(999999999));
```

Figura 3-10: Configuración del modelo de canal y de la retrocompatibilidad con sistemas anteriores al 5G

```
120     // configuración de la altura de las antenas acorde al escenario elegido
121     if (scenario == "RMa")
122     {
123         hBS = 40;
124         hUT = 1.5;
125         scenarioEnum = BandwidthPartInfo::RMa;
126     }
127     else if (scenario == "UMa")
128     {
129         hBS = 25;
130         hUT = 1.5;
131         scenarioEnum = BandwidthPartInfo::UMa;
132     }
133     else if (scenario == "UMi-StreetCanyon")
134     {
135         hBS = 10;
136         hUT = 1.5;
137         scenarioEnum = BandwidthPartInfo::UMi_StreetCanyon;
138     }
139     else if (scenario == "InH-OfficeMixed")
140     {
141         hBS = 3;
142         hUT = 1;
143         scenarioEnum = BandwidthPartInfo::InH_OfficeMixed;
144     }
145     else if (scenario == "InH-OfficeOpen")
146     {
147         hBS = 3;
148         hUT = 1;
149         scenarioEnum = BandwidthPartInfo::InH_OfficeOpen;
150     }
151     else
152     {
153         NS_ABORT_MSG("Scenario not supported. Choose among 'RMa', 'UMa', 'UMi-StreetCanyon', "
154                      "'InH-OfficeMixed', and 'InH-OfficeOpen'.");
155     }
156
```

Figura 3-11: Configuración de las alturas de las antenas acorde al escenario seleccionado (Autoría propia)

Acorde al tipo de escenario se eligen las alturas físicas de las antenas acorde al escenario elegido como se ve en la Figura 3-11. Estas alturas vienen definidas por defecto en la documentación del módulo 5G LENA [83] y podrían ser variadas acorde al modelo de antena real que se esté tratando de simular. En caso de que se haya empleado una sintaxis incorrecta para elegir el escenario se parará la simulación y se emitirá un mensaje de error por pantalla.

El siguiente segmento de código realiza la creación de tres nodos correspondientes a las antenas (ENB) un terminal móvil (UE). Cada conjunto de nodos se agrupa en un contenedor que facilitará el

acceso a los distintos elementos de cara configurar los parámetros de estos. Además, estos nodos han de ser posicionados, lo cual se hace creando el vector `enBpositionAlloc` que recoge las posiciones de las estaciones base. A continuación, se crean los objetos `enbmobility` y `uemobility` que recogerán el conjunto de las posiciones y en el caso de los UE's el movimiento si es que lo tiene. La configuración de los terminales de usuario viene reflejada en la Figura 3-12 y Figura 3-13.

```
157     // Creación de las estaciones base y de los terminales de usuario
158     NodeContainer enbNodes;
159     NodeContainer ueNodes;
160     enbNodes.Create(3);
161     ueNodes.Create(1);
162
163     // posición de las estaciones base
164
165
166     ns3::Vector senB1= {0.0, 0.0, hBS};
167     ns3::Vector senB2= {0.0, 500.0, hBS};
168     ns3::Vector senB3= {500.0,0.0, hBS};
169
170     Ptr<ListPositionAllocator> enbPositionAlloc = CreateObject<ListPositionAllocator>();
171     enbPositionAlloc->Add(senB1);
172     enbPositionAlloc->Add(senB2);
173     enbPositionAlloc->Add(senB3);
174     MobilityHelper enbmobility;
175     enbmobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
176     enbmobility.SetPositionAllocator(enbPositionAlloc);
177     enbmobility.Install(enbNodes);
178
179     // posición de los terminales de usuario y habilitación de la movilidad
180     MobilityHelper uemobility;
181     uemobility.SetMobilityModel("ns3::ConstantVelocityMobilityModel");
182     uemobility.Install(ueNodes);
```

Figura 3-12: Creación y configuración de las posiciones de las estaciones base (Autoría propia)

```
184     if (mobility)
185     {
186         ueNodes.Get(0)->GetObject<MobilityModel>()->SetPosition({0, 15, hUT}); // (x, y, z) en m
187         ueNodes.Get(0)->GetObject<ConstantVelocityMobilityModel>()->SetVelocity(
188             {speed, 0, 0}); // En este caso el dispositivo se movera a lo largo de x con velocidad constante
189
190
191     }
192     else
193     {
194         ueNodes.Get(0)->GetObject<MobilityModel>()->SetPosition({20, 200, hUT});
195         ueNodes.Get(0)->GetObject<ConstantVelocityMobilityModel>()->SetVelocity({0, 0, 0});
196
197     }
```

Figura 3-13: Caracterización de la posición y movimiento de los UE's (Autoría propia)

3.3.3 Configuración de las características del medio

La Figura 3-14 muestra la configuración de las características de la interfaz de radio. Para ello se hace uso de distintos objetos denominados como `Helpers` los cuales son objetos predefinidos que contienen los parámetros de interés por defecto. La forma de ajustarlos será modificar las distintas clases que constituyen dichos objetos. De las líneas 208 a la 222 se realiza la configuración de la señal portadora y de las bandas operativas. En el caso de la simulación solo se emplea una banda de trabajo (definida por un espectro de frecuencias) dentro de la cual se utiliza una única onda portadora. En caso de querer simular varios flujos de información independientes se tendrían que definir varias bandas de trabajo cada una con un intervalo de frecuencias asociado y con las portadoras deseadas, pero para el desarrollo de este trabajo solo es necesaria una. A continuación, se elige el conformado de haz a emplear (línea 224), así como el mecanismo de *scheduling* (línea 229) a emplear para la transmisión de señales. De nuevo,

este parámetro está más pensado para la gestión de múltiples flujos de información a transmitir en lugar de uno solo.

```
200     /* Creación de los simulation helpers de NR*/
201     Ptr<NrPointToPointEpcHelper> epcHelper = CreateObject<NrPointToPointEpcHelper>();
202     Ptr<IdealBeamformingHelper> idealBeamformingHelper = CreateObject<IdealBeamformingHelper>();
203     Ptr<NrHelper> nrHelper = CreateObject<NrHelper>();
204     nrHelper->SetBeamformingHelper(idealBeamformingHelper);
205     nrHelper->SetEpcHelper(epcHelper);
206
207     /*
208      * Configuración de espectro. Se crea una operational band única y se configura el escenario.
209      */
210   BandwidthPartInfoPtrVector allBwps;
211     CcBwpCreator ccBwpCreator;
212     const uint8_t numCcPerBand = 1; // en este caso se tiene una única banda, y esa banda
213                                     // esta compuesta de una única portadora
214
215     CcBwpCreator::SimpleOperationBandConf bandConf(frequency,
216                                                    bandwidth,
217                                                    numCcPerBand,
218                                                    scenarioEnum);
219     OperationBandInfo band = ccBwpCreator.CreateOperationBandContiguousCc(bandConf);
220     // Inicializa para la banda los módelos de canal y de propagación elegidos para la simulación
221     nrHelper->InitializeOperationBand(&band);
222     allBwps = CcBwpCreator::GetAllBwps({band});
223
224     // Configuración del método ideal de beamforming
225     idealBeamformingHelper->SetAttribute("BeamformingMethod",
226                                          TypeIdValue(DirectPathBeamforming::GetTypeId()));
227
228     // Configuración del scheduler
229     nrHelper->SetSchedulerTypeId(NrMacSchedulerTdmaRR::GetTypeId());
```

Figura 3-14: Configuración *beamforming* y de la portadora acorde a los parámetros ya seleccionados (Autoría propia)

3.3.4 Configuración de las características de los dispositivos

A través de los `Helpers` se configuran también los parámetros de la interfaz radio de los dispositivos UE y los ENB's. Atendiendo a la Figura 3-15 los comandos de las líneas 232 a 253 configuran las antenas como tal. Se observa que se configuran dos parámetros `NumRows` y `NumColumns` que ajustan el número de celdas que compondrán la antena MIMO empleada. A continuación, se combinan en un nuevo objeto, los `NetDeviceContainer`, los nodos de la red móvil con los parámetros que se han fijado. El segmento termina con la asignación de una variabilidad aleatoria al sistema que lo hagan más real (líneas 311 y 312) así como con la inclusión de la potencia de transmisión en el objeto de las antenas. Esta variabilidad aleatoria se sirve de la semilla de número pseudoaleatorios que se mencionó en la sección 3.3.2. La sección tiene una última serie de instrucciones que terminan de actualizar la configuración de todos los nodos como se muestra en la Figura 3-16.

```
231     // Configuración de las antenas de los UE's
232     nrHelper->SetUeAntennaAttribute("NumRows", UintegerValue(2));
233     nrHelper->SetUeAntennaAttribute("NumColumns", UintegerValue(4));
234     nrHelper->SetUeAntennaAttribute("AntennaElement",
235                                     PointerValue(CreateObject<IsotropicAntennaModel>()));
236
237     // Configuración de las antenas para los gNBs
238     nrHelper->SetGnbAntennaAttribute("NumRows", UintegerValue(8));
239     nrHelper->SetGnbAntennaAttribute("NumColumns", UintegerValue(8));
240     nrHelper->SetGnbAntennaAttribute("AntennaElement",
241                                      PointerValue(CreateObject<IsotropicAntennaModel>()));
242
243     // instalación de los dispositivos de red NR
244     NetDeviceContainer enbNetDev = nrHelper->InstallGnbDevice(enbNodes, allBwps);
245     NetDeviceContainer ueNetDev = nrHelper->InstallUeDevice(ueNodes, allBwps);
246
247     int64_t randomStream = 1;
248     randomStream += nrHelper->AssignStreams(enbNetDev, randomStream);
249     randomStream += nrHelper->AssignStreams(ueNetDev, randomStream);
250
251     nrHelper->GetGnbPhy(enbNetDev.Get(0), 0)->SetTxPower(txPower);
252     nrHelper->GetGnbPhy(enbNetDev.Get(1), 0)->SetTxPower(txPower);
253     nrHelper->GetGnbPhy(enbNetDev.Get(2), 0)->SetTxPower(txPower);
```

Figura 3-15: Elección de los parámetros característicos concernientes a la interfaz radio de los dispositivos (Autoría propia)

```
322     // Después de realizar toda la configuración, se ha de llamar explicitamente a la función UpdateConfig ()
323     for (auto it = enbNetDev.Begin(); it != enbNetDev.End(); ++it)
324     {
325         DynamicCast<NrGnbNetDevice>(*it)->UpdateConfig();
326     }
327
328     for (auto it = ueNetDev.Begin(); it != ueNetDev.End(); ++it)
329     {
330         DynamicCast<NrUeNetDevice>(*it)->UpdateConfig();
331     }
```

Figura 3-16: Actualización de los nodos de la red móvil (Autoría propia)

3.3.5 Creación de los elementos de red externos

A fin de dotar de un núcleo a la red móvil simulada, se crean dos nodos más en la simulación, el `pgw` y el `remoteHost`. El primero es la abreviatura de *principal Gateway*. Ambos constituyen los elementos que simulan un núcleo que es capaz de recibir y enviar paquetes de datos. Las líneas 299 a la 311 de la Figura 3-17 se corresponden con la creación de este núcleo. Prosigue la sección con la declaración de los parámetros que regirán las conexiones entre estos nodos, tanto entre sí como con el resto de la red.

```
297     // Creación de la conexión al núcleo e instalación de los protocolos IP en los UEs
298     // obtención SGW/PGW y creación de un único RemoteHost
299     Ptr<Node> pgw = epcHelper->GetPgwNode();
300     NodeContainer remoteHostContainer;
301     remoteHostContainer.Create(1);
302     Ptr<Node> remoteHost = remoteHostContainer.Get(0);
303     InternetStackHelper internet;
304     internet.Install(remoteHostContainer);
305
306     // Conexión remoteHost a pgw. Configuración del enrutamiento
307     PointToPointHelper p2ph;
308     p2ph.SetDeviceAttribute("DataRate", DataRateValue(DataRate("100Gb/s")));
309     p2ph.SetDeviceAttribute("Mtu", UintegerValue(2500));
310     p2ph.SetChannelAttribute("Delay", TimeValue(Seconds(0.000)));
311     NetDeviceContainer internetDevices = p2ph.Install(pgw, remoteHost);
```

Figura 3-17: Creación de los elementos del núcleo de la red (Autoría propia).

3.3.6 Enrutamiento y creación de la capa de aplicación.

Como en cualquier topología de red, es necesario establecer las rutas por las que va a circular la información. Es por ello por lo que tal y como se puede observar en la Figura 3-18, se fija una base de direcciones IP asignables las cuales son distribuidas mediante distintos `Helpers`. La línea 319 contiene el comando que asigna direcciones a los elementos del núcleo y la línea 324 tiene la sentencia que hace lo mismo con los UE's.

