Universitat Oberta de Catalunya

uoc.edu

Distribución de la energía en el entorno de una Smart City basado en BlockChain

José Andrés Collada Nevado
Máster Universitario en Ingeniería de Telecomunicación
Smart City

Víctor Monzón Baeza
Carlos Monzo

06 / 2020

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Distribución de energía en el entorno de una Smart City basado en blockchain.*
Nombre del autor:	*José Andrés Collada Nevado*
Nombre del consultor/a:	*Víctor Monzón Baeza*
Nombre del PRA:	*Carlos Monzo*
Fecha de entrega (mm/aaaa):	06 / 2020
Titulación:	*Máster Universitario en Ingeniería de Telecomunicación*
Área del Trabajo Final:	*Smart City*
Idioma del trabajo:	*Castellano*
Palabras clave	*BlockChain, Smart City, IoE*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

Para afrontar el desafío del incremento de población en las ciudades y hacer frente al cambio climático actual es fundamental introducir la tecnología en todas las áreas con el objetivo de crear ciudades inteligentes.

Este trabajo tiene como objetivo analizar las alternativas disponibles hoy en día y desarrollar un modelo para ayudar a implementar la transición al cambio partiendo del actual sistema de consumo de la energía.

Analizar la forma de realizar dicho tránsito entre el sistema analógico actual hacia un futuro digital mediante la incorporación de las nuevas tecnologías como el *BlockChain*, la *Smart City* y el *Internet of Energy*. Para su desarrollo es fundamental realizar una implementación realista del proceso de transformación y saber cómo llevarlo a cabo.

Los problemas a resolver son numerosos: mejorar la eficiencia, la seguridad, la privacidad, intercambio, el método de transmisión de los datos para monitorizar y enviar instrucciones, y determinar un sistema métrico.

En conclusión, se debe reformular y repensar como transformar y mejorar el sistema de consumo de energía gracias a la adopción de las nuevas tecnologías.

Abstract (in English, 250 words or less):

With the current challenge of rising populations in cities and the climate change crisis, it is necessary to phase in new technologies with the aim of creating smart cities.

The main goal is to analyse the current alternatives and to develop a model for the transition from the current energy consumption system.

It is necessary to analyse the transition between the current analog system, towards a digital future that incorporates new technologies such as, BlockChain, Smart City and the Internet of Energy. In order to implement that transition, it is essential to have a clear structure for implementation of the transformation process.

There are various challenges to be solved: improving the electricity transmission network, helping to connect and monitor large or small energy projects, and ensuring electricity reaches homes and businesses reliably, safely, and efficiently.

In conclusion, it is important to reconsider the essential role of new technologies in transforming and improving the consumption system.

Índice

Índice iii
Lista de tablas v
Lista de figuras v
1. Introducción 1
1.1. Contexto y justificación del Trabajo 1
1.2. Objetivos del Trabajo 3
1.3. Enfoque y método seguido 4
1.4. Planificación del Trabajo 1
1.5. Breve descripción de los otros capítulos de la memoria 1
2. Estado del Arte 2
2.1. Smart City 2
2.2. Internet of Things 3
2.3. Sistema eléctrico de potencia 5
2.3.1. Breve historia 5
2.3.2. Sistema actual 6
2.3.3. Futuro 7
2.4. BlockChain 9
2.5. Conclusión 10
3. Análisis 11
3.1. Topologías Red 11
3.2. Dispositivos 13
3.2.1. SoC 15
3.2.2. SBC 17
3.2.3. Paneles solares, baterías y sensores 19
3.3. Blockchain 20
3.3.1. Cifrado 20
3.3.2. Algoritmo Consenso 21
PoW: Proof of work 22
PoS: Prof of stake 23
3.3.3. Smart contract 24
3.3.4. Tecnologías 26
Publica 26
Privada 26
3.4. Tecnologías LPWAN 29
3.4.1. LoraWan 29
3.4.2. SigFox 30
3.4.3. NB-IoT 30
3.5. Protocolos de comunicación 32
3.5.1. CoAP 33
3.5.2. MQTT 34
3.6. Cloud Computing 36
3.6.1. Azure 37
3.6.2. Google Cloud 37
3.6.3. AWS 38
4. Selección 39

4.1. Topología de red ... 40
4.2. Dispositivos físicos analizados ... 40
4.3. Dispositivos físicos no analizados ... 41
4.4. Comunicación ... 42
4.5. Blockchain Publica ... 42
4.5.1. Cifrado ... 43
4.5.2. Algoritmo de consenso ... 43
4.5.3. Ethereum, Smart Contract y Tokens ... 43
4.6. Cloud Computing ... 46
5. Modelo ... 49
5.1. Funcionamiento: sensores ... 50
5.1.1. Baterías ... 51
5.1.2. Placas solares ... 52
5.1.3. Usuario ... 53
5.1.4. Actuador ... 53
5.2. Funcionamiento: cloud ... 54
5.2.1. AWS IoT ... 55
5.2.2. Amazon SageMaker ... 58
5.2.3. Amazon Managed Blockchain ... 60
5.3. Funcionamiento: smart contract ... 61
6. Conclusiones ... 63
7. Glosario ... 65
8. Bibliografía ... 68
9. Anexos ... 71
9.1. Smart Contract ... 71

Lista de tablas

Tabla 1. Comparativa de los SoC 16
Tabla 2. Comparativa de los SBC 18
Tabla 3. Comparativa entre PoW y PoS 24
Tabla 4. Comparativa de tecnologias de BC 28
Tabla 5. Comparativa de las tecnologias LPWAN. 31
Tabla 6. Comparativa de los protocolos de comunicacion 36

Lista de figuras

Figura 1. Central del Niágara. 5
Figura 2. Sistema SEP 6
Figura 3. Evolución del Smart Grid 7
Figura 4. Evolucion del SEP 8
Figura 5. Ejemplo de cadena de bloques (BC) 9
Figura 6. Topologías de Red 11
Figura 7. Red de router con repetidores y Red Mallada 12
Figura 8. Red base 13
Figura 9. Sensores en la red. 14
Figura 10. Red sobre una topologia mesh y sobre una topologia arbol 14
Figura 11. Placa Arduino y entorno 15
Figura 12. Entorno de un SBC conectado a la nube 18
Figura 13. Comparativa entre PoW y PoS 21
Figura 14. Red DAO 25
Figura 15. Cuadro comparativo entre tecnologia de conectividad 29
Figura 16. Entorno de una red tipo SigFox 30
Figura 17. Diagrama de las caracteristicas de SigFox, LoRa y NB.IoT. 31
Figura 18. Sistema OSI de los protocolos CoAP y MQTT 32
Figura 19. Mecanismo del protocolo CoAP 33
Figura 20. Mecanismo del protocolo MQTT 34
Figura 21. Ejemplo de estructura en arbol de una red MQTT 35
Figura 22. Esquema del funcionamiento de la AMB de Amazon 38
Figura 23. Estructura general de la solucion del proyecto 39
Figura 24. Topologia en arbol de la red del usuario 40
Figura 25. Funcionamiento del sensor de la red 41
Figura 26. Ejemplo de AWS IoT Core 46
Figura 27. Ejemplo de AWS IoT Device Management 46
Figura 28. Ejemplo de AWS IoT Events 47
Figura 29. Ejemplo del módulo Amazon SageMaker 47
Figura 30. Ejemplo de Amazon Managed Blockchain 48
Figura 31. Red del proyecto 49
Figura 32. Distribución de los sensores en la red 50
Figura 33. Detalle del sensor de corriente 50
Figura 34. Modelo de decisión del funcionamiento de las baterías 51
Figura 35. Modelo de decisión del funcionamiento de las placas solares 52
Figura 36 Modelo de decisión del funcionamiento del usuario final 53
Figura 37. Relación de la nube con Ethereum, web REE y usuario final 54

Figura 38. Modulo AWS IoT 55
Figura 39. Implementacion del protocolo MQTT 56
Figura 40. Modulo AWS SageMaker 58
Figura 41 Modelo de decisión: funcionamiento del usuario final 59
Figura 42. Modelo de decisión del funcionamiento de las baterías 59
Figura 43. Modulo AWS Blockchain 60
Figura 44. Definición del Token 61
Figura 45. Compra de electricidad 61
Figura 46. Venta de electricidad 61
Figura 47. Regalar electricidad 62
Figura 48. Consultar el balance de tokens 62

1. Introducción

1.1. Contexto y justificación del Trabajo

El desarrollo de las ciudades siempre ha estado supeditado a la evolución de las tecnologías y estas han podido desarrollarse gracias a la energía eléctrica. Según las Naciones Unidas en las próximas décadas más del 70% de la población, es decir, más de cinco billones de personas vivirán en ellas.

El cambio climático al que nos enfrentamos no es un futurible sino una realidad. El actual modo de distribución de la energía, que no ha tenido apenas modificaciones desde su introducción a finales del siglo XIX, es uno de los principales focos de polución que crea el efecto invernadero, enfermedades respiratorias y contamina el agua en su producción, en su distribución y en el consumo final.

Hay dos puntos clave para hacerle frente: generar **energía renovable** desde las empresas de electricidad y prosumidores para acelerar la sustitución de las energías fósiles, y **el uso de tecnología** para dimensionar apropiadamente las nuevas necesidades que están surgiendo debido al incremento de población, y para eficientar y optimizar su distribución ya que alrededor del 10% de la energía generada es perdida sin ser consumida.

En este trabajo nos centraremos en como introducir la tecnología digital en la vida cotidiana de la población para transformar, modernizar y así convertir las ciudades poco a poco en inteligentes.

Actualmente la informática y las tecnologías más novedosas como IoT, IoE, Smart Grid, Edge Computing, BlockChain, Big Data o Machine Learning están llamadas a revolucionar las ciudades dotando a las infraestructuras de inteligencia, es decir, crear un entorno conocido como Smart City.

El BlockChain es un tipo de red peer-to-peer que introduce inteligencia y seguridad a las transacciones por internet de una manera abierta, descentralizada, consistente, certificada y transparente. Esta tecnología ya se está empleando en el entorno de Smart City en áreas como la salud, el transporte, la de educación, o en seguridad ciudadana, donde ha proporcionado ya buenos resultados. En cambio, en el sector de la energía su introducción aún no es madura por lo que proporciona un campo interesante de investigación para conseguir mejorar la eficiencia de los sistemas de distribución de la energía.