```
313     Ipv4AddressHelper ipv4h;
314     ipv4h.SetBase("1.0.0.0", "255.0.0.0");
315     Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign(internetDevices);
316     Ipv4StaticRoutingHelper ipv4RoutingHelper;
317
318     Ptr<Ipv4StaticRouting> remoteHostStaticRouting =
319     ipv4RoutingHelper.GetStaticRouting(remoteHost->GetObject<Ipv4>());
320     remoteHostStaticRouting->AddNetworkRouteTo(Ipv4Address("7.0.0.0"), Ipv4Mask("255.0.0.0"), 1);
321     internet.Install(ueNodes);
322
323     Ipv4InterfaceContainer ueIpIface;
324     ueIpIface = epcHelper->AssignUeIpv4Address(NetDeviceContainer(ueNetDev));
```

Figura 3-18: Creación de base de IP's y asignación a los nodos (Autoría propia)

```
327     // asignación de las direcciones IP a las UEs y instalaciobn de las aplicaciones de UDP downlink
328     uint16_t dlPort = 1234;
329     ApplicationContainer clientApps;
330     ApplicationContainer serverApps;
331     for (uint32_t u = 0; u < ueNodes.GetN(); ++u)
332     {
333         Ptr<Node> ueNode = ueNodes.Get(u);
334         // Configuración de la gateway oor defecto para el UE
335         Ptr<Ipv4StaticRouting> ueStaticRouting =
336         ipv4RoutingHelper.GetStaticRouting(ueNode->GetObject<Ipv4>());
337         ueStaticRouting->SetDefaultRoute(epcHelper->GetUeDefaultGatewayAddress(), 1);
338
339         UdpServerHelper dlPacketSinkHelper(dlPort);
340         serverApps.Add(dlPacketSinkHelper.Install(ueNodes.Get(u)));
341
342         UdpClientHelper dlClient(ueIpIface.GetAddress(u), dlPort);
343         dlClient.SetAttribute("Interval", TimeValue(MicroSeconds(1)));
344         // dlClient.SetAttribute ("MaxPackets", UintegerValue(0xFFFFFFFF));
345         dlClient.SetAttribute("MaxPackets", UintegerValue(1));
346         dlClient.SetAttribute("PacketSize", UintegerValue(1500));
347         clientApps.Add(dlClient.Install(remoteHost));
348
349     nrHelper->AttachToEnb(ueNetDev.Get(0), enbNetDev.Get(cont));
```

Figura 3-19: Asignación direcciones IP y declaración aplicaciones de usuario (Autoría propia)

Tras la creación de las direcciones IP comienza el proceso de enrutamiento, el cual se simultanea con la creación de la capa de aplicación, la cual implica indicar que nodos enviarán información y cuales la recibirán. Para el caso, el núcleo de la red enviará los paquetes a los UE's. El proceso está generalizado para todos los dispositivos que existan, aunque para el caso de estudio solo sea uno. La Figura 3-19 recoge en primer lugar la creación del puerto de transmisión (línea 328) y la creación de las aplicaciones (líneas 329 y 330) para luego proseguir con el enrutamiento (líneas 336 y 337), la asignación de la aplicación de servidor (líneas 339 y 340) que será la receptora de paquetes; y de las líneas 342 en adelante se desarrolla la capa de aplicación cliente que enviará los paquetes en este caso uno único de un tamaño de 1500 bytes. Estos paquetes tienen el valor de 1500 por defecto a fin de simular una capacidad suficiente para la información de los diversos protocolos de red (en caso de no alcanzarse el tamaño mínimo acorde a los protocolos participantes, se emitirá un aviso). Esta aplicación se instala en el núcleo. La última línea mostrada realiza la asignación entre dispositivo y estación base. Para el caso se fuerza a conectarse a una determinada estación a fin de hallar la distancia a la misma, pero por defecto un dispositivo se enlazará con el ENB más cercano.

Respecto al tipo de protocolo de la capa de transporte a emplear, entre las dos opciones habituales, UDP (*User Datagram Protocol*) y TCP (*Transmission Control Protocol*) se suele escoger la primera al ser más simple de implementar, no generar flujos de control auxiliares a la transmisión principal y al estar en el contexto de una red inalámbrica donde el número de errores es mayor provocando que TCP sea menos eficiente.

```
351    // Comienzo de las aplicaciones de servidor y usuario
352    serverApps.Start(Seconds(0.2));
353    clientApps.Start(Seconds(0.2));
354    serverApps.Stop(Seconds(simTime));
355    clientApps.Stop(Seconds(simTime - 0.2));
```

Figura 3-20: Tiempos de comienzo de las distintas aplicaciones dentro de la simulación (Autoría propia)

El último paso para terminar con el funcionamiento de las aplicaciones es fijar los tiempos en los que cada aplicación comenzarán y terminarán de funcionar (Figura 3-20). Es recomendable evitar errores de tipo lógico como elegir un tiempo de comienzo superior al de duración de la simulación ya que es probable que, aunque el código se ejecute con aparente normalidad en realidad existan errores subyacentes en el mismo.

3.3.7 Creación de las herramientas de seguimiento y ejecución de la simulación

Los últimos pasos en la creación de la simulación vienen reflejados en la Figura 3-21. El primer segmento habilita las trazas propias del módulo `nr` para el seguimiento de los flujos de tráfico. De las líneas 362 a 365 se declara el objeto `flowmonHelper` y un nuevo contenedor para almacenar los nodos que serán los sujetos del seguimiento. En la línea 367 se declara el puntero de seguimiento y a continuación los parámetros de almacenamiento con los que trabajará. Para concluir, las tres últimas instrucciones ejecutan la simulación. La primera fija el tiempo de parada, la segunda pone en marcha el proceso hasta alcanzar el tiempo indicado y la tercera borra todos los objetos, punteros y variables involucrados en la simulación de forma que quede perfectamente libre para futuros usos.

```
357    // habilitación de las trazas de seguimiento propias del módulo NR
358    nrHelper->EnableTraces();
359
360    //Config::ConnectWithoutContext ("/UeNetDevice/ueNodes/ueNode(0)/", MakeCallback (&PacketTrace));
361
362    FlowMonitorHelper flowmonHelper;
363    NodeContainer endpointNodes;
364    endpointNodes.Add(remoteHost);
365    endpointNodes.Add(ueNodes);
366
367    Ptr<ns3::FlowMonitor> monitor = flowmonHelper.Install(endpointNodes);
368    monitor->SetAttribute("DelayBinWidth", DoubleValue(0.001));
369    monitor->SetAttribute("JitterBinWidth", DoubleValue(0.001));
370    monitor->SetAttribute("PacketSizeBinWidth", DoubleValue(20));
371
372    Simulator::Stop(Seconds(simTime));
373    Simulator::Run();
```

Figura 3-21: Declaración elementos de seguimiento y ejecución de la simulación (Autoría propia)

3.3.8 Obtención de datos del flujo

A la hora de realizar el seguimiento de paquetes es necesario ejecutar los comandos necesarios para ello después de ejecutar la función `simulator::run`. La Figura 3-22 muestra desde las líneas 367 a la 370 como obtener los datos del flujo mediante el uso de comandos específicos de NS-3. Estos datos se extraen del puntero `monitor` y se almacenan en un contenedor específico. Las líneas 375 y 376 crean dos variables que se usaran para mostrar la información por pantalla. La parte inferior de la figura describe los pasos para abrir un fichero externo donde almacenar la información de cara a querer procesar dichos datos en una aplicación externa.

```
367     monitor->CheckForLostPackets();
368     Ptr<Ipv4FlowClassifier> classifier =
369         DynamicCast<Ipv4FlowClassifier>(flowmonHelper.GetClassifier());
370     FlowMonitor::FlowStatsContainer stats = monitor->GetFlowStats();
371
372     double averageFlowThroughput = 0.0;
373     double averageFlowDelay = 0.0;
374
375     std::ofstream outFile;
376     std::string filename = outputDir + "/" + simTag;
377     outFile.open(filename.c_str(), std::ofstream::out | std::ofstream::trunc);
378     if (!outFile.is_open())
379     {
380         std::cerr << "Can't open file " << filename << std::endl;
381         return 1;
382     }
383
384     outFile.setf(std::ios_base::fixed);
```

Figura 3-22: Extracción de datos del puntero a flujos y apertura de fichero para escribir datos (Autoría propia)

En la Figura 3-22 y la Figura 3-23 se termina de exponer el proceso. Se emplea un bucle iterativo `for` para recorrer los elementos del contenedor. En primer lugar, de las líneas 400 a 410 se determina el tipo de protocolo de capa de transporte empleado. De las líneas 411 a 418 se formatea el flujo de información que se va a presentar por pantalla. La variable `i` actúa de almacén temporal de cada una de las variables de interés y por eso aparece múltiples veces. Si se cumple la condición de la línea 419, se procederá a ejecutar los comandos siguientes que presentarán por pantalla toda la información del flujo.

```
395     double flowDuration = (simTime - 0.2);
396     for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator i = stats.begin();
397         i != stats.end();
398         ++i)
399     {
400         Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow(i->first);
401         std::stringstream protoStream;
402         protoStream << (uint16_t)t.protocol;
403         if (t.protocol == 6)
404         {
405             protoStream.str("TCP");
406         }
407         if (t.protocol == 17)
408         {
409             protoStream.str("UDP");
410         }
411         outFile << "Flow " << i->first << " (" << t.sourceAddress << ":" << t.sourcePort << " -> "
412             << t.destinationAddress << ":" << t.destinationPort << ") proto "
413             << protoStream.str() << "\n";
414         outFile << "  Tx Packets: " << i->second.txPackets << "\n";
415         outFile << "  Tx Bytes:   " << i->second.txBytes << "\n";
416         outFile << "  TxOffered:  " << i->second.txBytes * flowDuration / 1000.0;
417             << " Mbps\n";
418         outFile << "  Rx Bytes:   " << i->second.rxBytes << "\n";
```

Figura 3-23: Extracción e impresión por pantalla de la información de interés (Autoría propia)

```
419         if (i->second.rxPackets > 0)
420         {
421             // Measure the duration of the flow from receiver's perspective
422             averageFlowThroughput += i->second.rxBytes * 8.0 / flowDuration / 1000 / 1000;
423             averageFlowDelay += 1000 * i->second.delaySum.GetSeconds() / i->second.rxPackets;
424 
425             outFile << "  Throughput: " << i->second.rxBytes * 8.0 / flowDuration / 1000 / 1000
426                     << " Mbps\n";
427             outFile << "  Mean delay:  "
428                     << i->second.delaySum.GetNanoSeconds() << " ms\n";
429                     if(cont==0){
430                         t1=i->second.delaySum.GetSeconds() / i->second.rxPackets;
431                     }
432                     if(cont==1){
433                         t2= i->second.delaySum.GetSeconds() / i->second.rxPackets;
434                     }
435                     if(cont==2){
436                         t3= i->second.delaySum.GetSeconds() / i->second.rxPackets;
437                     }
438             outFile << "  Mean jitter:  "
439                     <<i->second.jitterSum.GetNanoSeconds()<< " ms\n";
440         }
441         else
442         {
443             outFile << "  Throughput:  0 Mbps\n";
444             outFile << "  Mean delay:  0 ms\n";
445             outFile << "  Mean jitter: 0 ms\n";
446         }
447         outFile << "  Rx Packets: " << i->second.rxPackets << "\n";
448     }
449 
450     outFile << "\n\n  Mean flow throughput: " << averageFlowThroughput / stats.size() << "\n";
451     outFile << "  Mean flow delay: " << averageFlowDelay / stats.size() << "\n";
452 
453     outFile.close();
```

Figura 3-24: Finalización del proceso de extracción de datos e impresión por pantalla (Autoría propia)

Nótese como se puede emplear esta sección para extraer algún dato de interés como es el tiempo de transmisión (dato que servirá para el cálculo de la posición) y almacenarlo en una variable tal y como se hace en las líneas de la 425 a la 439. Finalizado el proceso de presentación de la información se procede a cerrar el fichero en la línea 453. Posteriormente a estas instrucciones habrá que proceder a finalizar la simulación y la ejecución con los comandos "`Simulator::Destroy();`" y "`return 0;`"

3.3.9 Algoritmo de trilateración

En el presente apartado se retomará la teoría ya expuesta en la sección 2.4 para desarrollar el algoritmo de trilateración a emplear. Tal y como se expuso en dicha sección, existen múltiples técnicas para posicionar un terminal de usuario e incluso dentro de las técnicas de trilateración, múltiples algoritmos basados en distintos procedimientos matemáticos. El algoritmo empleado es la resolución de un sistema de ecuaciones (observable en la ecuación **(2)**) por el método de mínimos cuadrados siendo este método de tipo algebraico. En concreto se emplea la variante SVP (*Singular Value Decomposition* o Descomposición en valores singulares) sobre la cual se puede encontrar más información en la referencia [84]. Otros algoritmos reportaran resultados más precisos, pero para realizar una prueba de concepto se considera el empleado lo suficientemente robusto. La Figura 3-25 resume los pasos a realizar.

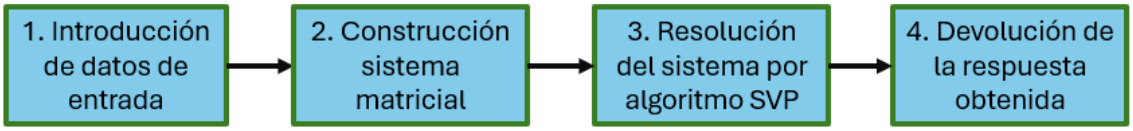

Figura 3-25: Diagrama de flujo del funcionamiento del algoritmo (Autoría propia)

```
44  struct Point {
45      double x, y, z;
46  };
47
48  Point trilaterate(const std::vector<Point>& positions, const std::vector<double>& distances) {
49      if (positions.size() < 3 || positions.size() != distances.size()) {
50          std::cerr << "Error: Número incorrecto de puntos o distancias." << std::endl;
51          return {0, 0, 0};
52      }
53
54      int n = positions.size(); // Número de puntos de referencia
55      MatrixXd A(n - 1, 3);
56      VectorXd b(n - 1);
57
58      // Construye la matriz de ecuaciones lineales
59      for (int i = 1; i < n; ++i) {
60          A.row(i - 1) << 2 * (positions[i].x - positions[0].x),
61                          2 * (positions[i].y - positions[0].y),
62                          2 * (positions[i].z - positions[0].z);
63          b(i - 1) = std::pow(distances[0], 2) - std::pow(distances[i], 2) +
64                     std::pow(positions[i].x, 2) - std::pow(positions[0].x, 2) +
65                     std::pow(positions[i].y, 2) - std::pow(positions[0].y, 2) +
66                     std::pow(positions[i].z, 2) - std::pow(positions[0].z, 2);
67      }
68
69      // Resuelve el sistema de ecuaciones lineales utilizando mínimos cuadrados
70      Vector3d solution = A.jacobiSvd(ComputeThinU | ComputeThinV).solve(b);
71
72      // Calcula las coordenadas del dispositivo
73      Point devicePosition;
74      devicePosition.x = solution(0);
75      devicePosition.y = solution(1);
76      devicePosition.z = solution(2);
77
78      return devicePosition;
79  }
```

Figura 3-26: Función que desarrolla y ejecuta el algoritmo de trilateración (Autoría propia)

La Figura 3-26 muestra el proceso de implementación. De las líneas 48 a 52 se implementa el paso 1. De la 54 a la 68 se realiza el paso 2. Por una parte, tenemos la matriz que se construye tomando las distancias desde la posición de una de las estaciones base que se establece como punto de referencia. Cada fila contendrá las distancias descompuestas por coordenadas. El vector se compone de las diferencias de distancias entre la distancia desde la estación de referencia y las distancias desde las otras dos estaciones base. En la línea 70 se llama a la función que resuelve el sistema completando el paso 3. Por último, de las líneas 73 a 77 se devuelve el resultado obtenido.

Con estos resultados previos, se procede a obtener las coordenadas individuales del punto de interés. Finalmente, las coordenadas de interés son almacenadas en el objeto creado para ello y la función las devuelve como resultado. Durante el transcurso de la simulación y como último paso antes de concluirla se llamará a esta función proporcionando lo datos de posiciones de las estaciones base y de distancias. La forma de obtener las distancias es multiplicar los tiempos de transmisión por la velocidad de propagación a través de todo el canal.

3.3.10 Implementación de edificios

Como pieza adicional del código de cara a la realización de una de las pruebas, se explican a continuación las líneas necesarias para crear edificaciones virtuales en la simulación de NS-3. Este viene desarrollado en la Figura 3-27. Es importante como paso inicial incluir las bibliotecas correspondientes con el módulo `buildings`. De la línea 273 a la 274 se establecen los puntos que conformaran los vértices del edificio los cuáles son unidos entre sí por segmentos rectos de manera automática. Este proceso se realiza con el comando de la línea 280. A continuación (líneas 281 a 286) se especifican más características del edificio como el material principal, el uso o el número de plantas y habitaciones. La

distribución de estas se hará en forma de cuadrícula (el ejemplo resultante del código se puede ver en la Figura 3-28). Finalmente se crea un contenedor para almacenar y gestionar todas las edificaciones creadas.

```
266     #include "ns3/core-module.h"
267   #include "ns3/network-module.h"
268   #include "ns3/mobility-module.h"
269   #include "ns3/propagation-module.h"
270   #include "ns3/internet-module.h"
271   #include "ns3/applications-module.h"
272
273   // Configuración las coordenadas de las esquinas del edificio
274   Vector edificio1a (250.0, 250.0, 0.0);
275   Vector edificio1b (250.0, 300.0, 0.0);
276   Vector edificio1c (250.0, 500.0, 0.0);
277   Vector edificio1d (300.0, 500.0, 0.0);
278
279   // Creamos el objeto de edificio
280   Ptr<Building> building = CreateObject<Building> ();
281   building->SetBoundaries (edificio1a, edificio1b, edificio1c, edificio1d);
282   building->SetBuildingType (Building::Residential);
283   building->SetExtWallsType(Building::ConcreteWithWindows);
284   building->SetNFloors(3);
285   building->SetNRoomsX(3);
286   building->SetNRoomsY(2);
287
288   // Configuramos el escenario de la simulación
289   Ptr<BuildingContainer> edificios = CreateObject<BuildingContainer> ();
290   edificios->AddBuilding (building);
291   Ptr<OutdoorMesh> mesh = CreateObject<OutdoorMesh> ();
292   mesh->SetBuildingContainer (edificios);
293   mesh->Install (node);
```

Figura 3-27: Código de simulación de edificios (Autoría propia)

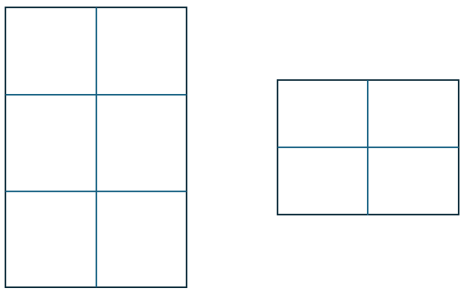

Figura 3-28: Esquema de la planta (izquierda) y alzado (derecha) del edifico creado (Autoría propia)

4. PRUEBAS REALIZADAS Y DISCUSIÓN DE RESULTADOS

Se plantea la realización de distintas pruebas utilizando para ello tres escenarios para probar la herramienta de simulación con la función de localización desarrollada. La primera prueba tratará de posicionar un dispositivo de usuario estático, la segunda llevará a cabo el mismo proceso, pero con el dispositivo en movimiento. Por último, en la tercera prueba se simulará un edificio mediante el código explicado en la sección 3.3.10 y se tratará igualmente de posicionar el dispositivo. El escenario básico elegido para todas las pruebas será el escenario urbano de baja densidad "`UMa`".

Figura 4-1: Posiciones que ocupará el dispositivo de usuario para la realización de las pruebas 1 y 2

El procedimiento diseñado para la evaluación de la herramienta de posicionamiento fue el siguiente. Para los dos primeros escenarios se simularon un total de 100 posiciones a lo largo de una cuadrícula de 500 por 500 metros en la cual las estaciones base ocuparán tres de sus vértices (coincidentes con lo descrito en el código, (0,0), (500,0) y (0,500) (representado gráficamente en la Figura 4-1). La forma de implementarlo en el código fue introduciendo todo el proceso de simulación en dos bucles iterativos anidados cada uno de los cuales se repetirá un total de 10 veces logrando en conjunto 100 iteraciones. Empleando las variables de control `x` e `y` se ajustan de igual forma las posiciones del dispositivo partiendo desde la posición (10,10). La simulación completa se repetirá tres veces dando como resultado final la media de las posiciones. Este proceso está resumido en pseudocódigo en la Figura 4-2.