Por otro lado, el concepto de Internet of Energy (IoE), un subgrupo del Internet of Things (IoT) aplicado a la energía, se basa en introducir una serie de sensores y controladores a lo largo de la red eléctrica para conectarla entre sí y dotarla de inteligencia, monitorizando su

comportamiento en tiempo real y enviando instrucciones para automatizarlo. Esto es una parte fundamental para crear el Smart Grid por el que mediante la informática se gestione de manera eficiente la producción y la distribución de la energía eléctrica.

Los inconvenientes de los sistemas de energía actuales impiden que las ciudades evolucionen y obtengan las ventajas que proporcionan las nuevas tecnologías basadas en Internet y servicios web. Esto motiva el estudio, análisis y desarrollo de este trabajo. Existe por ello la necesidad de migrar los actuales sistemas de distribución y consumo de energía eléctrica hacia sistemas inteligentes, autónomos y eficientes.

En el contexto de las tecnologías existentes, este trabajo se centra especialmente en el uso conjunto de las tecnologías de Smart City y Blockchain. Este tipo de solución a parte de descentralizar la distribución pretende resolver numerosos e importantes problemas como certificar la autenticidad de la identidad, asegurar la privacidad, facilitar la negociación y la interconexión entre prosumidores y empresas eléctricas con los consumidores finales mejorando la eficiencia, distribución, almacenaje y flexibilidad del sistema de energía.

1.2. Objetivos del Trabajo

Basado en el contexto y motivación descritas en la sección anterior, los objetivos extraídos y planteados para este proyecto se agrupan en 3 bloques: análisis, desarrollo y modelo de la solución a la problemática de los actuales sistemas de energía en usuario final. Estos objetivos se describen a continuación:

- Análisis las posibles tecnologías que permitan dotar al usuario final de medios digitales.

- Resolver problemas de escalabilidad, consistencia, privacidad, seguridad y autoridad a la hora de generar, almacenar, distribuir e intercambiar energía.

- Crear el entorno para transformar el actual sistema analógico e introducirlo en el mundo digital basado en el entorno de la Smart City donde la electricidad se cree y almacene cerca donde se consume.

- Conseguir un sistema fiable y seguro para intercambiar la energía por *tokens* tanto para la compra como en la venta.

- Construir una aplicación sobre tecnologías disruptivas como el blockchain, machine learning y el edge computing.

- Tratamiento de datos de forma abierta pero segura utilizando herramientas alojadas en internet.

- Monitorización: para trasladar los datos monitorizados utilizaremos sensores y una red distribuida de bajo consumo.

- Sistema de distribución: centralizado, distribuido o jerárquico.

- Entre todas las distintas tecnologías que ofrece el mercado desarrollar una plataforma de bajo coste para tratar de *tokenizar* la energía generada por unas placas solares de un entorno urbano.

1.3. Enfoque y método seguido

El enfoque que se planteará en este trabajo consistirá en un estudio teórico sobre como implantar las tecnologías disruptivas en el actual sistema del usuario final y así contribuir en convertir las ciudades en Smart City.

Basado en el contexto, la motivación y los objetivos descritos anteriormente, el método planteados para este proyecto se agrupan en 3 bloques: análisis, viabilidad y modelo de implantación. Este método se describe a continuación:

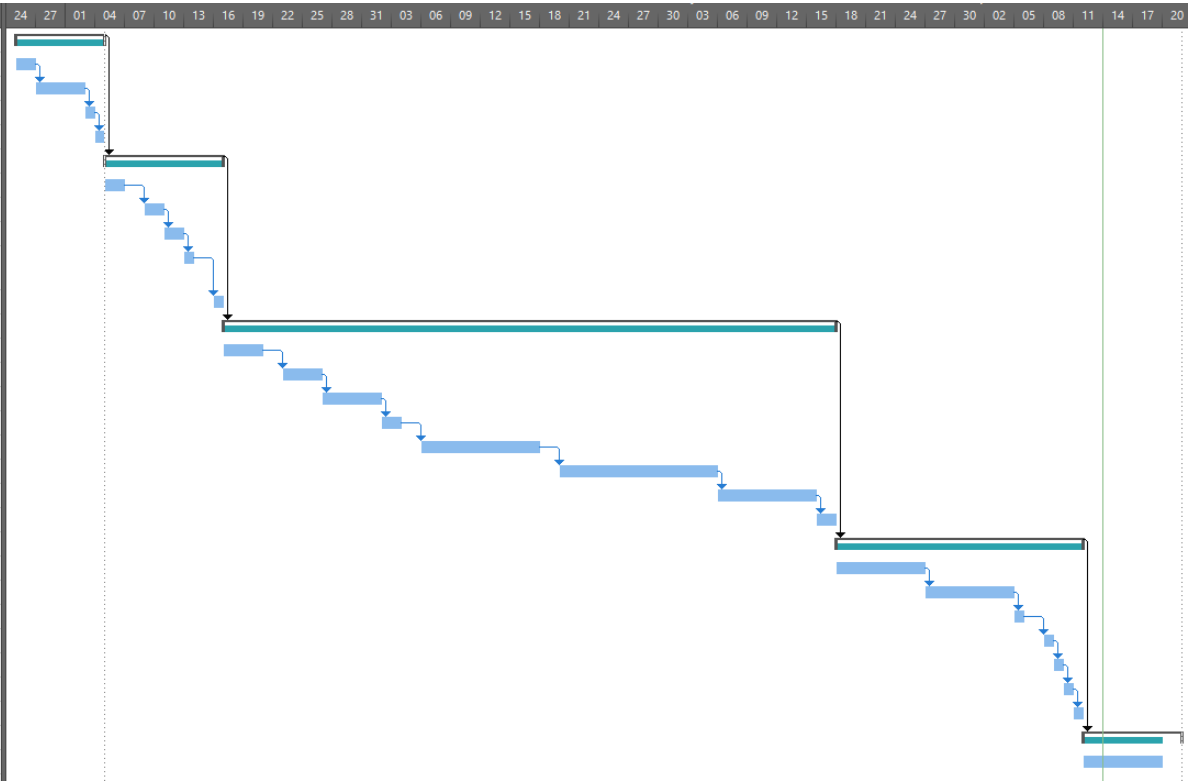
- Análisis: actualmente las ciudades necesitan evolucionar junto a la tecnología para convertirse en Smart Cities para hacer frente a los nuevos desafíos de futuro como aumento de la población, combatir al cambio climático haciendo eficiente las ciudades e introducir nuevos modelos de gestionar las ciudades.

- Viabilidad. La tecnología BC se puede integrar al entorno Smart City, en concreto en el sistema del usuario final de consumo de la energía eléctrica para hacerla más eficiente, segura y flexible.

- Estudio de implantación. El despliegue debe ser planificado donde primero se implantará sensores y actuadores, después se creará un sistema de comunicación para enviar los datos monitorizados a la nube; y una vez allí tratarlos bajo un modelo de machine learning.

1.4. Planificación del Trabajo

Nombre de tarea	Duración	Comienzo	Fin
PEC 1 - Titulo, objetivos, resumen	**7 días**	**mar 25/02/20**	**mié 04/03/20**
Motivación	2 días	mar 25/02/20	mié 26/02/20
Motivación, Objetivos, Enfoque, Metodo	3 días	jue 27/02/20	lun 02/03/20
Planificación, Resumen e Indice	1 día	mar 03/03/20	mar 03/03/20
Revisar y modificar el plan estratégico	1 día	mié 04/03/20	mié 04/03/20
PEC 2 - Estudio del Arte	**8 días**	**jue 05/03/20**	**lun 16/03/20**
Buscar Referencias Smart Cities	2 días	jue 05/03/20	vie 06/03/20
Buscar Referencias Block Chain	2 días	lun 09/03/20	mar 10/03/20
Buscar Referencias Distribucion de Energia Electrica	2 días	mié 11/03/20	jue 12/03/20
Relacionar Block Chain y la Distribucion de la Energia Electrica	1 día	vie 13/03/20	vie 13/03/20
Como implantar BlockChain como modelo	1 día	lun 16/03/20	lun 16/03/20
PEC 3 - Propuesta	**46 días**	**mar 17/03/20**	**dom 17/05/20**
Introducción	4 días	mar 17/03/20	vie 20/03/20
Descripcion Block Chain	4 días	lun 23/03/20	jue 26/03/20
Descripcion Smart City	4 días	vie 27/03/20	mié 01/04/20
Descripcion Internet of Energy	2 días	jue 02/04/20	vie 03/04/20
Analisis de la Propuesta	10 días	lun 06/04/20	vie 17/04/20
Implantacion del Modelo	12 días	lun 20/04/20	mar 05/05/20
Resultado del Estudio	8 días	mié 06/05/20	vie 15/05/20
Repaso	2 días	sáb 16/05/20	dom 17/05/20
PEC 4 - Memoria	**19 días**	**lun 18/05/20**	**jue 11/06/20**
Introducción	7 días	lun 18/05/20	mar 26/05/20
Conclusión	7 días	mié 27/05/20	jue 04/06/20
Abstract	1 día	vie 05/06/20	vie 05/06/20
Definiciones	1 día	lun 08/06/20	lun 08/06/20
Anexos	1 día	mar 09/06/20	mar 09/06/20
Referencias	1 día	mié 10/06/20	mié 10/06/20
Glosario	1 día	jue 11/06/20	jue 11/06/20
PEC 5 - Presentación PFM	**7 días**	**vie 12/06/20**	**dom 21/06/20**
Crear la Presentacion en PPT	6 días	vie 12/06/20	vie 19/06/20



1.5. Breve descripción de los otros capítulos de la memoria

Los siguientes capítulos se desgranan en tres apartados:

- Análisis: Identificar entre todas las tecnologías existentes y para cada parte del proyecto que nos ocupa, para realizar una comparativa entre las alternativas más potentes.

- Selección: Partiendo del análisis del apartado anterior escogemos tecnologías para llevar a cabo el proyecto.

- Modelo: Desarrollar como se va a distribuir las tecnologías seleccionadas y crear la lógica necesaria para que el sistema de sensores, actuadores y plataformas en la nube se comporte como un sistema de creación de energía y posterior utilización mediante *tokens.*

2. Estado del Arte

2.1. Smart City

Actualmente las ciudades se encuentran en un proceso de desarrollo enmarcado por la tecnología. Esta evolución pretende mejorar las ciudades actuales y el nivel de vida de los ciudadanos gracias al uso de las tecnologías de la información y comunicaciones (TIC).
En [1] [2] se pueden encontrar las bases fundamentales para convertir una ciudad en Smart City, entre las que destacan:

- Dispositivos que generan datos.
- Plataforma donde se almacenan, procesan y analizan los datos.
- Aplicaciones de servicios.