```
for (x = 0; x<10; x++){
    for(y = 0;y<=10;y++){
        posición dispositivo(10 + x*10, 10 + y*10);
        proceso de simulación;
    }
}
```

Figura 4-2: Pseudocódigo explicativo de los bucles iterativos para la realización de las pruebas 1 y 2

Para la tercera prueba, el edifico tendrá las dimensiones descritas en la sección 3.3.10 situándose por tanto en la mitad superior de la cuadricula de pruebas. El dispositivo simulará moverse en un recorrido rectilíneo desde la posición (350, 10) hasta la (350, 500). En dicho recorrido pasará a estar en línea de visión directa con todas las antenas a estarlo solamente con una de ellas. Cuando esto ocurre, NS-3 asume que la propagación se realiza por multitrayecto suponiendo que se produce difracción en las esquinas del edificio y que desde allí las ondas llegan al dispositivo móvil. En este caso basta con realizar un único bucle iterativo que controle la variable con un total de 51 iteraciones con incrementos de 10 en 10 metros. De nuevo se usará la media de los resultados obtenidos de realizar tres posicionamientos como para los escenarios 1 y 2.

El presente capítulo está divido en 4 secciones. En la primera se realizan una serie de aclaraciones sobre el desarrollo de las distintas pruebas planteadas, así como de los modelos que se emplearon. Las tres siguientes desarrollan el posicionamiento de un dispositivo en estático, en movimiento y en presencia de edificaciones respectivamente.

4.1 Resultados aplicación de geolocalización

Tal y como se ha descrito, se realizaron tres bloques de pruebas. En todas ellas se utiliza el algoritmo basado en el empleo del método de mínimos cuadrados en dos dimensiones, descrito en la sección 3.3.9, el cual aporta como ventaja la simplicidad y como desventaja la de una menor precisión. Tal y como se desarrolló en la sección 2.4, se ha tenido en consideración que existen otros métodos que no se han estudiado en este trabajo debido a la complejidad de su implementación en la realización de esta prueba de concepto, aunque teniendo claro que algunos mostrarían una mayor precisión respecto a los resultados obtenidos con el algoritmo desarrollado. Además, estos métodos resultarían más robustos ante variaciones como los que se estudian en el tercer bloque de pruebas en el que el trayecto desde la estación base al dispositivo no se produce por la línea de visión directa.

Por otra parte, cabe destacar que los resultados obtenidos se basan en el seguimiento de paquetes realizado con herramientas estándar de NS-3 y no con las específicas del módulo 5G LENA ya que las mismas no están plenamente desarrolladas o la documentación existente no es lo suficientemente clara para desarrollar una función de análisis de tráfico que permita obtener los datos necesarios para aplicar el algoritmo de posicionamiento de forma satisfactoria. Es por ello por lo que los datos obtenidos pudieran ser de mayor calidad en caso de emplear una función de análisis de tráfico específica de 5G.

Por último, resulta imprescindible establecer de manera adecuada tanto el modelo de canal como el modelo de propagación. Como se expuso en la explicación de la subsección 3.3.10 al configurar el escenario, la simulación asigna automáticamente los valores adecuados al escenario para ambos modelos de canal. Sin embargo, es muy recomendable asegurarse de que dicha asignación se haya hecho correctamente ya que, tal y como ocurrió en las pruebas iniciales de la simulación, los resultados obtenidos de tiempo eran idénticos a pesar de que el UE estuviera en distintas posiciones. El problema vino provocado por una asignación incorrecta de esos valores del escenario en los modelos de propagación y canal, provocando que la distancia no influyera en el cálculo provocando que el tiempo fuese igual indistintamente de la posición.

Realizadas estas aclaraciones se procede ahora a comentar los resultados obtenidos en las distintas pruebas.

4.2 Localización del dispositivo en estático

Los datos obtenidos para un dispositivo que se encuentra estático durante el transcurso de la simulación muestran un error medio de 7,53 metros y una mediana de 7,50. En la Figura 4-3 se pueden observar las posiciones reales donde se posicionó el dispositivo de usuario y en rojo la posición estimada obtenida a través de la función de localización. De la misma forma se ha destacado con rombos verdes las posiciones de las estaciones base. A la vista de los resultados parece existir un cierto sesgo en los errores pudiendo ser este sistemático.

Figura 4-3: Resultados en el posicionamiento de un dispositivo estático

A fin de comprobar la existencia o no de un error sistemático, se realizó una variación del escenario trasladando la antena de la posición (0,500) a la posición (500, 500) de tal forma que haya más antenas en la parte derecha del escenario. Este error radicaría en el hecho de que el programa al aplicar mínimos cuadrados para obtener la resolución analítica más aproximada tienda a aproximarse más a una zona que a otra.

Los resultados de esta segunda prueba se muestran en la Figura 4-4. En este caso se obtiene un error medio de 25,98 metros y una mediana de 25,22. En la figura es claramente observable como de nuevo parece haber algún tipo de error sistemático mostrando de nuevo una tendencia a posiciones más a la izquierda de las reales. Revisitando el código desarrollado y consultando la documentación de NS-3 se deduce que la causa de este sesgo puede estar en los números pseudoaleatorios empleados para hacer la simulación más realista.

Figura 4-4: Resultados en el posicionamiento de un UE estático con modificación en el posicionamiento de las estaciones base.

Por último, a fin de comprobar la posible causa del error aleatorio se realizan simulaciones independientes en lugar de una sola con el empleo de un bucle reiterativo `for`. La Figura 4-5 muestra cómo tras realizar la simulación en tiempos de ejecución distintos en cada iteración sí se logran errores de tipo aleatorio con un error medio de 8,87 metros y una mediana de 7,15 metros. El coste de esta mayor coherencia en los resultados es el tiempo empleado en realizar las simulaciones de manera independiente por lo que en caso de querer realizar un número de pruebas mayor habría que valorar encontrar una forma de reiniciar la semilla de número pseudoaleatorios en cada iteración. En la Figura 4-6 se expone la distribución de errores donde el 39% de los casos está por debajo de los 5 metros y el 70% está por debajo de los 9 metros de error.

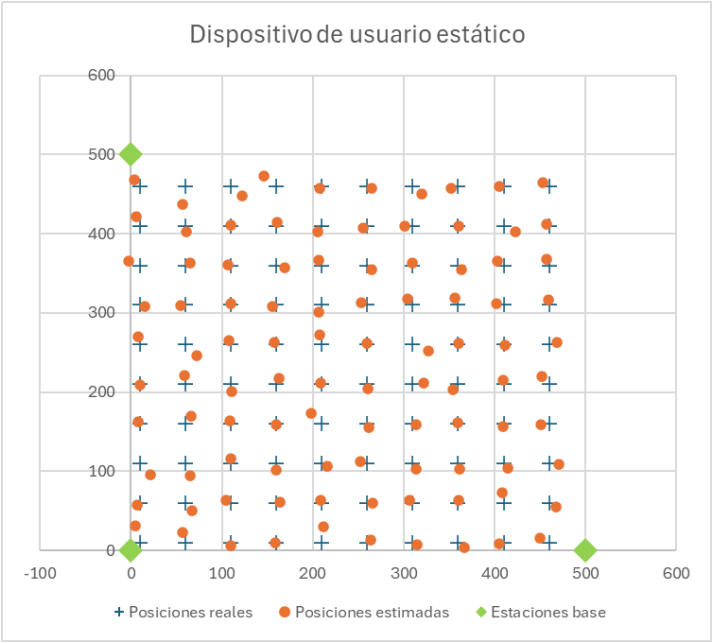

Figura 4-5: Resultados en el posicionamiento realizando la simulación en tiempos de ejecución distintos.

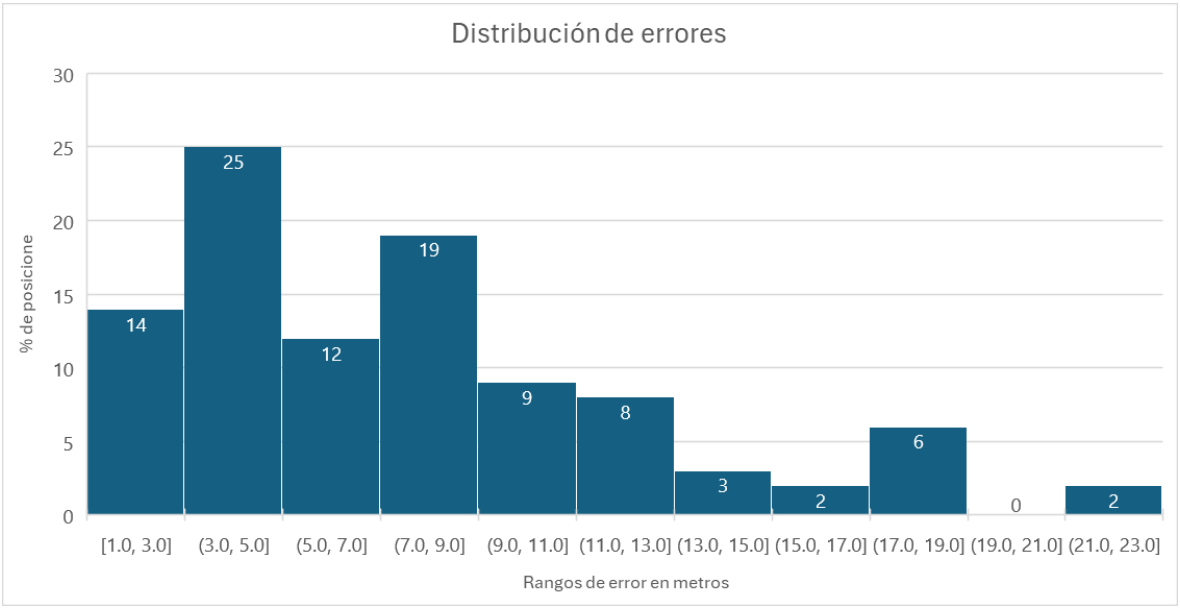

Figura 4-6: Distribución de errores en posicionamiento estático

4.3 Localización del dispositivo en movimiento

Para la segunda prueba se tuvo en cuenta desde el comienzo el error que podía surgir de no renovar la semilla de números pseudoaleatorios en cada iteración. Se simularon dos velocidades distintas, 10 y 30 metros por segundo tratando de nuevo estimar la posición de los dispositivos comparada con las que realmente ocupan. Para ello basta con cambiar en el código de la simulación el modelo de movilidad asignado al dispositivo de usuario y otorgarle la velocidad deseada. La forma de proceder con las mediciones será de tal forma que, aunque el dispositivo tenga ahora velocidad, este pase por las mismas posiciones que se determinaron para la primera prueba y la toma de datos coincida con el paso por dichas posiciones.

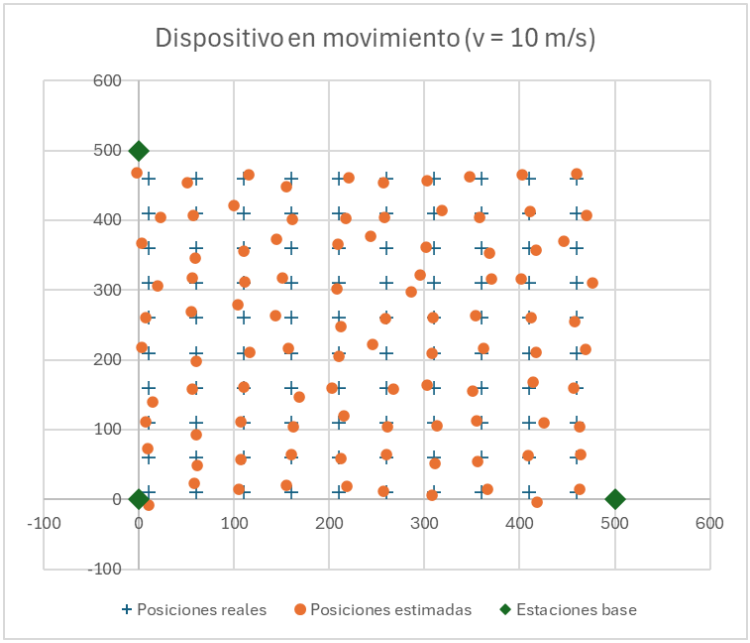

Figura 4-7: Resultados posicionamiento para un dispositivo con velocidad (v = 10m/s)

Para la primera de las velocidades elegidas se obtuvo un error máximo de 44,64 metros y uno mínimo de 2,01 metros. La media de los errores fue de 9,33 metros y una mediana de 8.09 metros. Se observa por tanto que la variación respecto al caso estático es muy leve, aunque ligeramente superior, en cualquier caso. La Figura 4-7 muestra los resultados para la primera velocidad seleccionada. En la Figura 4-8 se puede observar la distribución porcentual de errores donde el 41% está por debajo de los 7 metros de error y el 72% por debajo de los 11 metros de error.

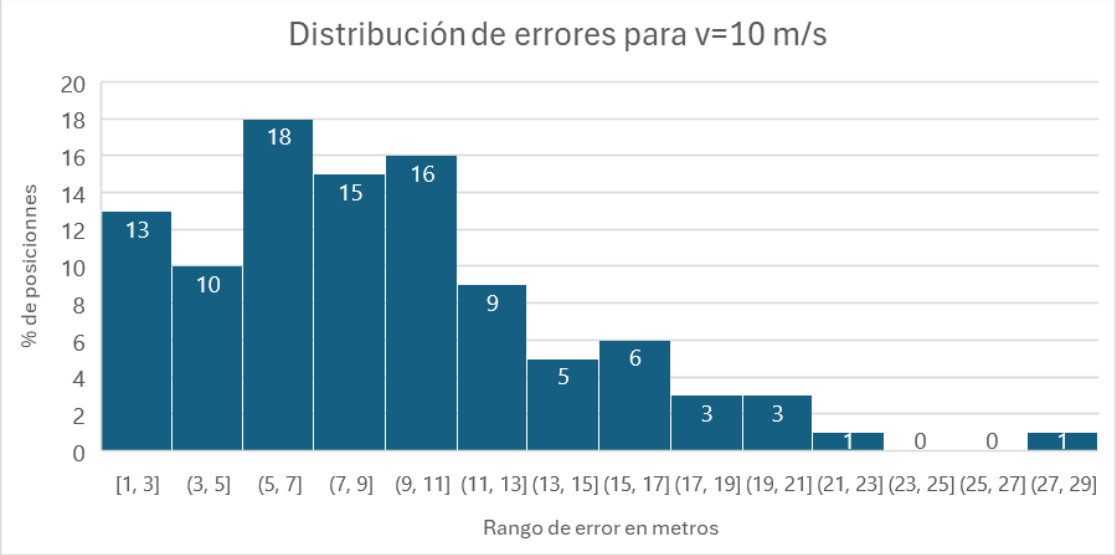

Figura 4-8: Distribución de errores en posicionamiento dinámico (v=10 m/s)

La Figura 4-9 muestra los resultados obtenidos para la velocidad de 30 m/s. Aunque no es aparente a simple vista el error medio ha aumentado a 9,77 metros y la mediana a 8,35 metros. Se puede concluir por tanto que la velocidad sí influye en la presión del posicionamiento, aunque de manera leve. Este resultado puede resultar de interés para el posicionamiento de vehículos autónomos, caso que se plantea como principal utilidad de una función de posicionamiento. La distribución de errores en este caso (Figura 4-10) muestra el 44% de los casos por debajo de los 7 metros de error y el 73% por debajo de los 13 metros.

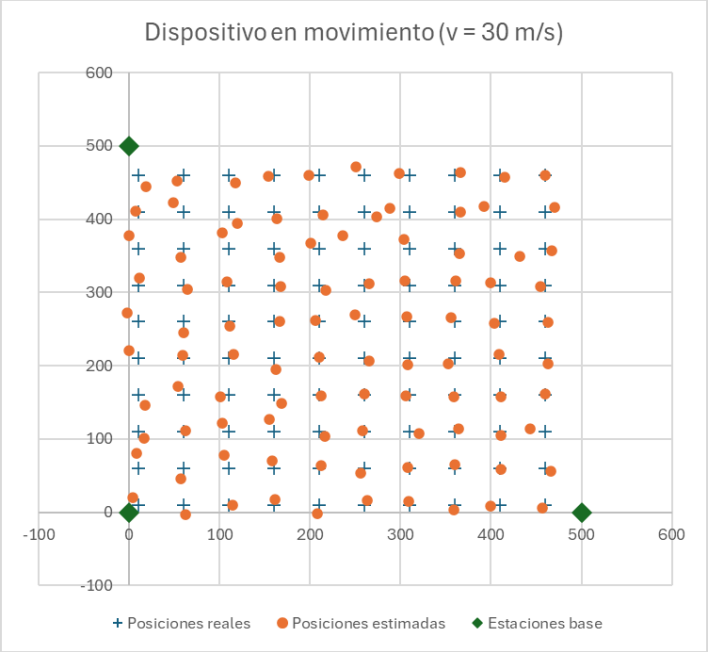

Figura 4-9 Resultados posicionamiento para un dispositivo con velocidad (v = 30 m/s)

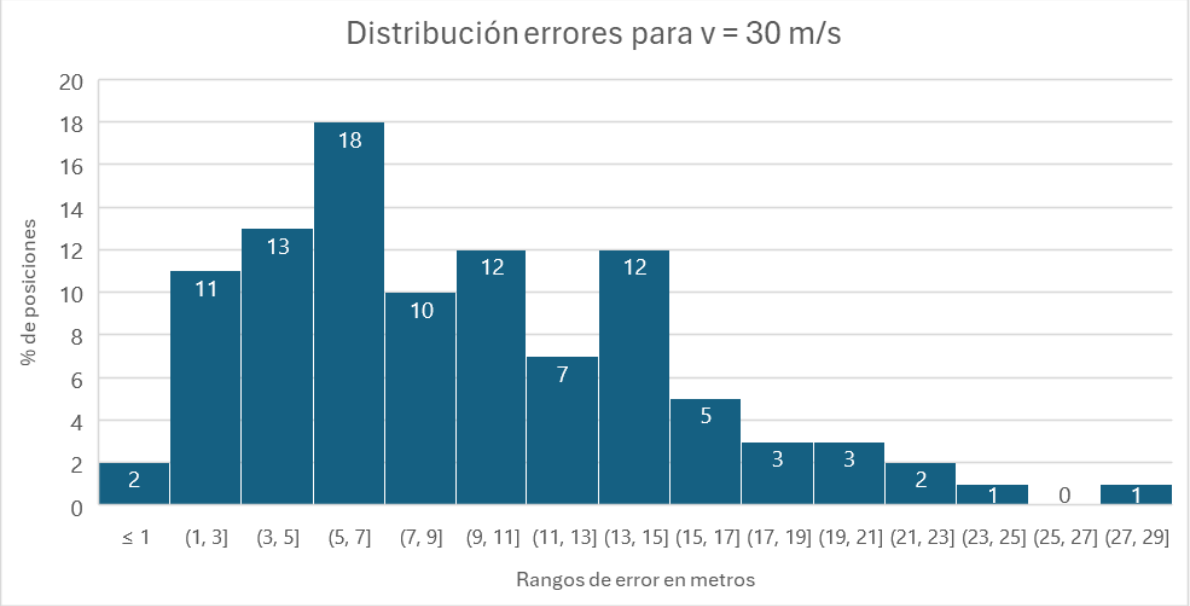

Figura 4-10: Distribución de errores en posicionamiento dinámico (v=30 metros)

4.4 Localización del dispositivo con la presencia de edificios

El caso de la ubicación de edificios demuestra lo susceptible que resulta la aplicación a cambios en el tiempo y por ello la distancia estimada. En la Figura 4-11 se puede ver como el dispositivo que inicialmente es capaz de obtener posiciones coherentes, pero en el momento en el cual se pierde la línea de visión directa entre el dispositivo y al menos una de las estaciones base, el tiempo de llegada de los paquetes aumenta provocando que el algoritmo de mínimos cuadrados obtenga resultados erróneos los cuales tienden a concentrarse en torno al centro del edificio. De las posiciones obtenidas con resultados coherentes se extrae un error medio de 8,87 metros y una mediana de 7,15 metros. Se observa que estos valores son ligeramente superiores a los obtenidos en 4.2.1 por lo cual se puede deducir que la propagación se ve afectada aun existiendo línea de visión directa.

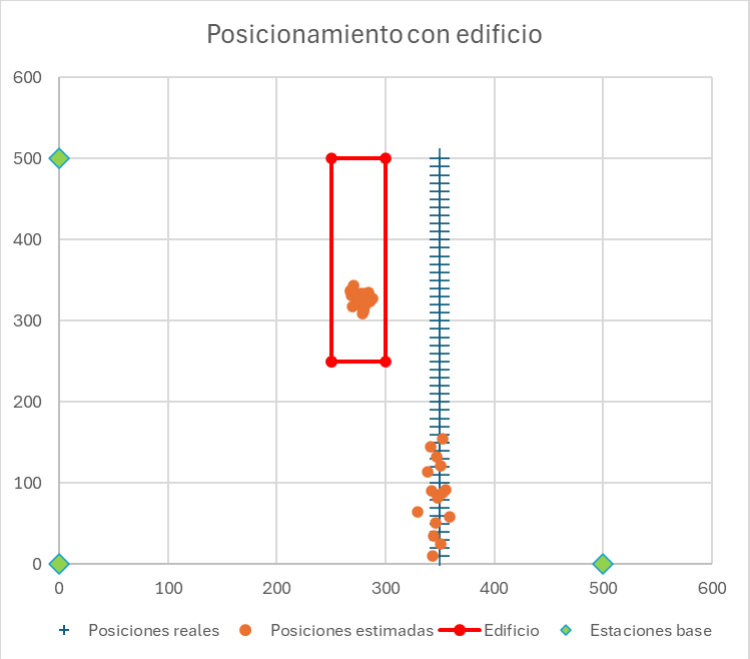

Figura 4-11: Resultados de las posiciones estimadas en un recorrido en el que se pierde la línea de visión directa con una o varias de las estaciones base

La conclusión por tanto de esta prueba es que el algoritmo de posicionamiento implementado solo funciona con modelos de propagación por rayo directo y no es capaz de obtener resultados coherentes cuando se trata con señales fruto de la composición de ondas recibidas por diversos fenómenos como la propagación por multitrayecto o la penetración de objetos. En la Figura 4-12 se puede ver la distribución de errores para las posiciones con errores coherentes con lo esperable con un 80% de las posiciones con un error por debajo de los 11 metros.

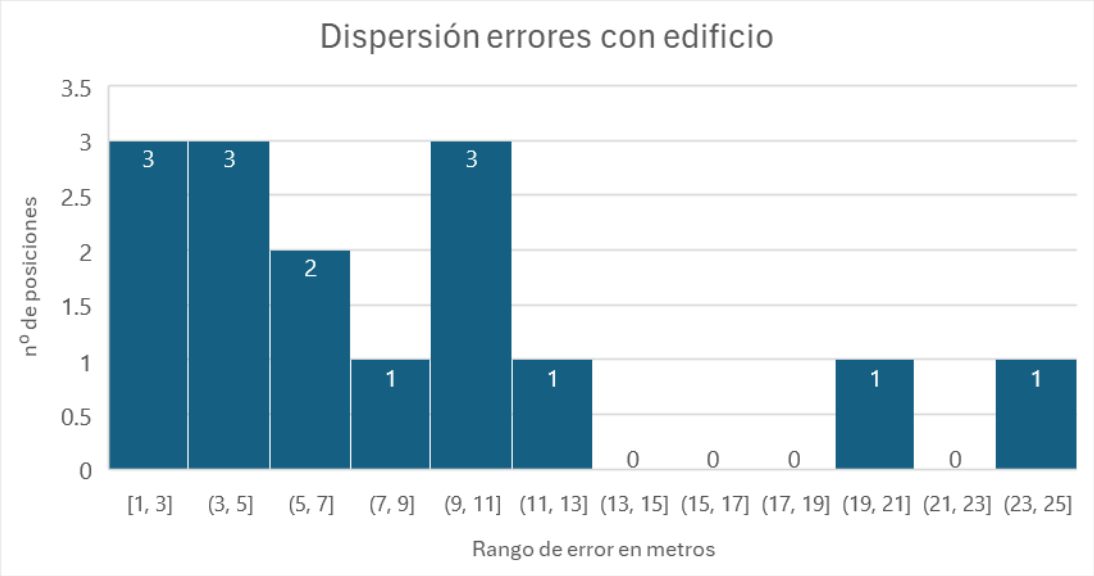

Figura 4-12: Distribución de los errores obtenidos para el posicionamiento con edificios

5. CONCLUSIONES Y LÍNEAS FUTURAS

5.1 Estado actual del desarrollo de burbujas tácticas

El avance de la tecnología 5G, aunque ciertamente veloz es también gradual y progresivo tal y como se expuso en la sección 2.1.3 pues aún no se ha alcanzado plenamente el estándar fijado para lograr una infraestructura de red 5G SA. LA 3GPP sigue trabajando con diligencia en nuevos estándares que exploran y exprimen al máximo las capacidades del 5G. Hay incluso trabajos desarrollados sobre el paso a la siguiente generación.

En este contexto las burbujas 5G son un reto a la par que una oportunidad. Reto pues se trata de desarrollar un sistema útil para su empleo por parte de los cuerpos y fuerzas de seguridad en escenarios donde los errores pueden suponer consecuencias fatales. Oportunidad, pues al no tener que lidiar con la retrocompatibilidad con otras generaciones, las burbujas 5G nacen libres de estas restricciones y pueden aprovechar todo el potencial de las ventajas del 5G SA descritas. Asimismo, los futuros usuarios y mantenedores de estos sistemas son miembros de organizaciones donde el manejo de sistemas de comunicaciones similares es habitual y por ello su adiestramiento en el uso de esta tecnología sería más sencillo. Unido a esto, al ser el 5G una tecnología originaría del mundo civil, muchas de sus características y dispositivos resultan familiares para el usuario. El ejemplo más claro con los teléfonos móviles rugerizados que solo se diferencian de un dispositivo civil en su mayor resiliencia física y cibernética.

El desarrollo de las burbujas tácticas no es un mero proyecto sino una realidad que va a aparecer en los años venideros. Por ello la formación y culturización sobre las capacidades que pueden aportar estos sistemas es crucial para que de esa manera los usuarios entiendan el funcionamiento interno de esta herramienta. Así, igual que comprenden el uso de otros instrumentos habituales para ellos (armas de fuego, vehículos, equipamiento), al comprender el uso de una nube táctica 5G, podrán destacar errores y proponer mejoras que mejoren su eficacia y eficiencia a la hora de desarrollar sus roles en el teatro de operaciones.

5.2 Empleo de 5G-LENA/NS-3 como simulador de una burbuja táctica

NS-3 es como se describió en la sección 2.3.1 una herramienta de simulación de redes de código libre. Está característica le otorga sus dos principales ventajas: una es la gran flexibilidad y abanico de posibilidades que ofrece a un desarrollador para crear una simulación acorde a las características que desea y la otra ventaja es la abundante documentación elaborada por la comunidad de usuarios de la que

se puede disponer de manera libre a través de la red. Esta misma comunidad es muy activa en los foros creados por NSAM con lo que es fácil encontrar apoyo o ayuda en caso de encontrarse con problemas.

Por otra parte, 5G LENA es un módulo específico de NS-3 desarrollado por el CTTC que pese a compartir el mismo enfoque de código libre y colaboración entre usuario, aún no goza de la popularidad de NS-3 y por ello la información necesaria para trabajar adecuadamente con las herramientas que este módulo aporta no siempre es fácilmente accesible.

De cara a desarrollar una simulación de una burbuja 5G se ha de tener en cuenta que ciertos aspectos de su desarrollo o de las conclusiones extraídas pueden ser susceptibles de reserva. Como se discutió en la sección 2.2.1, las redes de comunicaciones militares han de ser seguras en lo que se refiere a la información y por ende el código desarrollado para una simulación de una de estas redes no puede ser de acceso público. Esto choca de forma evidente con la filosofía abierta de las plataformas empleadas, pero solo en el caso de que en la simulación intervengan elementos o módulos susceptibles de contener información a proteger.

Se concluye por tanto que existen tres posibles vías de trabajo. En primer lugar, se puede optar por desarrollar una simulación de una nube táctica completa cuyo contenido sea público con lo que ello implicaría para mantener la superioridad en el desarrollo de la tecnología. La segunda opción sería lograr un acuerdo de colaboración con los organismos implicados (NSAM y CTTC) para desarrollar simulaciones que gocen de soporte técnico junto con el adecuado nivel de confidencialidad. La tercera vía sería la creación de un software propio que cumpla con las características deseadas manteniendo el uso de dicha herramienta de simulación limitada al ámbito privado de las agencias de defensa y seguridad implicadas.

5.3 Desarrollo de la función de localización

Respecto a la integración de una función de localización, se ha de destacar que la integración de este tipo de funciones no es un artificio limitado a un uso en simuladores. La arquitectura basada en software permite que una función plenamente operativa dentro del contexto virtual de un simulador sea fácilmente transferible e integrable dentro del núcleo de red 5G. La utilidad de esta característica resulta de enorme interés para el desarrollo de aplicaciones en el contexto de una nube táctica ya que toda aplicación o sistema cuyo fin último sea ser empleado en el teatro de operaciones ha de ser sometido a numerosas pruebas de calidad. Por tanto, el alto grado de compatibilidad tanto con las simulaciones como con la realidad facilita enormemente el proceso de pruebas al que se puede someter una utilidad destinada a funcionar dentro de la burbuja 5G.

En referencia a los resultados obtenidos como tal en las pruebas de localización, se puede concluir que estos se ajustan de manera coherente con los resultados esperados de un sistema de localización mediante el uso de sistemas 5G, obteniendo unos errores de en torno a los 9 y los 10 metros lo cual se ajusta a lo estudiado en otros trabajos similares como el de la referencia [85] donde se obtuvieron errores medios menos de 10 metros en el 80% de los casos empleando un único método de posicionamiento (ToA, AoA o CP) de Es cierto de igual manera que tal y como se mencionó en la sección 4.1. ni el método de posicionamiento empleando exclusivamente TDOA ni el algoritmo de resolución empleado, siendo este un método puramente algebraico, son las herramientas ideales para lograr los niveles de precisión deseados para un sistema de esta clase destinado por ejemplo a la gestión de vehículos autónomos.

La mejora del sistema de posicionamiento desarrollado pasa por la integración y comparación de varios métodos de posicionamiento en una sola aplicación (combinación de TDOA con AOA por ejemplo) así como por realizar una composición de las posiciones obtenidas por otros tipos de señal (GPS, GLONASS o redes wifi) para así lograr llegar a niveles de error menores del metro, críticos en el

desarrollo de ciertas aplicaciones cuyo empleo también puede ser de utilidad en el contexto de una nube táctica. Un ejemplo de ello sería un dron autónomo con misiones de desactivación de minas que tenga que establecer una serie de corredores seguros enviando luego las coordenadas de los puntos limpiados al resto de unidades de la red táctica.

5.4 Líneas futuras

El desarrollo de las burbujas tácticas pasa por realizar pruebas con prototipos o sistemas lo más parecido posibles a los que se pretendan emplear en un escenario real. Así mismo el desarrollo y refinado de los sistemas de simulación ha de orientarse no solo a realizar ensayo de mayor calidad sino a crear herramientas de predicción. De esta forma el organismo que actúe de mando de la operación que se quiera llevar a cabo contará con un sistema que le ayude a obtener datos críticos para realizar el despliegue de una nube táctica 5G.

El presente trabajo ha empleado como guías fundamentales el *release 15* y el *release 16* cuyos estándares están desarrollados de manera robusta. En esta línea de desarrollo sería de gran interés el estudio de documentos de la 3GPP posteriores a los mencionados en los cuáles se desarrollan nuevos usos y aplicaciones de interés para su empleo en el contexto de una burbuja táctica. Un ejemplo de ello serían las NTN's (*Non Terrestrial Networks*) cuyo objetivo es lograr instalar un núcleo de red completo junto con las antenas necesarias en una plataforma situada a gran altura (dron, aeronave o satélite en órbita LEO o MEO) proporcionando una cobertura a un área mucho más extensa. Este ejemplo sería de gran utilidad para el desarrollo de burbujas tácticas de gran alcance.

La interconexión de distintas burbujas tácticas asociadas cada una a diversos grupos operativos, así como la conexión de la burbuja con el exterior son también otros de los posibles campos de trabajo dónde es necesaria más investigación y trabajo.

Por último, respecto a la aplicación de localización, es necesario trabajar en la robustez de los resultados obtenidos, así como en sistemas que permitan mayor precisión ya sea implementando un método distinto (uso del desfase en las señales recibidas) o mejorando el algoritmo de cálculo (uso de métodos probabilísticos y estadísticos). En esta línea habría que investigar si existe la posibilidad de desarrollar dicha aplicación dentro de una herramienta de simulación. Si no es así caben tanto la posibilidad de hacer un módulo propio que implemente estas funciones, o realizar la investigación con prototipos de dispositivos y antenas 5G reales. Por otra parte, el desarrollo de estudios estadísticos de los errores obtenidos tanto en simulaciones iguales a la desarrollada como en similares arrojarían más luz sobre los efectos de la velocidad en la presión del posicionamiento.

6. BIBLIOGRAFÍA

[1] «5G System Overview». Accedido: 18 de diciembre de 2023. [En línea]. Disponible en: https://www.3gpp.org/technologies/5g-system-overview

[2] F. J. G. García, «5G: Aplicación de las comunicaciones móviles en el ámbito de la defensa».

[3] R. D. InfoDefensa, «Telefónica, elegida por la Armada para implantar sus primeras redes 5G», Infodefensa - Noticias de defensa, industria, seguridad, armamento, ejércitos y tecnología de la defensa. Accedido: 16 de marzo de 2024. [En línea]. Disponible en: https://www.infodefensa.com/texto-diario/mostrar/4414957/armada-sigue-pasos-ejercito-tierra-encarga-telefonica-primeras-redes-5g

[4] «¿cómo funciona una red móvil? - Las ondas», https://radio-waves.orange.com/. Accedido: 24 de enero de 2024. [En línea]. Disponible en: https://radio-waves.orange.com/es/como-funciona-una-red-movil/

[5] «Gestión de la movilidad – Rubén Sánchez». Accedido: 31 de enero de 2024. [En línea]. Disponible en: http://rubensm.com/gestion-de-la-movilidad/

[6] «¿Quién inventó el primer teléfono móvil? - Telefónica». Accedido: 24 de enero de 2024. [En línea]. Disponible en: https://www.telefonica.com/es/sala-comunicacion/blog/quien-invento-primer-telefono-movil/

[7] J. Ramírez Ruiz, «Despliegue de un prototipo de red móvil 5G completo». Accedido: 31 de enero de 2024. [En línea]. Disponible en: https://oa.upm.es/72303/

[8] «GSM». Accedido: 31 de enero de 2024. [En línea]. Disponible en: http://isa.uniovi.es/domotica/Temas/T3/T3-GSM

[9] «Cuáles son las diferencias entre E, GPRS, 3G, 4G, 5G y esas otras redes a las que se conecta tu celular (y cómo te afectan tu conexión a internet)», *BBC News Mundo*. Accedido: 31 de enero de 2024. [En línea]. Disponible en: https://www.bbc.com/mundo/noticias-37247130

[10] «Introducing 3GPP», 3GPP. Accedido: 31 de enero de 2024. [En línea]. Disponible en: https://www.3gpp.org/about-us/introducing-3gpp

[11] «La historia de Blackberry, la compañía que revolucionó la industria de los dispositivos móviles y fracasó por falta de visión». Accedido: 31 de enero de 2024. [En línea]. Disponible en: https://www.negociosyemprendimiento.org/2022/07/historia-blackberry.html

[12] por C. F. Aparicio, «¿Qué es el ADSL?», JVS Informática Blog. Accedido: 31 de enero de 2024. [En línea]. Disponible en: https://www.jvs-informatica.com/blog/glosario/adsl/

[13] «¿Qué es el 3G | Lenovo España»? Accedido: 31 de enero de 2024. [En línea]. Disponible en: https://www.lenovo.com/es/es/faqs/pc-life-faqs/que-es-3g/

[14] «El acceso a banda ancha. 3G, 4G y Wimax | Blog SEAS». Accedido: 1 de febrero de 2024. [En línea]. Disponible en: https://www.seas.es/blog/informatica/el-acceso-a-banda-ancha-3g-4g-y-wimax/

[15] P. Alonso García, «Caracterización de los servicios ofrecidos en redes Utran, modelado de su consumo de recursos y análisis de su grado de servicio», info:eu-repo/semantics/doctoralThesis, 2012. doi: 10.35376/10324/2623.

[16] «Qué es la tecnología MIMO y para qué se utiliza en redes WiFi», RedesZone. Accedido: 1 de febrero de 2024. [En línea]. Disponible en: https://www.redeszone.net/tutoriales/redes-wifi/tecnologia-mimo-red-wifi-que-es/

[17] «Teleco in a nutshell v8.7: Modulación de Amplitud en Cuadratura». Accedido: 1 de febrero de 2024. [En línea]. Disponible en: https://www.flu-project.com/2019/11/teleco-in-nutshell-v87-modulacion-amplitud-cuadratura.html

[18] «Qué es LTE: Cómo funciona y por qué es importante». Accedido: 1 de febrero de 2024. [En línea]. Disponible en: https://es.digi.com/blog/post/what-is-lte

[19] «LTE Tutorial: SAE Technology». Accedido: 1 de febrero de 2024. [En línea]. Disponible en: https://www.artizanetworks.com/resources/tutorials/sae_tec.html

[20] «Specification # 28.612». Accedido: 1 de febrero de 2024. [En línea]. Disponible en: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1537

[21] «¿Qué es LTE Advanced? - Teldat». Accedido: 8 de febrero de 2024. [En línea]. Disponible en: https://www.teldat.com/es/blog/que-es-lte-advanced/

[22] «Release 15». Accedido: 10 de enero de 2024. [En línea]. Disponible en: https://www.3gpp.org/specifications-technologies/releases/release-15

[23] «LTE-Advanced PRO establece las bases para 5G - Teldat». Accedido: 14 de febrero de 2024. [En línea]. Disponible en: https://www.teldat.com/es/blog/lte-advanced-pro-establece-las-bases-para-5g/

[24] C. de la Cuesta de Bedoya, «5G como vector para la digitalización y robotización del teatro de operaciones», Centro Universitario de la Defensa, Escuela Naval Militar, 24 de enero de 2024.

[25] redeweb, «Un gran avance 5G para la transmisión de alta velocidad». Accedido: 4 de marzo de 2024. [En línea]. Disponible en: https://www.redeweb.com/articulos/un-gran-avance-5g/

[26] González, Jackson; Salamanca, Oscar, «El camino hacia el 5g», Universidad Privada Dr. Rafael Belloso.

[27] W. Xia, Y. Wen, C. H. Foh, D. Niyato, y H. Xie, «A Survey on Software-Defined Networking», *IEEE Commun. Surv. Tutor.*, vol. 17, n.º 1, pp. 27-51, 2015, doi: 10.1109/COMST.2014.2330903.

[28] A. O. Watanabe, M. Ali, S. Y. B. Sayeed, R. R. Tummala, y M. R. Pulugurtha, «A Review of 5G Front-End Systems Package Integration», *IEEE Trans. Compon. Packag. Manuf. Technol.*, vol. 11, n.º 1, pp. 118-133, 2021, doi: 10.1109/TCPMT.2020.3041412.

[29] «Release 16», 3GPP. Accedido: 4 de marzo de 2024. [En línea]. Disponible en: https://www.3gpp.org/specifications-technologies/releases/release-16

[30] J. M. López, «De Open RAN a Cloud RAN: el 5G evoluciona en la nube», Blogthinkbig.com. Accedido: 20 de enero de 2024. [En línea]. Disponible en: https://blogthinkbig.com/cloud-ran-5g

[31] S. He, L. Guo, Y. Guo, C. Wu, M. Ghanem, y R. Han, «Elastic Application Container: A Lightweight Approach for Cloud Resource Provisioning», en *2012 IEEE 26th International Conference on Advanced Information Networking and Applications*, 2012, pp. 15-22. doi: 10.1109/AINA.2012.74.

[32] «What is the 5G Service-Based Architecture (SBA)?», TECHCOMMUNITY.MICROSOFT.COM. Accedido: 4 de marzo de 2024. [En línea]. Disponible en: https://techcommunity.microsoft.com/t5/azure-for-operators-blog/what-is-the-5g-service-based-architecture-sba/ba-p/3831367

[33] «5G - New Radio - LS telcom». Accedido: 4 de marzo de 2024. [En línea]. Disponible en: https://www.lstelcom.com/es/casos-de-uso/5g-new-radio/

[34] H. Yin y S. Alamouti, «OFDMA: A Broadband Wireless Access Technology», en *2006 IEEE Sarnoff Symposium*, 2006, pp. 1-4. doi: 10.1109/SARNOF.2006.4534773.

[35] F. W. Vook, A. Ghosh, E. Diarte, y M. Murphy, «5G New Radio: Overview and Performance», en *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, 2018, pp. 1247-1251. doi: 10.1109/ACSSC.2018.8645228.

[36] «EEI: Alcances de Massive MiMo en 5G». Accedido: 1 de abril de 2024. [En línea]. Disponible en: https://www.arastechnologies.com/2019/04/eei-alcances-de-massive-mimo-en-5g.html

[37] E. Ali, M. Ismail, R. Nordin, y N. F. Abdulah, «Beamforming techniques for massive MIMO systems in 5G: overview, classification, and trends for future research», *Front. Inf. Technol. Electron. Eng.*, vol. 18, n.º 6, pp. 753-772, jun. 2017, doi: 10.1631/FITEE.1601817.

[38] chema, «How Network Slicing works and why it is key to 5G», Telefónica. Accedido: 5 de marzo de 2024. [En línea]. Disponible en: https://www.telefonica.com/en/communication-room/blog/how-network-slicing-works-and-why-it-is-key-to-5g/

[39] «Qué son las VLAN, para qué sirven y cómo funcionan con ejemplos de uso», RedesZone. Accedido: 5 de marzo de 2024. [En línea]. Disponible en: https://www.redeszone.net/tutoriales/redes-cable/vlan-tipos-configuracion/

[40] «5G Network Slicing, What is it? | 5G Slicing Architecture and Solutions». Accedido: 5 de marzo de 2024. [En línea]. Disponible en: https://www.viavisolutions.com/en-us/5g-network-slicing

[41] S. Zhang, «An Overview of Network Slicing for 5G», *IEEE Wirel. Commun.*, vol. 26, n.º 3, pp. 111-117, 2019, doi: 10.1109/MWC.2019.1800234.

[42] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, y H. Flinck, «Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions», *IEEE Commun. Surv. Tutor.*, vol. 20, n.º 3, pp. 2429-2453, 2018, doi: 10.1109/COMST.2018.2815638.

[43] comonline, «Cisco, Telefónica y la Universidade de Vigo impulsan el 'Network Slicing' sobre 5G», Telefónica. Accedido: 5 de marzo de 2024. [En línea]. Disponible en: https://www.telefonica.com/es/sala-comunicacion/prensa/cisco-telefonica-y-la-universidade-de-vigo-impulsan-el-network-slicing-sobre-5g/

[44] M. Á. Navas, «¿Qué es la ley de Moore y para qué sirve?», Profesional Review. Accedido: 6 de marzo de 2024. [En línea]. Disponible en: https://www.profesionalreview.com/2018/04/01/que-es-la-ley-de-moore-y-para-que-sirve/

[45] N. Hassan, K.-L. A. Yau, y C. Wu, «Edge Computing in 5G: A Review», *IEEE Access*, vol. 7, pp. 127276-127289, 2019, doi: 10.1109/ACCESS.2019.2938534.

[46] «¿Qué es cloud computing o computación en la nube ?» Accedido: 6 de marzo de 2024. [En línea]. Disponible en: https://www.redhat.com/es/topics/cloud