Principalmente una Smart City (SC) se sitúa en el entorno urbano en el que se busca optimizar los recursos de los servicios públicos, hacerlos más accesibles y ahorrar costes de operación. Con todo esto, mejorar la calidad de vida de los ciudadanos.

Actualmente existen en desarrollo una gran variedad de tecnologías, protocolos y arquitecturas. Como ejemplo encontramos para Big Data [3], IoT [4], 5G [5], BlockChain [6], Cloud Computing [7] o Machine Learning [8]. Esta diversidad produce un hándicap en este sector: encontrar la correcta combinación entre tecnologías en cada área de la ciudad. La infraestructura debe ser simple, accesible al ciudadano y abierta a sinergias. La participación ciudadana es crucial para que dichos desarrollos sean útiles y aporten valor a la sociedad.

Hay numerosos sectores que entran en juego en una SC:

- El *Smart Governance* es la nueva forma de dirigir una ciudad permitiendo a los ciudadanos que aporten ideas, información sobre eventos y participen en la toma de decisiones. Un ejemplo es el proyecto MOAD [9] desarrollado en Andalucía que se basa en una plataforma de tramitación digital que permite al ciudadano de forma telemática gestionar ciertos tramites con la Administración.

- El *Smart Mobility* surgió principalmente para gestionar el tráfico, el sistema de aparcamiento, reducir el impacto ambiental, mejorar la planificación de los medios de transporte y la movilidad de las personas.

  Una medida para su implantación es dotar de conectividad a los vehículos eléctricos [10] para conseguir sostenibilidad, disminuir la contaminación, generar la ruta óptima evitando atascos y reducción de costes respecto a los vehículos de combustible fósil “analógico”.

La Diputación de Gipuzkoa [11] ha lanzado una propuesta para impulsar el desarrollo de este concepto entre los sectores “de actividad relacionados con la electromovilidad, el almacenamiento de energía y el vehículo autónomo.” Estos adelantos cambiarán nuestra forma de desplazarnos en la ciudad.

- Las *Smart Utilities* se focalizan en abaratar costes de operación y distribución en empresas dedicadas al gas, al agua o a la electricidad. La convención de empresas eléctricas americanas Smart Utility Summit [12] cada año celebran una cumbre presentando los últimos desarrollos en este campo.

- La inteligencia de los edificios es el área que cubre el *Smart Buildings*. Eficientar el consumo, integrar un sistema de control y automatización de sistemas como la seguridad, la iluminación o la climatización. Un ejemplo es el edificio londinense [13] “The Crystal” donde todo el agua de los inodoros es reciclada, consiguen un 70% menos de emisión de CO2 que una oficina normal y el gasto de calefacción es nulo.

- Y por último el *Smart Environment* se centra en los ámbitos de energía, agua, residuos y el medio ambiente. La Universidad de Alicante [14] tiene implementado a nivel de campus este sistema. En este caso implementa una arquitectura centralizada, flexible y un *web service* adecuado que sirve como base a los servicios desplegados.

Los ejemplos anteriores abarcan una parte del concepto de SC donde los dispositivos inteligentes, el acceso a los datos, conectividad y las aplicaciones en la nube son fundamentales para que dichos servicios se lleven a cabo.

2.2. Internet of Things

La tecnología llamada “internet de las cosas” es una de las principales habilitantes de la SC y ha sido palanca para el desarrollo de nuevos modelos de negocio, denominado Industria 4.0.

Internet of Things (IoT) es una tecnología que en la última década ha sido desarrollada tanto en la parte hardware como en la parte software. Estos desarrollos han permitido mejorar la recepción de los datos de telemetría generados para su análisis y posterior ejecución de comandos e implementar distintas tareas. Estos dispositivos son capaces de interactuar tanto con el usuario final como con otros terminales inteligentes, *smart devices,* por lo que son válidos para su uso en cualquier ámbito. Además, al estar conectados a internet la toma de decisiones se realiza en tiempo real. Todo ello permite crear métodos y modelos eficientes de sostenibilidad a cuanto el uso energético y de recursos.

Una de las causas del gran desarrollo que ha tenido el IoT es la miniaturización de los componentes. Este hecho elimina varias restricciones: qué tipo de dispositivos es elegible, el consumo eléctrico y su ubicación. Por ello cualquier

dispositivo es susceptible de ser equipado de microprocesadores, sensores, actuadores, baterías y módulos de comunicación inalámbricas.

El futuro del IoT está asegurado ya que es capaz de adaptarse a nuevos cambios y añadir los últimos avances. Como por ejemplo [15] muestra como el Big Data y el Cloud Computing se añaden al IoT para optimizar el análisis del comercio electrónico e implementar nuevos negocios. En [16] se observa como gracias al Fog y el Edge Computing se resuelven problemas estructurales que aparecen cuando surge la necesidad de gestionar gran cantidad de dispositivos IoT. En el campo de la seguridad hay estudios donde las características del BlockChain se utilizan para resolver problemas de seguridad en la comunicación del ecosistema de IoE [17].

Dentro del hogar el IoT cubre otro tipo de escenarios, en los que es el propio usuario quien los gestiona y configura en remoto para que ejecuten instrucciones automáticamente. Ejemplos son el detector de movimiento para alarmas, el control de temperatura para el uso del aire acondicionado, el sensor de luz para la apertura de persianas si hace sol, encender la calefacción cuando se está llegando a casa o encargar la compra de leche en caso de que se acabe en la nevera.

Desde hace tiempo ya existen servicios web para controlar este tipo de escenarios, como por ejemplo IFTTT (If This, Then That). Esta plataforma [18] permite crear y programar acciones, llamadas applets, para automatizar usos y acciones con otros servicios webs o dispositivos IoT. Mas de 600 empresas como Philips, iRobot o Wyze han colaborado con IFTTT para ofrecer a sus clientes una mejor experiencia de sus productos.

Otro ejemplo parecido presenta [19], donde implementa una arquitectura para controlar dispositivos heterogéneos con el añadido de seguridad, como bloquear automáticamente un equipamiento en caso de incidencia.

En la empresa privada cada vez es mayor la penetración. El articulo [20] hace referencia a varios casos. Uno de ellos es Uber que mediante algoritmos, sensores y teléfonos inteligentes ha logrado conectar al conductor y al usuario en tiempo real y ajustar el coste a la demanda. El otro es el caso de Rolls Royce donde han implementado una red de sensores IoT para mejorar la calidad en la fabricación de sus turbinas que utilizan para desarrollar sus vehículos.

A nivel urbano los gobiernos locales son quienes se encargan de gestionar los proyectos. En el anterior apartado ya hemos repasado algunos ejemplos. Y uno de los principales retos que se enfrentan es el consumo de recursos, en particular el consumo eléctrico. El aumento de población y la multiplicación de los dispositivos IoT derivan en un gran incremento en el consumo de energía. Debido al encarecimiento progresivo del coste de la electricidad será clave para las administraciones públicas pactar con el sector privado una evolución en el modelo de gestión energético. Por ello es urgente introducir los avances tecnológicos en la SC como el IoT, el BC o el Machine Learning, para así conseguir un escenario sostenible con reducciones del gasto público y la eficiencia de los recursos como veremos a continuación.

2.3. Sistema eléctrico de potencia

El sistema eléctrico de potencia desde su origen en el siglo XIX y durante el siglo XX ha sido desarrollado de manera analógica. **La evolución en el siglo XXI** de cualquier sector pasa por introducir las tecnologías TIC en sus sistemas.

Una parte fundamental en el desarrollo de la ciudad inteligente del mañana es el motor que lo mueve: la electricidad. Y es por ello que **es clave agilizar su incorporación a lo largo del sistema eléctrico, desde su generación hasta su consumo.**

2.3.1. Breve historia

En el año 1882 la *Pearl Street Central* fundada por Thomas Edison implementó el primer sistema eléctrico de potencia (SEP) basado en la corriente continua (DC) [21]. Esta elección de distribuir la energía contaba con numerosas limitaciones: alto coste de transporte, grandes pérdidas por disipación de calor, poca distancia entre la planta generadora y el consumidor por la pérdida de voltaje o la incapacidad de cambiar los niveles de voltajes en todo ese recorrido. Nikola Tesla, contemporáneo y antiguo empleado de dicha compañía, ingresó en la *Westinghouse Electric Corporation* desatando la conocida "Guerra de las Corrientes" [22] ya que desarrolló la corriente alterna (AC) para emplearla en el SEP. La AC solventaba parte de dichos problemas: permite el transporte a largas distancias, cambiar el voltaje mediante transformadores y funcionar en ambos sentidos. Finalmente, en el año 1896 terminó la mentada guerra debido a la adjudicación de AC para crear la gran Central del Niágara (Figura1).

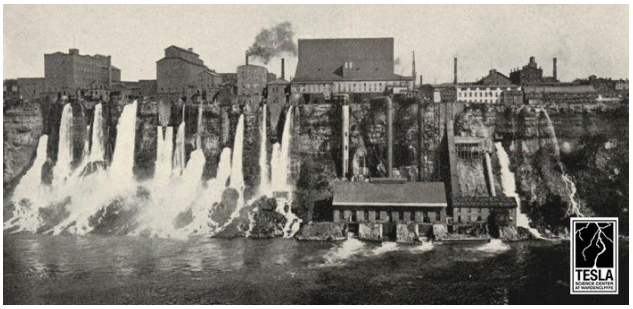


Figura 1. Central del Niágara.

A principios del siglo pasado la evolución fue basada en la construcción de las centrales y las líneas de transporte de AC. La introducción del sistema trifásico europeo, que triunfó frente al bifásico americano, permitió aumentar la longitud de la transmisión y la inclusión del "aislador de tipo suspendido" consiguió que la tensión soportada fuese superior.

Según avanzaba el siglo, el desarrollo industrial trajo un gran aumento de la demanda. Se empezaron a crear centrales térmicas para asegurar la producción energética dejando en un segundo lugar a las primera centrales creadas, las hidroeléctricas. Además, se empezó a interconectar centrales que hasta ese momento funcionaban de forma aislada sin comunicación entre ellas. Esta serie

de acciones lograron un abaratamiento de costes de producción y desembocaron en la necesidad de normalizar las frecuencias entre dichas centrales. Por ejemplo, en 1924 en Inglaterra había 14 frecuencias diferentes en sus redes de AC y 43 tensiones distintas de suministro en baja tensión. En los años 30 Europa eligió la frecuencia de 50 Hz, en cambio Canadá y Estados Unidos se decantaron por 60Hz [21].