[47] S. Barbarossa, S. Sardellitti, y P. Di Lorenzo, «Communicating While Computing: Distributed mobile cloud computing over 5G heterogeneous networks», *IEEE Signal Process. Mag.*, vol. 31, n.º 6, pp. 45-55, 2014, doi: 10.1109/MSP.2014.2334709.

[48] K. Cao, Y. Liu, G. Meng, y Q. Sun, «An Overview on Edge Computing Research», *IEEE Access*, vol. 8, pp. 85714-85728, 2020, doi: 10.1109/ACCESS.2020.2991734.

[49] «Creating User:NoMore201 - Wikimedia Commons». Accedido: 6 de marzo de 2024. [En línea]. Disponible en: https://commons.wikimedia.org/wiki/User:NoMore201

[50] J. A. S. Aranda, R. dos S. Costa, V. W. de Vargas, P. R. da S. Pereira, J. L. V. Barbosa, y M. P. Vianna, «Context-aware Edge Computing and Internet of Things in Smart Grids: A systematic mapping study», *Comput. Electr. Eng.*, vol. 99, p. 107826, 2022, doi: https://doi.org/10.1016/j.compeleceng.2022.107826.

[51] C. Zhao, Y. Cai, M. Zhao, y Q. Shi, «Joint Hybrid Beamforming and Offloading for mmWave Mobile Edge Computing Systems», en *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1-6. doi: 10.1109/WCNC.2019.8885934.

[52] R. Bajracharya, R. Shrestha, y H. Jung, «5G and Beyond Private Military Communication: Trend, Requirements, Challenges and Enablers», vol. 11, 2023.

[53] D. A. Eisenberg, D. L. Alderson, M. Kitsak, A. Ganin, y I. Linkov, «Network Foundation for Command and Control (C2) Systems: Literature Review», *IEEE Access*, vol. 6, pp. 68782-68794, 2018, doi: 10.1109/ACCESS.2018.2873328.

[54] W. Simpson, «IP in IP Tunneling», RFC Editor, RFC1853, oct. 1995. doi: 10.17487/rfc1853.

[55] «Política de los Sistemas y Tecnologías de la Información y las Comunicaciones del Ministerio de Defensa».

[56] «Los diferentes tipos de órbita de los satélites - Darwin Innovación». Accedido: 13 de marzo de 2024. [En línea]. Disponible en: https://darwincav.com/es/different-types-of-satellite-orbit/

[57] «Un millón de nuevos satélites esperan para llegar a la órbita de la Tierra». Accedido: 13 de marzo de 2024. [En línea]. Disponible en: https://www.elconfidencial.com/tecnologia/novaceno/2023-10-18/starlink-millon-satelites-saturar-orbita-terrestre-musk_3756785/

[58] «Informe 39: Nuevas armas contra la ética y las personas. Drones armados y drones autónomos», Delas. Accedido: 12 de marzo de 2024. [En línea]. Disponible en: https://centredelas.org/publicacions/nuevasarmascontraeticaypersonas/?lang=es

[59] G. Capela, W. Low, y L. Bastos, «5G for deployable and maritime communications», en *2021 International Conference on Military Communication and Information Systems (ICMCIS)*, 2021, pp. 1-7. doi: 10.1109/ICMCIS52405.2021.9486397.

[60] A. Lagorio, C. Cimini, R. Pinto, y S. Cavalieri, «5G in Logistics 4.0: potential applications and challenges», *Procedia Comput. Sci.*, vol. 217, pp. 650-659, 2023, doi: https://doi.org/10.1016/j.procs.2022.12.261.

[61] «Armada española: Fuerte despliegue militar con el portaaeronaves Juan Carlos I, Harrier, un buque de asalto, una fragata y un batallón». Accedido: 12 de marzo de 2024. [En línea]. Disponible en: https://www.eldebate.com/espana/defensa/armada/20240205/fuerte-despliegue-militar-portaaeronaves-juan-carlos-i-harrier-buque-asalto-fragata-batallon_172090.html

[62] S. M. Morales, «La herramienta más valiosa de una potencia naval: la proyección del poder en la zona litoral».

[63] M. Mezzavilla *et al.*, «End-to-End Simulation of 5G mmWave Networks», *IEEE Commun. Surv. Tutor.*, vol. 20, n.º 3, pp. 2237-2263, 2018, doi: 10.1109/COMST.2018.2828880.

[64] «Performance Evaluation of 5G Simulation Software | Top 6 List Explained | Network Simulation Tools». Accedido: 1 de abril de 2024. [En línea]. Disponible en: https://networksimulationtools.com/5g-simulation-software/

[65] S. Siraj, A. Gupta, y R. Badgujar, «Network simulation tools survey», *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 1, n.º 4, pp. 199-206, 2012.

[66] «INET Framework». Accedido: 1 de abril de 2024. [En línea]. Disponible en: https://omnetpp.org/download-items/INET.html

[67] «Logiciel démarche qualité : GED & processus». Accedido: 1 de abril de 2024. [En línea]. Disponible en: https://www.qualnet.fr/

[68] «Riverbed Modeler». Accedido: 1 de abril de 2024. [En línea]. Disponible en: https://support.riverbed.com/content/support/software/steelcentral-npm/modeler-index.html

[69] «NetSim™ Network Simulator™ & Router Simulator». Accedido: 1 de abril de 2024. [En línea]. Disponible en: https://www.boson.com/netsim-cisco-network-simulator

[70] «5G Toolbox». Accedido: 13 de marzo de 2024. [En línea]. Disponible en: https://es.mathworks.com/products/5g.html

[71] A. Solano-Barliza, «Revisión conceptual de sistemas de recomendación y geolocalización aplicados a la seguridad turística», *Comput. Electron. Sci. Theory Appl.*, vol. 2, pp. 37-43, dic. 2021, doi: 10.17981/cesta.02.02.2021.05.

[72] M. J. J. Martínez, A. M. Mateu, J. M. P. Asencio, y M. V. Cano, «Progreso en la práctica del ajuste gaussiano de una red local: método de triangulateración».

[73] Z. Yang y Y. Liu, «Quality of Trilateration: Confidence-Based Iterative Localization», *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, n.º 5, pp. 631-640, 2010, doi: 10.1109/TPDS.2009.90.

[74] F. VADILLO, «Una introducción a los problemas de mínimos cuadrados».

[75] F. Thomas y L. Ros, «Revisiting trilateration for robot localization», *IEEE Trans. Robot.*, vol. 21, n.º 1, pp. 93-101, 2005, doi: 10.1109/TRO.2004.833793.

[76] J. Nikonowicz, A. Mahmood, M. I. Ashraf, E. Björnson, y M. Gidlund, «Indoor Positioning in 5G-Advanced: Challenges and Solution towards Centimeter-level Accuracy with Carrier Phase Enhancements». arXiv, 26 de febrero de 2024. Accedido: 24 de marzo de 2024. [En línea]. Disponible en: http://arxiv.org/abs/2209.01183

[77] T.-Y. Chen, C.-C. Chiu, y T.-C. Tu, «Mixing and combining with AOA and TOA for the enhanced accuracy of mobile location», en *2003 5th European Personal Mobile Communications Conference (Conf. Publ. No. 492)*, 2003, pp. 276-280. doi: 10.1049/cp:20030261.

[78] «Carrier Phase - an overview | ScienceDirect Topics». Accedido: 2 de abril de 2024. [En línea]. Disponible en: https://www.sciencedirect.com/topics/engineering/carrier-phase

[79] S. Fan, W. Ni, H. Tian, Z. Huang, y R. Zeng, «Carrier Phase-Based Synchronization and High-Accuracy Positioning in 5G New Radio Cellular Networks», *IEEE Trans. Commun.*, vol. 70, n.º 1, pp. 564-577, 2022, doi: 10.1109/TCOMM.2021.3119072.

[80] M. Koivisto, A. Hakkarainen, M. Costa, P. Kela, K. Leppanen, y M. Valkama, «High-Efficiency Device Positioning and Location-Aware Communications in Dense 5G Networks», *IEEE Commun. Mag.*, vol. 55, n.º 8, pp. 188-195, 2017, doi: 10.1109/MCOM.2017.1600655.

[81] «ns-3 Installation Guide».

[82] «ns-3: ns-3 Documentation». Accedido: 19 de marzo de 2024. [En línea]. Disponible en: https://www.nsnam.org/doxygen/

[83] B. Bojovic, K. Koutlia, S. Lagen, N. Patriciello, Zoraze Ali, y L. Giupponi, «5G-LENA ns-3 nr module». [object Object], 23 de noviembre de 2022. doi: 10.5281/ZENODO.7780747.

[84] «Singular Value Decomposition - A Comprehensive guide on Singular Value Decomposition - Machine Learning Plus». Accedido: 3 de abril de 2024. [En línea]. Disponible en: https://www.machinelearningplus.com/linear-algebra/singular-value-decomposition/

[85] M. de los Á. López Fernández, «Positioning system of high-speed trains using 5G networks», 2022.

[86] J. M. López, «¿Por qué las redes 5G consumen menos energía?», Blogthinkbig.com. Accedido: 20 de enero de 2024. [En línea]. Disponible en: https://blogthinkbig.com/consumo-energia-5g

[87] «¿Es peligroso el 5G para la salud?», www.nationalgeographic.com.es. Accedido: 2 de abril de 2024. [En línea]. Disponible en: https://www.nationalgeographic.com.es/ciencia/es-peligroso-5g-para-salud_15514

[88] «Ciudades inteligentes y el rol del 5G en su funcionamiento». Accedido: 2 de abril de 2024. [En línea]. Disponible en: https://blogthinkbig.com/ciudades-inteligentes-5g

[89] O. Tremblay, L.-A. Dessaint, y A.-I. Dekkiche, «A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles», en *2007 IEEE Vehicle Power and Propulsion Conference*, Arlington, TX, USA: IEEE, sep. 2007, pp. 284-289. doi: 10.1109/VPPC.2007.4544139.

[90] A. Konert y T. Balcerzak, «Military autonomous drones (UAVs) - from fantasy to reality. Legal and Ethical implications.», *Transp. Res. Procedia*, vol. 59, pp. 292-299, 2021, doi: https://doi.org/10.1016/j.trpro.2021.11.121.

[91] E. E. País, «El dilema del tranvía: ¿debo sacrificar una vida para salvar cinco?», Verne. Accedido: 3 de abril de 2024. [En línea]. Disponible en: https://verne.elpais.com/verne/2017/03/27/articulo/1490625074_938459.html

ANEXO I: IMPLICACIONES SOCIALES, Y/O ECONÓMICAS, Y/O AMBIENTALES

El 5G tiene como uno de sus pilares ser una tecnología pensada para ser eficiente energéticamente [1]. Como se vio en la sección 2.1.3.3, el empleo de *beamforming* permite la utilización de la potencia disponible de una manera más inteligente. Unido a esto, al disponer de servicios de *edge computing* se puede emplear una IA que controle de manera inteligente el consumo acorde al número de usuarios y la demanda pudiendo activar periodos de descanso donde las emisiones totales se reduzcan. El caso más claro de esto sería durante la franja nocturna del día donde la mayoría de la población está durmiendo y por ello sus dispositivos no estarían en uso. Su empleo en el ámbito de la defensa no solo podría ayudar a economizar un recurso crítico como es la energía sino también reducir las emisiones electromagnéticas no deseadas, las cuales pueden ser empleadas por el oponente como fuente de información [86].

El rango de frecuencias empleado ha sido objeto de diversos estudios debido a las afirmaciones por parte de ciertos grupos sociales sobre el potencial nocivo de las ondas 5G en la salud. Pese a ello ninguno de los estudios realizados hasta la fecha de realización del trabajo ha demostrado que efectivamente el 5G sea dañino para el ser humano y el medio ambiente en circunstancias normales [87]. Por circunstancias normales se entiende que existe una cierta distancia de la fuente de emisiones y que la radiación recibida es de baja potencia.

Por otra parte, el 5G se plantea como uno de los cimientos de las llamadas ciudades inteligentes (*Smart cities*). Estas se basan en el despliegue masivo de dispositivos IoT y sensores estableciendo un "sistema nervioso" urbano. El análisis de la información recibida permitirá aumentar la eficiencia energética y disminuir la contaminación. Un ejemplo de ello puede ser el control inteligente de la calidad del aire. Mediante diversos sensores localizados en distintas zonas urbanas se podría medir la calidad del aire y en caso de alcanzar niveles por debajo de la establecido, emitir una alerta que llegue a los diversos dispositivos de control de tráfico (semáforos, pantallas o señales luminosas) y así desviar el tráfico de esa zona para mejorar la calidad del aire de la zona [88].

El presente trabajo ha utilizado la herramienta de simulación NS-3. Esta incorpora un módulo denominado e*nergy framework* que incorpora funciones y objetos predefinidos que permiten simular la generación y consumo de energía en los distintos nodos y elementos de la red simulada. El conocimiento de este consumo de forma al despliegue a nivel real de la infraestructura 5G puede ayudar a optimizar el diseño de los elementos que han de alimentar a la red, evitando un sobredimensionamiento excesivo que pueda conducir a un gasto energético por encima de lo necesario [89].

ANEXO II: REFLEXIONES ÉTICAS Y SOCIALES

El ámbito de estudio del presente trabajo es el de la seguridad y la defensa, un ámbito con evidentes implicaciones éticas. Aunque las Fuerzas Armadas no tienen como primera reacción el empleo de fuerza letal que pueda acabar con vidas humanas, su función última como defensores de la soberanía e intereses de un país los podría obligar a emplear dicha fuerza, situación para la que se adiestran continuamente.

En un principio podría parecer que el empleo de una nube táctica 5G no tiene implicaciones éticas como las podrían tener las armas de fuego convencionales. Esta afirmación es cierta en lo que se refiere a la red en sí misma, pero no en cuanto a los servicios a los que da soporte. El caso más paradigmático es el empleo de drones [52]. El empleo de e*dge computing* habilita la posibilidad de desplegar drones que actúen de manera autónoma respecto a unos protocolos establecidos. Este manejo autónomo lo realizaría una IA hospedada en el servidor *edge* de la red. Un dron con simples funciones de exploración no supone ningún problema, pero la existencia de posibles sistemas autónomos armados si acarrea consigo un dilema.

Se podría apelar al hecho de que, en última instancia, la orden de fuego ha de ser aprobada por un operador humano. Sin embargo, seguirían existiendo casos en las que el tiempo de reacción necesario ha de ser mínimo como por ejemplo en caso de detectar un misil en aproximación que deba ser derribado. La idea es que los protocolos que se provean al sistema sean lo suficientemente detallados para tener en cuenta todas las posibles situaciones. Pero pese a todo ello, se seguiría llegando a situaciones moralmente ambiguas en las que se ha de elegir entre el menor de dos males lo cual conduce a la evidente pregunta de cómo cuantificar el mal [90].

Esta clase de dilemas encuentran su mejor explicación en el famoso “dilema del tranvía”. Hay un tranvía en movimiento camino de una bifurcación. Sobre los dos tramos de vía que se abren ante él unos malhechores han situado rehenes atados sin posibilidad de liberarse. Sobre cada tramo hay una persona La pregunta es cuál de los dos caminos debería escoger el tranvía. Suponiendo que ha de ser una IA quien tome la decisión, habrá que decidir los criterios con que lo hará. La elección de estos no es sencilla pues el tiempo limitado y querer asignar más valor a una persona que a la otra es emplear un criterio subjetivo que puede resultar injusto a los ojos de un tercero. Pudiera ocurrir igualmente que exista una tercera vía que la IA no es capaz de detectar que salve las vidas de ambos. O incluso que esta tercera vida pueda no salvar ninguna de las dos vidas. Se hace evidente por tanto que casi cualquier decisión podría tener un contraargumento lógico [91].

En conclusión, el despliegue de una burbuja táctica sí puede acarrear problemas éticos como los propuestos u otros que aún no hayan sido planteados por nadie. Es por ello por lo que tanto las investigaciones y desarrollos ya realizados como los futuros han de tener en cuenta las posibles implicaciones que puedan tener para el cumplimiento de la misión encomendada a los distintos Cuerpos y Fuerzas de seguridad de un estado.

ANEXO III: CÓDIGO DE LA SIMULACIÓN DESARROLLADA