Dos avances importantes se produjeron a mitad de siglo. En 1954 se construyó cerca de Moscú la primera central nuclear que generaba energía eléctrica para el uso civil. En ese mismo año la empresa sueca *ASEA* construyó la primera red moderna que utilizaba DC para el transporte de alta tensión, denominado HVDC. Este sistema estaba formado por rectificadores en ambos extremos de la línea para convertir la DC en AC y esto permitía que las líneas de transmisión necesitasen menos aislamiento, evitasen caídas de tensión de tipo inductivo y la posibilidad de controlar la potencia transmitida.

2.3.2. Sistema actual

El SEP engloba las instalaciones y los equipos para generar, transportar y distribuir la energía eléctrica. Según se detalla en la Figura 2 [23], estas infraestructuras están formadas por tres etapas tres subsistemas y la aparamenta eléctrica.

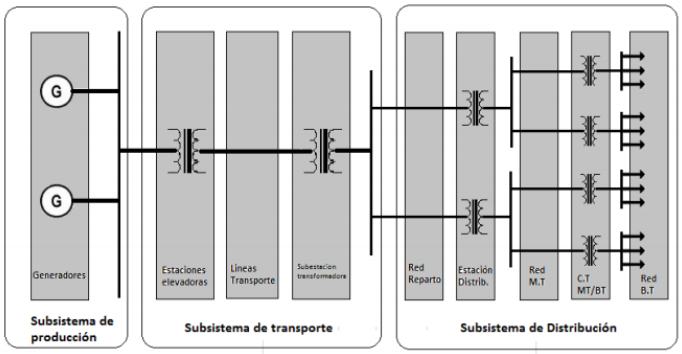


Figura 2. Sistema SEP

Las etapas están formadas por la adaptación, transformación y maniobra.

Los subsistemas son la producción desde centrales generadoras, seguido por el transporte mediante líneas de transporte y centrales de transmisión, y por último la distribución por medio de redes de reparto y líneas de distribución.

La aparamenta eléctrica sirve para regular, medir, proteger y controlar las instalaciones tanto de alta como de baja tensión y aumenta la seguridad, la calidad y la eficiencia.

2.3.3. Futuro

Algunos de los retos a los que se enfrenta el sistema energético son la falta de oferta, el encarecimiento y el cambio climático debido respectivamente, al aumento de la población en las ciudades, el encarecimiento del coste debido al agotamiento de las reservas de los combustibles fósiles y el impacto ambiental, si no se sustituyen por las energías renovables definitivamente.

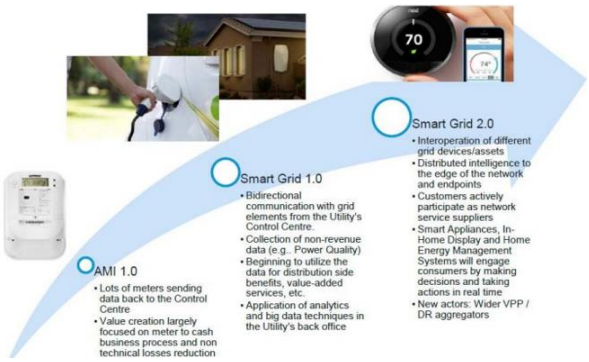


Figura 3. Evolución del Smart Grid

La inclusión de las TIC en el SEP ha tenido una evolución que refleja la Figura 3 [24]. Primeramente, empezó por AMI que contempló los contadores eléctricos que mandaban señales a la central. Después evolucionó a Smart Grid que tuvo dos fases [24]. La primera versión introdujo la comunicación bidireccional entre central y contador eléctrico, la recopilación de datos para determinar la calidad y mediante el análisis ofertar servicios de valor añadido. La segunda versión permitió la interacción directamente con el equipamiento que el cliente tiene en casa para tomar decisiones en tiempo real.

Teniendo en cuenta lo comentado podemos afirmar que las previsiones no fueron acertadas acerca del desarrollo de la Smart Grid 2.0. En [25] se creía que el FTTH suministraría a la vez datos y energía eléctrica y la creación de una plataforma abierta para el intercambio de energía. Y en esa línea [26] auguró que la electricidad se transmitiría en paquetes de “energía” que serían despachados por switches, routers y gateways.

Actualmente es denominada como Internet of Energy (IoE). Este concepto engloba a todos los controladores, sensores, actuadores y medidores instalados a lo largo de la red eléctrica [27]. Este avance permite dotar de inteligencia monitorizando su comportamiento en tiempo real y enviando instrucciones para automatizarlo. Con la ayuda del *big data* y el *cloud computing* la producción sería gestionada eficientemente e incorporaría seguridad a la distribución.

La modernización de las redes se basa en desarrollar la capa de análisis de datos, capa de aplicación, capa de control y comunicación y capa de equipamiento físico en el extremo de la red.

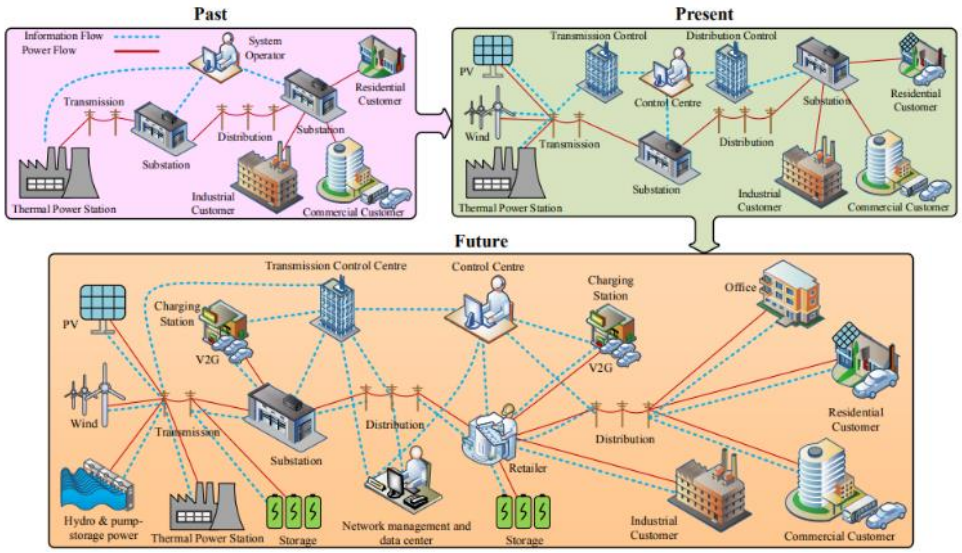


Figura 4. Evolucion del SEP

La Figura 4 [28] muestra el pasado y el futuro de las redes. El cuadro rosa refleja cuando el operador solo manejaba información de las centrales generadoras y las subestaciones de transporte. El cuadro verde describe como los datos controlados por remoto ya se van extendiendo a más etapas, como en el control de transmisión que aparece debido a la irrupción de las energías renovables. Por último, se observa cómo será el control sobre la generación de las energías renovables, el almacenamiento y el consumo del cliente final.

A pequeña escala hay proyectos enfocados a las PYMES con bajo presupuesto [29] para que sean capaces de implementar una maquina IoT, virtualizar procesos, extraer los datos para analizarlos en la nube y tomar decisiones sobre como eficientar sus sistemas. En otros estudios se presentan prototipos [30] que sirven para monitorizar y tomar acciones para alargar la vida de las baterías distribuidas en distintos terminales de una empresa.

Un punto de clave de este desarrollo es el despliegue de baterías para almacenar la electricidad. La Unión Europea presenta varias iniciativas para la investigación en este campo [31]. Por otro lado, hay estudios [32] donde la implementación de una red de baterías y la instalación de placas solares en los tejados por parte de los usuarios permitiría ahorrar costes al almacenar la energía en “horas valle” para consumirla en “hora punta”.

A gran escala la idea es convertir en *smart* cada una de las partes del SEP haciendo uso de tecnologías como el IoT o BC que aportará seguridad en las transacciones como se detallará en el siguiente apartado. Algunos de los objetivos son una distribución eficiente, permitir descentralizar la arquitectura a fin de aportar robustez y mejorar la seguridad de la distribución para evitar pérdidas y robo de electricidad [33]. La seguridad es crucial para la detección de

fallos en tiempo real y estar preparados frente a ataques cibernéticos. Por ello, se pretende incluir el BC en un sistema que mida correctamente los flujos eléctricos en los nodos del sistema [34] y en la plataforma para predecirlos [35].

2.4. BlockChain

La tecnología de “cadena de bloques” o blockchain (BC) es un sistema descentralizado, público y abierto que está basado en las redes *peer-to-peer* y sobre una base de datos distribuida.

El principal objetivo de incluirlo en el SEP es aprovechar su principal ventaja, la seguridad. El BC dificulta el hackeo gracias a que los datos de operaciones y registros son íntegros: no es posible corromperlos, modificarlos o borrarlos, sin que los usuarios se percaten. Además, su funcionamiento es transparente, anónimo y ejecutado en tiempo real.

Como se observa la Figura 5 [35] cada bloque que forma una cadena tiene registradas las transacciones y el intercambio de datos entre los propios usuarios. Los integrantes de esta red son a la vez usuarios y nodos, por lo que estas transacciones deben ser verificadas y posteriormente consensuadas mediante un algoritmo.

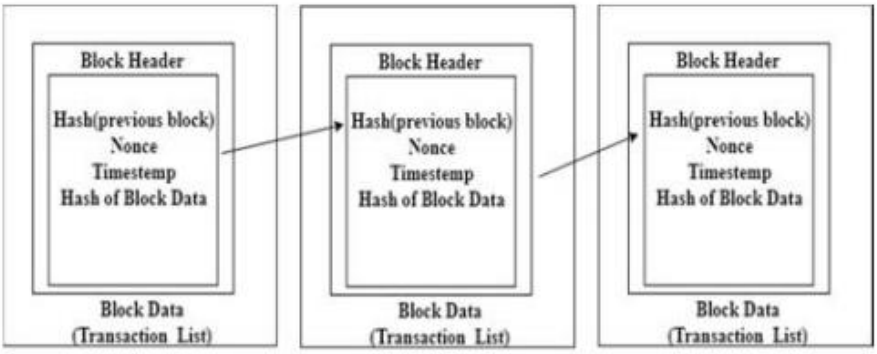


Figura 5. Ejemplo de cadena de bloques (BC)

Actualmente, la importancia de la tecnología BC tiene el origen por resolver muchos problemas que origino crear una moneda intercambiable en internet [36]. El punto de inflexión para obtener de manera segura y estable una moneda electrónica por Internet tuvo lugar cuando Nakamoto incorporó el BC al desarrollo del Bitcoin. A partir de aquí el BC se empezó a utilizar al sector financiero. Un ejemplo es el servicio de pago ‘One Pay FX’ lanzado en el 2018 por el Banco Santander. Otra solución del Nasdaq fue representar digitalmente la propiedad de acciones bajo la aplicación “Linq”. Ambas fueron viables gracias a la descentralización y el anonimato al validar las transacciones.

En la Smart Cities son varios los sectores en los que el BC tiene gran utilidad y ha sido ya empleado. El articulo [37] contempla los siguientes casos:

- El almacenamiento de los datos de los ciudadanos que se generan a diario de manera segura gracias al BC para después acceder y utilizarlos forma consciente y personal.

- En el sector transporte, el BC facilita la gestión y mejora la seguridad al crear un sistema descentralizado.
- Para la gestión de la cadena de suministro donde permite tener un control exhaustivo de la trazabilidad (procedencia, los puntos intermedios y el destino) de cualquier producto.

En el área de la salud [38]. Las ventajas que aporta el BC son mejorar la privacidad y la seguridad en el almacenamiento y tratamiento de los datos de los pacientes en el sistema de salud pública que se consiguen mediante una modificación del BC en el Bitcoin para que sea privado y centralizado.

Por último, el sector de la energía en la SC es un campo fundamental para el desarrollo humano. Aplicar BC en este sector aún está en vías de implantación a gran escala debido a la complejidad de la diversidad de subconjuntos que la conforman y que impiden su madurez plena.
Hay varios estudios como [39] donde utilizan el BC para construir una *smart contract* para el intercambio de energía entre el consumidor, el prosumidor y el productor. O la visión global que [40] ofrece sobre el sector de la energía, donde presentan al BC como solución teórica para mejorar y optimizar dicho sistema. Además, este mismo estudio presenta ciertos casos reales como un piloto en Alemania donde una plataforma basada en BC conecta sistemas de energía renovable con los consumidores o como en Reino Unido donde se ha aplicado para gestionar que la energía producida llegue al consumidor más cercano evitando un desperdicio en la propia distribución que de media se sitúa en el 10% del total generado.

2.5. Conclusión

En este apartado hemos analizado como el desarrollo de la *Smart City* está estrechamente ligado a la introducción de las nuevas tecnologías. Entre las muchas existentes hemos profundizado en el *Internet of Things,* la *Smart City* y el *BlockChain* que será la base de la que partiremos para edificar este trabajo.

Este proyecto está enfocado en sector energético. Lograr la eficiencia energética queda enmarcado en el campo del Internet of Energy y buscaremos encontrar la fórmula para transformar el actual sistema de consumo tradicional por uno eficiente, inteligente y seguro.

3. Análisis

En los siguientes apartados vamos a identificar y analizar entre todas las tecnologías que ofrece el mercado para poder desarrollar una plataforma de bajo coste para un entorno urbano que *tokenice* la energía generada por unas placas solares.

3.1. Topologías Red

A continuación, como muestra la Figura 6 analizaremos algunas de las típicas topologías de red que servirá como base al proyecto.

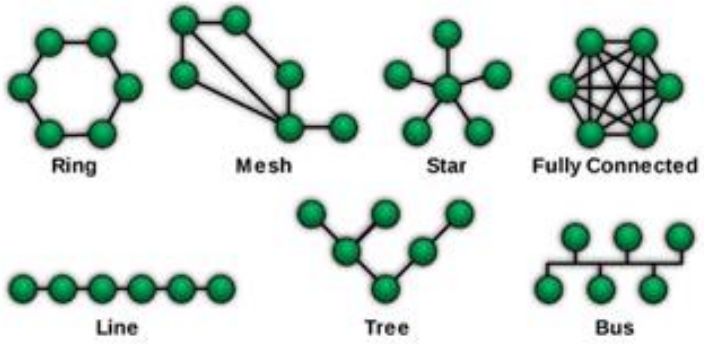


Figura 6. Topologías de Red

3.1.1. Estrella

Esta topología está formada por un nodo central y nodos periféricos. El nodo central es activo y soporta toda la carga de tráfico, todos los mensajes de la red pasan por este nodo central.

Es recomendable para redes pequeñas y locales ya que su escalabilidad es limitada. La contra es que tiene un claro riesgo de inoperatividad si cae el nodo central.

3.1.2. Mesh

Esta topología al contrario que la topología de estrella no requiere un nodo central, sino que provee un acceso equitativo.

Sin reducir rendimiento es capaz de proporcionar a todos los nodos que lo componen de las capacidades de ruteo dinámico para interactuar entre ellos. La forma de distribuir el tráfico de manera eficiente depende de la intensidad de señal que llega al dispositivo que se conecta, como por ejemplo un móvil, o de la congestión que tiene cada nodo.

Una red mallada se diferencia de una red con repetidores [Figura 7] en que la comunicación entre los nodos no implica hacerlo con el nodo central. Razón por la cual no es necesario que todos los nodos tengan visión con el principal, solo es necesario que tenga visión con algún otro nodo.

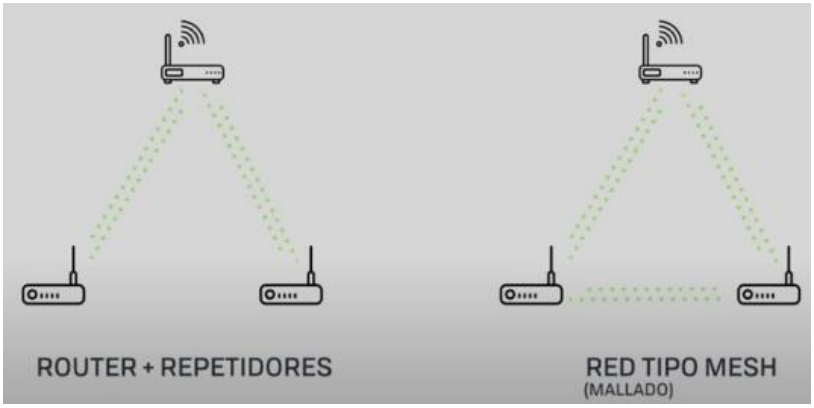


Figura 7. Red de router con repetidores y Red Mallada

Esta topología al utilizan los estándares tradicionales 802.11 a/b/g de WiFi se recomienda para redes inalámbricas que cubra grandes espacios en donde cablear no es la mejor opción. Únicamente es necesario que un único nodo esté conectado a la red de manera física, generalmente el router, para transmitir inalámbricamente conexión al resto de la malla.

Ejemplo de redes mesh son entornos [41] de *smart grid* rurales donde es necesario el bajo consumo sin necesidades de intercambiar gran cantidad de datos. El estándar Zigbee utiliza este tipo de tecnología para entornos [42] parecidos.

3.1.3. Árbol

La topología árbol, conocida como topología de estrella expandida o topología jerárquica, se basa en que desde un único nodo se conecte y sea el punto de partida de toda la red ramificada de manera jerárquica. Es posible verlo como una combinación de las topologías en estrella y de bus. Al ser una red jerárquica cada nivel está conectado al superior.

Ventajas son la reducción del tráfico, detecta rápido de fallos, monitoreo centralizado, posibilidad de crear subredes dentro de la red global permitiendo a los usuarios tener varios servidores en la red y la facilidad de escalar necesitando únicamente conectarse al nodo central o a algún otro nodo de un subnivel de la jerarquía. Las desventajas a nivel de cableado con las tecnologías inalámbricas se quedan a un lado, por lo que tener un único nodo central se presenta como el punto flaco.

Enfocada a grandes redes de cualquier tamaño, necesario un nodo central y en cuando sea necesario diferenciar grupos.

3.2. Dispositivos

En este apartado analizaremos que dispositivos son necesarios para crear una red en un entorno urbano.

La conexión a internet será proporcionada por un router-wifi que será la puerta de entrada a la red.

La red será formada por placas solares, baterías, sensores y otros dispositivos. Servirán para obtener los datos de la energía generada, la energía consumida y además para ejecutar tareas de intercambio con la red de transporte y distribución (RTD).

- Las placas solares obtendrán la energía solar y mediante un transformador lo convertirán en electricidad.

- Las baterías tienen el objetivo de almacenar energía proveniente tanto de las placas solares como de la RTD dependiendo del caso.

Teniendo en cuenta lo anterior y como refleja la Figura 8 habrá un punto de conexión de entrada (PE) con la RDT. Este PE será el lugar del sistema donde se ejecutará las decisiones tomadas por la aplicación alojada en la nube. En dicha aplicación es donde con los datos recabados del sistema se decidirá la venta de excedente de electricidad con la RDT o la compra de lo necesaria para llenar las baterías y así dar servicio a los clientes de la red.

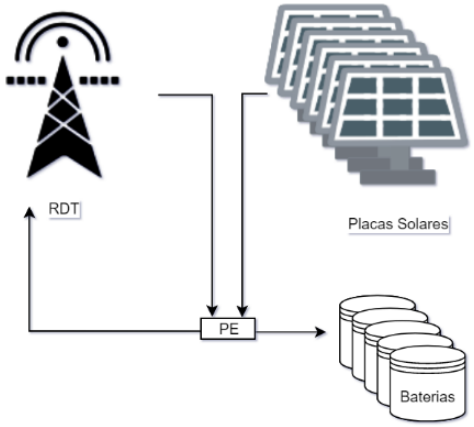


Figura 8. Red base

El sistema está diseñado para que los usuarios de la red siempre utilicen la energía almacenada en las baterías. Por lo tanto, serán conectadas simultáneamente al PE y a la red interna para distribuir la energía en cada casa según las necesidades.

El aplicativo alojado en la nube decidirá de donde obtener la energía, desde las placas o desde la RDT. Esto va a depender de los precios de venta de la RDT

en tiempo real, la distribución de la cantidad generada de las placas por horas y el estudio del consumo de la demanda de los usuarios.

Los datos que se utilizarán para realizar la gráfica de la demanda, hacer las previsiones y ejecutar las tareas en cada momento saldrán de los sensores [Figura 9] que habrá que desplegar en distintos puntos de la red.

Además de los sensores debemos tener en cuenta que dispositivos serán los que conecten dichos sensores al nodo central para poder enviarlo a la nube.