```
#include "ns3/applications-module.h"
#include "ns3/config-store.h"
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/log.h"
#include "ns3/mobility-module.h"
#include "ns3/network-module.h"
#include "ns3/nr-helper.h"
#include "ns3/nr-mac-scheduler-tdma-rr.h"
#include "ns3/nr-module.h"
#include "ns3/nr-point-to-point-epc-helper.h"
#include "ns3/point-to-point-helper.h"
#include <ns3/antenna-module.h>
#include <ns3/buildings-helper.h>
#include <cmath>
#include "math.h"
#include "ns3/netanim-module.h"
#include "ns3/flow-monitor.h"
#include "ns3/flow-monitor-helper.h"
#include "ns3/node-list.h"
#include "ns3/ipv4-flow-classifier.h"
#include "iostream"
#include <Eigen/Dense>


using namespace ns3;

using namespace Eigen;


void PacketTrace (Ptr<const Packet> p)
{
    std::cout << "Paquete recibido en el eNB a tiempo: " << Simulator::Now
().GetSeconds () << std::endl;}

struct Point {
    double x, y, z;
};

Point trilaterate(const std::vector<Point>& positions, const std::vector<double>&
distances) {
    if (positions.size() < 3 || positions.size() != distances.size()) {
        std::cerr << "Error: Número incorrecto de puntos o distancias." << std::endl;
        return {0, 0, 0};
    }
```

```
    int n = positions.size(); // Número de puntos de referencia
    MatrixXd A(n - 1, 3);
    VectorXd b(n - 1);

    // Construye la matriz de ecuaciones lineales
    for (int i = 1; i < n; ++i) {
        A.row(i - 1) << 2 * (positions[i].x - positions[0].x),
                        2 * (positions[i].y - positions[0].y),
                        2 * (positions[i].z - positions[0].z);
        b(i - 1) = std::pow(distances[0], 2) - std::pow(distances[i], 2) +
                   std::pow(positions[i].x, 2) - std::pow(positions[0].x, 2) +
                   std::pow(positions[i].y, 2) - std::pow(positions[0].y, 2) +
                   std::pow(positions[i].z, 2) - std::pow(positions[0].z, 2);
    }

    // Resuelve el sistema de ecuaciones lineales utilizando mínimos cuadrados
    Vector3d solution = A.jacobiSvd(ComputeThinU | ComputeThinV).solve(b);

    // Calcula las coordenadas del dispositivo
    Point devicePosition;
    devicePosition.x = solution(0);
    devicePosition.y = solution(1);
    devicePosition.z = solution(2);

    return devicePosition;
}




Int //comienzo función principal
main(int argc, char* argv[])
{

    //Bloque de configuración inicial del escenario
    std::string scenario = "UMa"; // escenario general de la simulación a elegir
entre
                                  //'RMa', 'UMa', 'UMi-StreetCanyon', 'InH-
OfficeMixed' y 'InH-OfficeOpen'
    double frequency = 28e9;      // frecuencia central
    double bandwidth = 100e6;     // ancho de banda
    double mobility = false;      // mobilidad habilitada si/no
    double simTime = 1.5;           // tiempo de simulación en segundos. Este ha de
tener la sufieciente duración para
                                    // albergar todos los eventos que queramos
simular
    double speed = 1;             // velocidad en m/s para un UT caminando
```

```
  bool logging = true; // habilitación de los servicios de análisis de
                       // resultados básicos
  double hBS;          // altura de la estación base en metros
  double hUT;          // altura de la antena de usuario en metros
  double txPower = 50; // Potencia de tx
  double t1,t2,t3;
  std::string simTag = "default";
  std::string outputDir = "./";

 std::vector<Point> positions = {{0, 0, 0}, {500, 0, 0}, {0, 500, 0}};

  //Configurado de las opciones seleccionadas en la simulación
  enum BandwidthPartInfo::Scenario scenarioEnum = BandwidthPartInfo::UMa;
  Config::SetDefault("ns3::LteRlcUm::MaxTxBufferSize", UintegerValue(999999999));

  for(int iter=1;iter<=10;iter++){
  for(int cont=0;cont<3;cont++){
  // enable logging



  // configuración de la altura de las antenas acorde al escenario elegido
  if (scenario == "RMa")
  {
      hBS = 40;
      hUT = 1.5;
      scenarioEnum = BandwidthPartInfo::RMa;
  }
  else if (scenario == "UMa")
  {
      hBS = 25;
      hUT = 1.5;
      scenarioEnum = BandwidthPartInfo::UMa;
  }
  else if (scenario == "UMi-StreetCanyon")
  {
      hBS = 10;
      hUT = 1.5;
      scenarioEnum = BandwidthPartInfo::UMi_StreetCanyon;
  }
  else if (scenario == "InH-OfficeMixed")
  {
      hBS = 3;
      hUT = 1;
      scenarioEnum = BandwidthPartInfo::InH_OfficeMixed;
  }
  else if (scenario == "InH-OfficeOpen")
  {
      hBS = 3;
```

```
        hUT = 1;
        scenarioEnum = BandwidthPartInfo::InH_OfficeOpen;
    }
    else
    {
        NS_ABORT_MSG("Scenario not supported. Choose among 'RMa', 'UMa', 'UMi-
StreetCanyon', "
                     "'InH-OfficeMixed', and 'InH-OfficeOpen'.");
    }