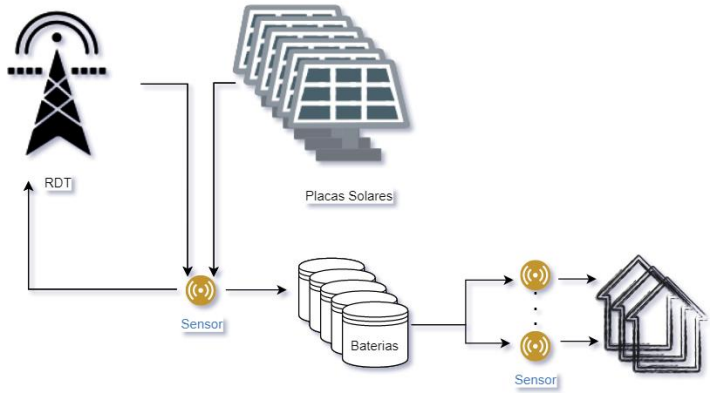


Figura 9. Sensores en la red.

Para ello la red se dimensionará con dos tipos de nodos que llamaremos nodo final y nodo primario.

- El Nodo Final (NF) se encargará de recolectar los datos, comprobar que son fiables y enviarlos al nodo NP.
- El Nodo Primario (NP) será el encargado de firmar los *smart contract* acercando el *edge computing* al extremo de la red, enviar los datos a la nube y recibir las tareas a ejecutar en los distintos dispositivos.

En la Figura 10 observamos, según la topología elegida, la colocación de los nodos, sensores y router que hemos comentado.

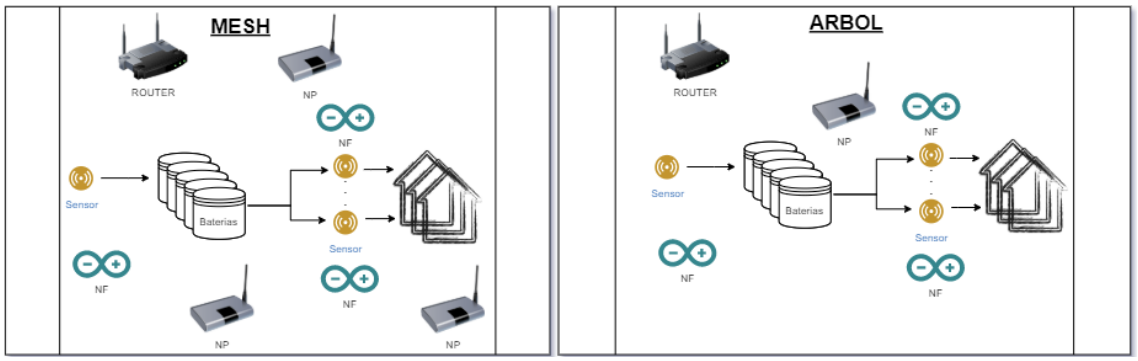


Figura 10. Red sobre una topologia mesh y sobre una topologia arbol

3.2.1. SoC

Los dispositivos *system on a chip* (SoC) tratan señales digitales y analógicas mediante un microcontrolador para ejecutar tareas como un autómata. Además, son económicos y de bajo consumo de energía.

Estos dispositivos van a llevar a cabo las funciones de los nodos primarios. Ya que están optimizados para automatizar tareas e interactuar con los sensores y los actuadores.

Entre las opciones posibles hemos decido analizar las más populares, Arduino y Espressif. Las razones son que poseen el apoyo de un amplio grupo de desarrolladores y una gran comunidad que aporta continuamente proyectos nuevos. Esto hace que tengamos un soporte “no oficial” que nos garantiza tener una solución que no quedará obsoleta a medio plazo.

Arduino es una plataforma abierta tanto hardware como software. El punto predominante es la sencillez de implementación tanto conexiones físicas como las formas de programar.

Físicamente consta de una placa de desarrollo con un microcontrolador Atmel con arquitectura ARM de 32 bits y puertos de comunicación de entrada y de salida.

A nivel software proporciona un entorno de desarrollo integrado o un IDE (integrated development environment) compuesto por un lenguaje de programación, las herramientas para transmitir código al micro y un bootloader que ejecuta la placa.

Esto permite crear proyectos con un control directo sobre cualquier tipo de sensores y de actuadores. La comunidad ha desarrollado numerosas librerías programadas en C++ y compiladas en lenguaje máquina. Esto es muy importante porque la placa no incorpora un sistema operativa al uso y es lo que le convierte en un elemento de ejecución de tareas en tiempo real, de inmediatez y con un bajo consumo.

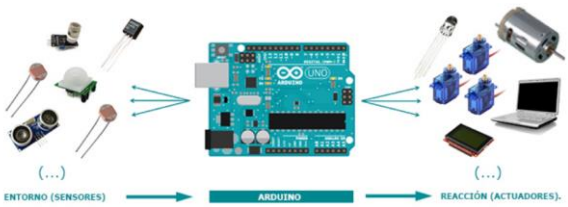


Figura 11. Placa Arduino y entorno

En la Figura 11 se observa cómo funciona la placa [43]. Obtiene datos de sensores y puede ejecutar tareas en los actuadores. Además, hay una línea de placas para IoT que incorporan conectividad wifi.

Espressif es otro fabricante erradicado en China de placas de desarrollo con un microcontrolador con arquitectura RISC de 32 bits y para comunicarse ofrecen un puerto serial UART.

A nivel de programación proporciona distintas posibilidades para programar que hasta que se liberó su API en el 2014 el único entorno para programarlo era el IDF. A partir de ese momento se dieron los pasos precisos para integrarse con el IDE de Arduino. También es posible utilizar otros lenguajes como MicroPython o Javascript.

Desde el inicio se integró en la placa un módulo wifi con gestión de TCP/IP. Por ello muchas empresas como Itead lo eligieron para crear sus productos de domotización.

El modelo ESP8266 tiene numerosas versiones, aunque las más utilizadas son la ESP-01 y ESP-12 [44]. Y el modelo ESP-32 es el de más potencia con mayor procesamiento, bluetooth y con sistema de bajo consumo.

En la Tabla 1 veremos la comparativa entre Aduino y ESP.

Tabla 1. Comparativa de los SoC

	Arduino Uno	**ESP 8266**
Microcontrolador	ATmega328	Tensilica Xtensa X36
Arquitectura	RISC 8-bit	32 bits
Alimentacion	5V	3,3
Voltaje de Entrada	7-12V	2.5 V ~ 3.6 V
Pin Digital I/O	14 (6 salidas PWM)	16 (8 salidas PWM)
Corriente DC I/O Pin	40 mA	80 mA
SRAM	2 KB	160 KB
EEPROM	1 KB	NO
Reloj	16 MHz	80 MHz
WIFI	No	Si

3.2.2. SBC

Los llamados ordenadores de placa única o single board computer están formado fundamentalmente por la memoria, entradas/salidas y un microprocesador.

Las funciones que deben realizar los NP necesitan, al contrario de los NF, como unidad principal un microprocesador. Con un mayor poder de procesamiento son capaces de realizar tareas pesadas y descargar así a los NF de tareas que exijan mayor dedicación.

Como en el caso anterior nos hemos decantado por las opciones más extendidas por razones parecidas: amplia oferta de dispositivos, gran apoyo de desarrolladores, una gran comunidad que aporta los últimos desarrollos y un modelo a futuro estable.

Hay una gran oferta entre las placas SBC donde encontramos las Raspberry Pi, Banana Pi HummingBoard, Pandaboard, PCDuino y las BeagleBone.

Raspberry Pi (RBP) pertenece a la Fundación Raspberry Pi fundada en 2009 para ofrecer un entorno educativo mediante la creación de un mini-ordenador.

Las características principales son un procesador ARM, memoria de hasta 1GB, conexiones de usb, de video y de sonido y conectividad wifi, bluetooth y ethernet.

A nivel software posee un sistema operativo limitado. Entre los distintos distros encontramos principalmente los basados en Linux como Raspbian e incluso hay versiones especiales de Windows. Python es el lenguaje principal con el que se programa, de ello viene parte de su nombre: Pi. Se alimenta por un microusb y se le puede conectar un ratón y un teclado por sus puertos.

Los pasos que hay que seguir para realizar un proyecto es primeramente realizar un programa en Python para luego compilarlo y ejecutarlo en el sistema operativo de la propia RBP. Estas capas adicionales de software le permite ser un dispositivo multitarea, aunque también añaden cierto retraso en el control de su puerto GPIO. Por lo que no es muy recomendable trabajar con equipos que necesiten supervisión y acción inmediata. Para ello se usan los SoC.

Hoy en día la RBP es el más vendido y gracias a la potencia ofrecida lo hace ideal para diversos proyectos de uso doméstico. Podemos realizar un básico ordenador de escritorio, sistema central para una smart home o hasta servidores web [45] [46].

Otras opciones que podemos encontrar son:

- *HummingBoard* es un SBC fabricado por SolidRun desde 2014. es muy similar a la RBP pero con mayor potencia y un precio superior.

- *Banana Pi* compuesto por un Allwinner A20 (doble núcleo Cortex-A7 a 1 GHz., GPU Mali400MP2), 1 GB de memoria RAM DDR3, SD y HDMI.

- *BeagleBone Black* está basado en tecnología ARM, CPU Ti Sitara ARM Cortex-A8 a 1 GHz., 512 MB de memoria DDR3 y gráficos SGX530. Es la gran competencia con RBP por su mayor potencia de placa, tiene una distribución Debian preinstalada y es escalable con gran compatibilidad entre dispositivos.

En la Figura 12 se muestra cómo se conecta cualquier tecnología a la nube por medio de una SBC.

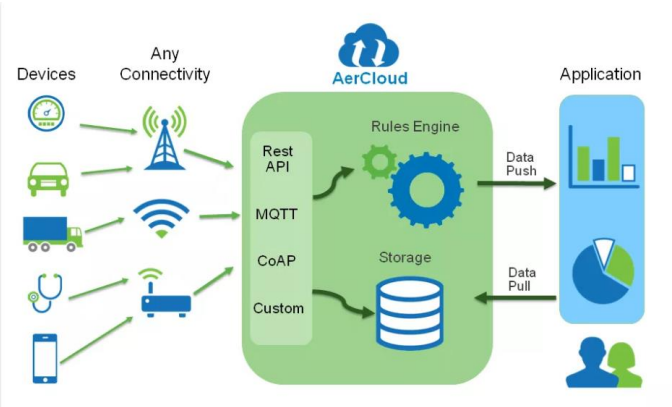


Figura 12. Entorno de un SBC conectado a la nube

La Tabla 2 muestra una comparativa de los distintas modelos de SBC comentados.

Tabla 2. Comparativa de los SBC

	RPI MODEL A	RPI MODEL A+	RPI MODEL B	RPI MODEL B+	RPI 2 MODEL B	PC-Duino	BeagleBone
SOC - chip	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2836		TI AM3359
CPU	ARM11 ARMv6 700 MHz.	ARM11 ARMv6 700 MHz.	ARM11 ARMv6 700 MHz.	ARM11 ARMv6 700 MHz.	ARM11 ARMv7 ARM Cortex-A7 4 núcleos @ 900 MHz.	1GHz ARM Cortex A8	ARM Cortex-A8 1 GHz
GPU	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0	OpenGL ES2.0, OpenVG 1.1 Mali 400 core	PowerVR SGX530
RAM	256 MB LPDDR SDRAM 400 MHz.	256 MB LPDDR SDRAM 400 MHz.	512 MB LPDDR SDRAM 400 MHz.	512 MB LPDDR SDRAM 400 MHz.	1 GB LPDDR2 SDRAM 450 MHz.	1GB	512 MB DDR3.
USB 2.0	1	1	2	4	4		2
HDMI - SALIDAS DE VÍDEO	HDMI 1.4 @ 1920x1200 píxeles	HDMI 1.4 @ 1920x1200 píxeles	HDMI 1.4 @ 1920x1200 píxeles	HDMI 1.4 @ 1920x1200 píxeles	HDMI 1.4 @ 1920x1200 píxeles		microHDMI
audio	3.5 mm jack						microHDMI
ALMACENAMIENTO	SD/MMC	microSD	SD/MMC	microSD	microSD		SD/MMC
ETHERNET	No	No	Sí, 10/100 Mbps	Sí, 10/100 Mbps	Sí, 10/100 Mbps		Sí, 10/100 Mbps
TAMAÑO	85,60x56,5 mm	65x56,5 mm.	85,60x56,5 mm	85,60x56,5 mm	85,60x56,5 mm		85,60x56,5 mm
PESO	45 g.	23 g.	45 g.	45 g.	45 g.		45 g.
PRECIO	25 dólares	20 dólares	35 dólares	35 dólares	35 dólares		35 dólares
WIFI							80211 b/g/n 2.4 GHz / 5 GHz
BLUETOOTH							4.1

3.2.3. Paneles solares, baterías y sensores

Para terminar de analizar los distintos dispositivos que van a formar la red pasamos a describir los siguientes elementos:

- Paneles solares fotovoltaicos. El tipo de panel monocristalino es el más eficiente. Aunque si no se tiene limitaciones de espacio para su instalación el panel del tipo policristalino es el más vendido.

- Inversores autoconsumo transforman la energía en corriente continua (CC) procedente de los paneles solares en energía de corriente alterna (CA) y mediante un bucle de enganche de fase sincronizar fase y frecuencia con la energía de la red de distribución eléctrica.

- Baterías son la parte más cara de la instalación, con una vida aproximada de 15 años. Dos tipo: plomo-ácido y las de litio que son las más rentables

- Reguladores de carga: controlar la corriente que es absorbida por la batería y así evitar que se sobrecargue peligrosamente. Al mismo tiempo evita que se deje de aprovechar la energía captada por los paneles o el aerogenerador.

- Sensores requieren una serie de características: bajo consumo, coste económico, cobertura en entornos cerrados, ancho de banda baja, baja latencia y escalabilidad.

3.3. Blockchain

Este apartado está compuesto por varios conceptos importantes. Analizaremos los cifrados para el intercambio de información, los protocolo de consenso para confirmar si es una transacción se ha realizado correctamente, los *smart contract* para implementar el código dentro de la red blockchain y las distintas tecnologías blockchain existentes.

Para utilizar la tecnología blockchain necesitamos implementar y vincular un nodo a la red y crear una aplicación mediante un *smart contract (SC)*. El SC sirve para que se ejecuten transacciones sin una autoridad central.

La red para que sea escalable debe ser configurada, administrada y monitorizada continuamente a nivel de infraestructura, gestión de usuarios y software. Afortunadamente ya existen plataformas que cumplen dichos requisitos. Aunque tienen diferentes enfoques de implementación y desarrollo, y distintos campos de aplicación.

Estas plataformas, que usan internet para su continua expansión, están formadas por bloques o listas de registros y protegidos por algoritmos criptográficos o hash.

3.3.1. Cifrado

Un algoritmo de cifrado o de encriptación son funciones que, mediante una clave de cifrado, permite transforma un mensaje en claro en una serie de caracteres aparentemente aleatorio.

Gracias a este tipo de funciones es posible transmitir información privada públicamente por internet y evitar que otros accedan al mensaje fácilmente.

Hay diferentes tipos de cifrados. Vamos a hacer un repaso a los más populares:

- El *Secure Hash Algorithms* (SHA) es una familia de funciones hash creadas por el National Institute of Standards and Technology (NIST). Entre todas las versiones existentes, el SHA-256 es el utilizado por numerosas aplicaciones como TLS, SSL, SSH, IPsec o por el Bitcoin.

- El *Advanced Encryption Standard* (AES) es un algoritmo estándar que permite usar claves de 128 bits, 192 bits e incluso de 256 bits. Excepto los ataques de fuerza bruta, es bastante resistente al resto de ataques.

- El *Rivest, Shamir y Adleman* (RSA) es un algoritmo de cifrado asimétrico de dos tipos de clave. Una clave pública para cifrar y otra privada para descifrarlos. Es el estándar para cifrar la información intercambiada a través de Internet.

- El *Triple Data Encryption Standard* (3DES) es el sucesor del algoritmo DES, se basa en tres teclas individuales con 56 bits cada una y con una longitud de clave de 168 bits. Utilizado en el sector bancario y financiero.

- *Blowfish* es algoritmo de cifrado simétrico, rápido y flexible. Cifra individualmente cada uno de los bloques de 64 bits en que divide los mensajes. Utilizado en los pagos en comercio electrónico o controlar contraseñas.

- *Twofish* es el sucesor del Blowfish. Utiliza una única clave, es simétrico y con una longitud de hasta de 256 bits.

Para algoritmos simétricos o una función hash el tamaño de clave de 128 bits es suficiente, aunque puede llegar a ser de 256 bits para aplicaciones críticas. En cambio, para un algoritmo asimétrico el tamaño debe llegar a 1280 bits al menos, aunque lo recomendable son 2048 bits para necesidades especiales.

3.3.2. Algoritmo Consenso

Un algoritmo de consenso es una forma dinámica por la que un grupo de usuarios llegan a un acuerdo. Este tipo de algoritmos se emplean en redes distribuidas de múltiples procesos aislados que se comunican mediante mensajes.

A parte de donde se realiza estos algoritmos su implementación también tiene en cuenta solucionar los fallos a nivel de red, caída de usuarios, malas sincronizaciones o ataques.

Actualmente hay dos tipos principales que se utilizan como algoritmos de consenso [Figura 13] [47], pero utilizan caminos diferentes [48].

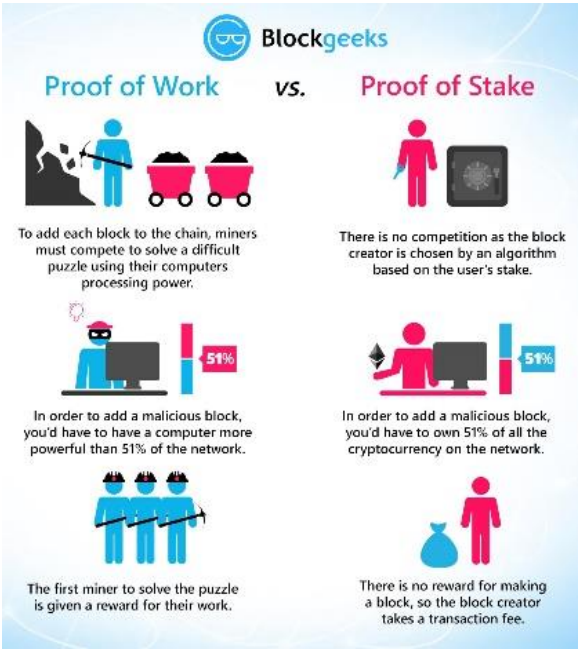


Figura 13. Comparativa entre PoW y PoS

PoW está basado en el término conocido como minería donde cada usuario debe resolver con sus propios ordenadores unos cálculos para lograr añadir un bloque por el que sí es el primero que lo hace se le recompensa.

Por el contrario, en el PoS nadie compite por ningún premio, no se mina ni se realiza cálculos intensivos porque está basado en las participaciones del usuario.

PoW: Proof of work

Este algoritmo [49] fue utilizado por Satoshi Nakamoto para implementar el *BitCoin*, ya que logra un consenso distribuido y confiable. No necesita tener una tercera entidad que lo valide como en los métodos tradicionales que generalmente esa entidad era representada por un un banco.

Todos los usuarios tienen una copia de todas las transacciones, por lo tanto, su hackeo es prácticamente inviable. Esto evita que un cliente consuma todos los recursos incontroladamente ya que provocaría una denegación de servicio. Y al estar basado en el protocolo HashCash, disuade ataques de denegación de servicios distribuidos (DDoS).

El término de minería se refiere a calcular el esfuerzo. El esfuerzo representa el trabajo a nivel de potencia de cálculo y en tiempo necesario para crear un bloque de la cadena. Cada bloque se corresponde con un grupo de transacciones dentro de la cadena. Podemos contemplar la cadena de bloque como un gran libro de contabilidad que lleva todas las cuentas.

El segundo paso, después de minar, es verificar. Esto se realiza de forma sencilla y rápida por parte del resto de los nodos que forman la red.

Verificar tiene dos objetivos: uno es legitimar la transacción y otro la recompensa. Si la respuesta es positiva se entrega la recompensa al minero que lo crea, por supuesto en moneda digital.

Sus principales características son:

- Seguridad. Evita la DoS y aumenta a mayor número de mineros. El objetivo son crear redes distribuidas grandes.

- Simplicidad. La implementación del algoritmo no es compleja, al igual que el mantenimiento software y las auditorias.

- Adaptabilidad. Permite añadir y prohibir cualquier tipo de tecnología (ASIC, GPU, FPGA, CPU) en la red según las necesidades de cada momento.