    // Creación de las estaciones base y de los terminales de usuario
    NodeContainer enbNodes;
    NodeContainer ueNodes;
    enbNodes.Create(3);
    ueNodes.Create(1);

    // posición de las estaciones base


    ns3::Vector senB1= {0.0, 0.0, hBS};
    ns3::Vector senB2= {0.0, 500.0, hBS};
    ns3::Vector senB3= {500.0,0.0, hBS};

    Ptr<ListPositionAllocator> enbPositionAlloc =
CreateObject<ListPositionAllocator>();
    enbPositionAlloc->Add(senB1);
    enbPositionAlloc->Add(senB2);
    enbPositionAlloc->Add(senB3);
    MobilityHelper enbmobility;
    enbmobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
    enbmobility.SetPositionAllocator(enbPositionAlloc);
    enbmobility.Install(enbNodes);

    // posición de los terminales de usuario y habilitación de la movilidad
    MobilityHelper uemobility;
    uemobility.SetMobilityModel("ns3::ConstantVelocityMobilityModel");
    uemobility.Install(ueNodes);

    if (mobility)
    {
        ueNodes.Get(0)->GetObject<MobilityModel>()->SetPosition({0, 15, hUT}); // (x,
y, z) en m
        ueNodes.Get(0)->GetObject<ConstantVelocityMobilityModel>()->SetVelocity(
            {speed, 0, 0}); // En este caso el dispositivo se movera a lo largo de x
con velocidad constante


    }
    else
```

```
    {
        ueNodes.Get(0)->GetObject<MobilityModel>()->SetPosition({20, 200, hUT});
        ueNodes.Get(0)->GetObject<ConstantVelocityMobilityModel>()->SetVelocity({0,
0, 0});

    }


     /* Creación de los simulation helpers de NR*/
    Ptr<NrPointToPointEpcHelper> epcHelper = CreateObject<NrPointToPointEpcHelper>();
    Ptr<IdealBeamformingHelper> idealBeamformingHelper =
CreateObject<IdealBeamformingHelper>();
    Ptr<NrHelper> nrHelper = CreateObject<NrHelper>();
    nrHelper->SetBeamformingHelper(idealBeamformingHelper);
    nrHelper->SetEpcHelper(epcHelper);

    /*
     * Configuración de espectro. Se crea una operational band única y se configura
el escenario.
     */
   BandwidthPartInfoPtrVector allBwps;
    CcBwpCreator ccBwpCreator;
    const uint8_t numCcPerBand = 1; // en este caso se tiene una única banda, y esa
banda
                                    // esta compuesta de una única portadora

    CcBwpCreator::SimpleOperationBandConf bandConf(frequency,
                                                   bandwidth,
                                                   numCcPerBand,
                                                   scenarioEnum);
    OperationBandInfo band = ccBwpCreator.CreateOperationBandContiguousCc(bandConf);
    // Inicializa para la banda los módelos de canal y de propagación elegidos para
la simulación
    nrHelper->InitializeOperationBand(&band);
    allBwps = CcBwpCreator::GetAllBwps({band});

    // Configuración del método ideal de beamforming
    idealBeamformingHelper->SetAttribute("BeamformingMethod",
                                         TypeIdValue(DirectPathBeamforming::GetTypeId
()));

    // Configuración del scheduler
    nrHelper->SetSchedulerTypeId(NrMacSchedulerTdmaRR::GetTypeId());

    // Configuración de las antenas de los UE's
    nrHelper->SetUeAntennaAttribute("NumRows", UintegerValue(2));
    nrHelper->SetUeAntennaAttribute("NumColumns", UintegerValue(4));
    nrHelper->SetUeAntennaAttribute("AntennaElement",
```

```
                                    PointerValue(CreateObject<IsotropicAntennaModel>(
()));

    // Configuración de las antenas para los gNBs
    nrHelper->SetGnbAntennaAttribute("NumRows", UintegerValue(8));
    nrHelper->SetGnbAntennaAttribute("NumColumns", UintegerValue(8));
    nrHelper->SetGnbAntennaAttribute("AntennaElement",
                                     PointerValue(CreateObject<IsotropicAntennaModel>
()));

    // instalación de los dispositivos de red NR
    NetDeviceContainer enbNetDev = nrHelper->InstallGnbDevice(enbNodes, allBwps);
    NetDeviceContainer ueNetDev = nrHelper->InstallUeDevice(ueNodes, allBwps);

    int64_t randomStream = 1;
    randomStream += nrHelper->AssignStreams(enbNetDev, randomStream);
    randomStream += nrHelper->AssignStreams(ueNetDev, randomStream);

    nrHelper->GetGnbPhy(enbNetDev.Get(0), 0)->SetTxPower(txPower);
    nrHelper->GetGnbPhy(enbNetDev.Get(1), 0)->SetTxPower(txPower);
    nrHelper->GetGnbPhy(enbNetDev.Get(2), 0)->SetTxPower(txPower);

    // Después de realizar toda la configuración, se ha de llamar explicitamente a la
función UpdateConfig ()
    for (auto it = enbNetDev.Begin(); it != enbNetDev.End(); ++it)
    {
        DynamicCast<NrGnbNetDevice>(*it)->UpdateConfig();
    }

    for (auto it = ueNetDev.Begin(); it != ueNetDev.End(); ++it)
    {
        DynamicCast<NrUeNetDevice>(*it)->UpdateConfig();
    }


  // Configuración las coordenadas de las esquinas del edificio
  Vector edificio1a (250.0, 250.0, 0.0);
  Vector edificio1b (250.0, 300.0, 0.0);
  Vector edificio1c (250.0, 500.0, 0.0);
  Vector edificio1d (300.0, 500.0, 0.0);

  // Creamos el objeto de edificio
  Ptr<Building> building = CreateObject<Building> ();
  building->SetBoundaries (edificio1a, edificio1b, edificio1c, edificio1d);
  building->SetBuildingType (Building::Residential);
  building->SetExtWallsType(Building::ConcreteWithWindows);
  building->SetNFloors(3);
  building->SetNRoomsX(3);
  building->SetNRoomsY(2);
```

```
  // Configuramos el escenario de la simulación
  Ptr<BuildingContainer> edificios = CreateObject<BuildingContainer> ();
  edificios->AddBuilding (building);
  Ptr<OutdoorMesh> mesh = CreateObject<OutdoorMesh> ();
  mesh->SetBuildingContainer (edificios);
  mesh->Install (node);



    // Creación de la conexión al núcleo e instalación de los protocolos IP en los
UEs
    // obtención SGW/PGW y creación de un único RemoteHost
    Ptr<Node> pgw = epcHelper->GetPgwNode();
    NodeContainer remoteHostContainer;
    remoteHostContainer.Create(1);
    Ptr<Node> remoteHost = remoteHostContainer.Get(0);
    InternetStackHelper internet;
    internet.Install(remoteHostContainer);

    // Conexión remoteHost a pgw. Configuración del enrutamiento
    PointToPointHelper p2ph;
    p2ph.SetDeviceAttribute("DataRate", DataRateValue(DataRate("100Gb/s")));
    p2ph.SetDeviceAttribute("Mtu", UintegerValue(2500));
    p2ph.SetChannelAttribute("Delay", TimeValue(Seconds(0.000)));
    NetDeviceContainer internetDevices = p2ph.Install(pgw, remoteHost);

    Ipv4AddressHelper ipv4h;
    ipv4h.SetBase("1.0.0.0", "255.0.0.0");
    Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign(internetDevices);
    Ipv4StaticRoutingHelper ipv4RoutingHelper;

    Ptr<Ipv4StaticRouting> remoteHostStaticRouting =
    ipv4RoutingHelper.GetStaticRouting(remoteHost->GetObject<Ipv4>());
    remoteHostStaticRouting->AddNetworkRouteTo(Ipv4Address("7.0.0.0"),
Ipv4Mask("255.0.0.0"), 1);
    internet.Install(ueNodes);

    Ipv4InterfaceContainer ueIpIface;
    ueIpIface = epcHelper->AssignUeIpv4Address(NetDeviceContainer(ueNetDev));


    // asignación de las direcciones IP a las UEs y instalaciobn de las aplicaciones
de UDP downlink
    uint16_t dlPort = 1234;
    ApplicationContainer clientApps;
    ApplicationContainer serverApps;
    for (uint32_t u = 0; u < ueNodes.GetN(); ++u)
    {
```

```
        Ptr<Node> ueNode = ueNodes.Get(u);
        // Configuración de la gateway por defecto para el UE
        Ptr<Ipv4StaticRouting> ueStaticRouting =
        ipv4RoutingHelper.GetStaticRouting(ueNode->GetObject<Ipv4>());
        ueStaticRouting->SetDefaultRoute(epcHelper->GetUeDefaultGatewayAddress(), 1);

        UdpServerHelper dlPacketSinkHelper(dlPort);
        serverApps.Add(dlPacketSinkHelper.Install(ueNodes.Get(u)));

        UdpClientHelper dlClient(ueIpIface.GetAddress(u), dlPort);
        dlClient.SetAttribute("Interval", TimeValue(MicroSeconds(1)));
        dlClient.SetAttribute("MaxPackets", UintegerValue(1));
        dlClient.SetAttribute("PacketSize", UintegerValue(1500));
        clientApps.Add(dlClient.Install(remoteHost));

    nrHelper->AttachToEnb(ueNetDev.Get(0), enbNetDev.Get(cont));

    // Comienzo de las aplicaciones de servidor y usuario
    serverApps.Start(Seconds(0.2));
    clientApps.Start(Seconds(0.2));
    serverApps.Stop(Seconds(simTime));
    clientApps.Stop(Seconds(simTime - 0.2));

    // habilitación de las trazas de seguimiento propias del módulo NR
    nrHelper->EnableTraces();

     FlowMonitorHelper flowmonHelper;
    NodeContainer endpointNodes;
    endpointNodes.Add(remoteHost);
    endpointNodes.Add(ueNodes);

    Ptr<ns3::FlowMonitor> monitor = flowmonHelper.Install(endpointNodes);
    monitor->SetAttribute("DelayBinWidth", DoubleValue(0.001));
    monitor->SetAttribute("JitterBinWidth", DoubleValue(0.001));
    monitor->SetAttribute("PacketSizeBinWidth", DoubleValue(20));

    Simulator::Stop(Seconds(simTime));
    Simulator::Run();

    //
    monitor->CheckForLostPackets();
    Ptr<Ipv4FlowClassifier> classifier =
        DynamicCast<Ipv4FlowClassifier>(flowmonHelper.GetClassifier());
    FlowMonitor::FlowStatsContainer stats = monitor->GetFlowStats();

    double averageFlowThroughput = 0.0;
    double averageFlowDelay = 0.0;

    std::ofstream outFile;
```

```
    std::string filename = outputDir + "/" + simTag;
    outFile.open(filename.c_str(), std::ofstream::out | std::ofstream::trunc);
    if (!outFile.is_open())
    {
        std::cerr << "Can't open file " << filename << std::endl;
        return 1;
    }

    outFile.setf(std::ios_base::fixed);

    double flowDuration = (simTime - 0.2);
    for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator i = stats.begin();
         i != stats.end();
         ++i)
    {
        Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow(i->first);
        std::stringstream protoStream;
        protoStream << (uint16_t)t.protocol;
        if (t.protocol == 6)
        {
            protoStream.str("TCP");
        }
        if (t.protocol == 17)
        {
            protoStream.str("UDP");
        }
        outFile << "Flow " << i->first << " (" << t.sourceAddress << ":" <<
t.sourcePort << " -> "
                << t.destinationAddress << ":" << t.destinationPort << ") proto "
                << protoStream.str() << "\n";
        outFile << "  Tx Packets: " << i->second.txPackets << "\n";
        outFile << "  Tx Bytes:   " << i->second.txBytes << "\n";
        outFile << "  TxOffered:  " << i->second.txBytes * flowDuration / 1000.0;
                << " Mbps\n";
        outFile << "  Rx Bytes:   " << i->second.rxBytes << "\n";
        if (i->second.rxPackets > 0)
        {
            // Measure the duration of the flow from receiver's perspective
            averageFlowThroughput += i->second.rxBytes * 8.0 / flowDuration / 1000 /
1000;
            averageFlowDelay += 1000 * i->second.delaySum.GetSeconds() / i-
>second.rxPackets;

            outFile << "  Throughput: " << i->second.rxBytes * 8.0 / flowDuration /
1000 / 1000
                    << " Mbps\n";
            outFile << "  Mean delay:  "
                    << i->second.delaySum.GetNanoSeconds() << " ms\n";
                    if(cont==0){
```

```
                        t1=i->second.delaySum.GetSeconds() / i->second.rxPackets;
                    }
                    if(cont==1){
                        t2= i->second.delaySum.GetSeconds() / i->second.rxPackets;
                    }
                    if(cont==2){
                        t3= i->second.delaySum.GetSeconds() / i->second.rxPackets;
                    }
            outFile << "  Mean jitter:  "
                    <<i->second.jitterSum.GetNanoSeconds()<< " ms\n";
        }
        else
        {
            outFile << "  Throughput:  0 Mbps\n";
            outFile << "  Mean delay:  0 ms\n";
            outFile << "  Mean jitter: 0 ms\n";
        }
        outFile << "  Rx Packets: " << i->second.rxPackets << "\n";
    }

    outFile << "\n\n  Mean flow throughput: " << averageFlowThroughput / stats.size()
<< "\n";
    outFile << "  Mean flow delay: " << averageFlowDelay / stats.size() << "\n";

    outFile.close();

    std::ifstream f(filename.c_str());

    if (f.is_open())
    {
        std::cout << f.rdbuf();
    }




    Simulator::Destroy();
    }

    }std::vector<double> distances = {t1*299792458, t2*299792458, t3*299792458};

    // Calcula la posición del dispositivo utilizando trilateración
    Point devicePosition = trilaterate(positions, distances);

    // Muestra la posición del dispositivo
    std::cout << "Posición del dispositivo: (" << devicePosition.x << ", "
              << devicePosition.y << ", " << devicePosition.z << ")" << std::endl;
 }
    return 0;}
```