El principal problema:

- El consumo de energía eléctrica debido al intensivo trabajo computacional.

PoS: Prof of stake

Para combatir el problema principal del PoW que era la gran cantidad de energía que se necesita para crear un bloque, Vitalik Buterin creó la comunidad de Ethereum.

Este algoritmo en vez de mineros tiene validadores [50]. El método que usan es apostar parte de su dinero electrónico, llamada participación, por los bloques que intuyen que se van a unir a la cadena. Si finalmente se unen, dicho usuario se lleva una proporción a su apuesta. Esta comunidad utiliza una moneda electrónica basada en PoS en vez de en PoW.

Además, añade seguridad frente a ataques por cantidad de potencia ya que este algoritmo directamente no depende de ello. Sino más bien depende del tiempo que un usuario permanece en la red. Esto evita ataques de usuarios recién creados y que únicamente depende de la cantidad de monedas que tenga. Además, estas medidas también evitan la fuerte fluctuación que sufren las monedas virtuales.

Sus principales características:

- Seguridad. Evita ataques llamados del 51%.

- Usabilidad, escalabilidad y velocidad. Facilita la tarea para utilizarse en pagos de pequeñas cantidades. Permite realizar muchas transacciones en poco tiempo.

- Eficiente. Al no utilizar minería, el consumo energético es reducido. No necesita realizar trabajos intensivos, no se mina.

- Estabilidad. Los incentivos están asociados a permanecer en la red mucho tiempo.

- Ecuánime. Todos los nodos que cumplen su cuota de participación tienen la opción de participar en la red, no necesitan potentes maquinas. Las recompensas son más proporcionales a la cantidad de monedas que tiene cada usuario. Quien tiene más monedas, tiene más posibilidades de hacer verificaciones y recibir recompensas.

El mayor riesgo es:

- Perder todo el dinero debido a ataques si se descubre vulnerabilidades en la red. Esto es posible porque es obligatorio que todos los usuarios tengan la cartera siempre abierta y conectada a internet.

La Tabla 3 refleja los punto clave de la diferencia entre ambos consensos que acabamos de analizar.

Tabla 3. Comparativa entre PoW y PoS

	PoW	PoS
Formato	Minar	Partcipacion
Gasto energetico	Alto	Bajo
Rentabilidad	Dependiendo de lo minado	Lineal a todos proporcionado
Tipo	Generador dinero	Inversion
Seguridad	Mayor	Menor

3.3.3. Smart contract

El concepto *smart contract* fue introducido por Nick Szabo a mediados de los noventa. Es un protocolo para realizar las transacciones según los términos de un contrato. Dicho contrato debe cumplir una serie de condiciones. Este procedimiento sirve para eliminar los intermediario y minimizar las excepciones maliciosas y accidentales. Es un sistema descentralizado que no lo controla ninguna parte del acuerdo.

Actualmente este concepto se materializa en un código que representa el acuerdo entre las partes. Si se cumple una serie de condiciones se autoejecuta: es firmado por ambas partes y se introduce en un blockchain.

Es programado sobre el lenguaje orientado a objetos *Solidity*, se ejecutan sobre una máquina virtual de Ethereum (EVM) y es accesible a todos los nodos.

Solidity permite utilizar bucles, es decir, estructuras de codificación repetitivas. Esto permite crear aplicaciones más complejas, eficientes y de manera sencilla. Para evitar fallos como bucles infinitos que colapsarían la red, se introduce el concepto de GAS. Esta funcionalidad permite calcular el coste de las transacciones a nivel de gasto computacional para procesar una transacción o contrato inteligente en la red, y las necesidades de su almacenamiento.

Remix es el Entorno de Desarrollo Integrado (IDE) y está formado por un editor de código fuente, herramientas de construcción automáticas y un depurador de errores. La EVM se sitúa como un intérprete del IDE tanto a nivel del entorno del contrato como del lenguaje Solidity. Es posible utilizarlo de forma online *remix.ethereum.org*.

Cada nodo de Ethereum tiene su propia máquina virtual, aunque el concepto es que hay una única máquina que engloba a todas.

La EVM es una máquina de Turing completa, es decir, es capaz de resolver cualquier problema computacional siempre y cuando se disponga de un programa. Esta máquina es la encargada de ejecutar el código de los Smart Contracts. Una vez el SC se compila y se despliega en la red Ethereum, su contenido es traducido a bytecode, que es el tipo de instrucciones que entiende la máquina virtual.

Un ejemplo de una transacción en la red blockchain es la siguiente:

1- La transacción se ejecuta en la máquina EVM del nodo que ha generado la transacción.
2- Los nodos empiezan a buscar el hash específico.
3- El primero que lo encuentra añade el bloque a la cadena y lo distribuye a los demás nodos.
4- El resto de los nodos comprueban que el resultado es válido.

Como caso extremo hay organizaciones que son controladas basándose únicamente en este SC [Figura 14], son los llamados Decentralized Autonomous Organizations (DAO) [51].

Figura 14. Red DAO

En este tipo de organizaciones todas las transacciones son transparentes. El objetivo es que la toma de decisiones entre en un nuevo nivel de cooperación y sea utilizado entre humanos, máquinas y otros smart contracts. Ejemplos de organizaciones enteramente DAO son The DAO en Ethereum, Dash DAO Governance y DigixDAO.

Las ventajas [52]que presenta el uso de este tipo de contratos son:

- Encriptación. Este tipo de contratos son encriptados lo que añade confianza y mejora al contrato tradicional.
- Determinista. Añade seguridad al saber a priori como se va a comportar el *smart contract* según las condiciones prefijadas.
- Confiable frente a manipulaciones. No es posible modificar los contratos una vez realizados.
- Agilidad. Aumenta al ser ejecutado automáticamente reduciendo así los costes y evitando errores humanos.

3.3.4. Tecnologías

El acicate del desarrollo de la tecnología de blockchain ha sido propiciado por el uso intensivo de las distintas criptomonedas. Su característica de *tecnología de registro distribuido* (DLT) ha ido evolucionando según las necesidades de los particulares o clientes como ahora veremos.

Principalmente hay dos tipos de blockchain, publica y privada, que veremos a continuación.

Publica

La Fundación Ethereum logró introducir el *contrato inteligente* a su red y significo un gran avance sobre el BC tradicional, el *Bitcoin*. A partir de este momento se tenía un método seguro digital y sobre internet para realizar transacciones sin necesidad de terceras entidades.

En este entorno se crean aplicaciones distribuidas para obtener inmutabilidad en los datos, evitar la censura y dar seguridad contra la piratería. Obliga a que los nodos deben estar activos continuamente, lo que permite tener un acceso, una privacidad y una confidencialidad totalmente transparente.

La transparencia obliga a hacer todo de forma pública. Cualquiera nodo tiene acceso a toda transacción que hace otro usuario.

El consenso, donde participan todos los usuarios, se basa en PoS, el algoritmo de consenso programado en Solidity que hemos visto. Los usuarios que participan tienen un recompensa mediante el pago de la moneda Ether. Este tipo de consenso se puede utilizar para sistema de votaciones como muestra el artículo [53].

Empresas como Microsoft, Accenture, J.P. Morgan o el Santander crearon la *Enterprise Ethereum Alliance.* Esta alianza permitió desarrollos personalizados que desarrollan estas mismas empresas. Microsoft implementó servicios para vender a terceros como el *BC as a Service* (BaaS). El banco J.P. Morgan creó una plataforma de BC privada enfocada al sector financiero con el objetivo de ofrecer soluciones seguras y rápidas transacciones sobre Ethereum llamada Quorum.

Privada

Precisamente la transparencia hizo que otras empresas que necesitaban privacidad en sus transacciones buscasen alternativas. Para ello era necesario una tecnología que modificando la red Ethereum aun conservase la posibilidad de seguir utilizando los SC pero de forma privada. El resultado fue Hyperledge.

Hyperledge es un concentrador de código tanto de protocolos como de estándares abiertos y posee una arquitectura modular. Su objetivo es que los desarrolladores creen soluciones para la industria. Fue creado a finales del año

2015 por la *Linux Foundation* y ya lo usan centenares de empresas, entre las que destacan IBM, Microsoft, Cisco, Intel o SAP.

Su modularidad permite agrupar diferentes tecnologías de forma rápida y sencilla para desarrollar soluciones. Únicamente es necesario unir varias funciones para obtener el servicio necesario.

El código es escrito desde la plataforma de código abierto Hyperledger Fabric. Este código o *chaincodes*, se usa para escribir los consensos, donde se eligen que nodos se les permite participar y a cuáles no.

Al no haber implementado ningún mecanismo de consenso ni haber minería, se ofrece la libertad al usuario de crear un algoritmo de consenso propio. El programador tiene la posibilidad de implementar varios niveles de transacción y que no todos los nodos participen. Generalmente los nodos validadores se ejecutan dentro del "contenedor Docker" de Amazon. Estos nodos son los encargados de validar las transacciones, mantener el registro y llevar a cabo los consensos como por ejemplo en el proyecto [54]

Las empresas lo eligen para tener privacidad y confidencialidad, es decir, controlar la visibilidad de las transacciones. Y esto lo hace gracias al uso de claves.

Hay varios ejemplos de proyectos que mejoran el IoT, la red de producción, la escalabilidad, el rendimiento y la confiabilidad. Por ejemplo, el Hyperledger Explorer sirve para ver el estado de tu plataforma, el Hyperledger Sawtooth permite implementar el mecanismo de consenso, y el Hyperledger Fabric es una plataforma modular que añade flexibilidad para programar en distintos lenguajes como Java, GO o Node.js.

Corda es una plataforma de BC enfocada al sector financiero y creado por la empresa R3. Esta plataforma de BC de código abierto permite interactuar y gestionar fácilmente contratos legales entre entidades. Además de tener compatibilidad de otros servicios entre organizaciones que forman la red y que confían mutuamente.

Mediante transacciones complejas, implementados en *smart contract*, permite controlar y autorizar la contabilidad distribuida. Con el objetivo de obtener un alto nivel de privacidad, seguridad y accesibilidad para datos confidenciales utiliza estándares de la industria programados en JavaScript y JVM.

Admiten una amplia variedad de mecanismos de consenso entre individuos que realizan transacciones, en lugar de uno para todo la red. Tampoco tiene moneda electrónica.

A continuación, vemos en la Tabla 4 comparando las tres tecnologías [55] que hemos estado analizando.

Tabla 4. Comparativa de tecnologias de BC

CARACTERISTICA	ETHEREUM	HYPERLEDGER	CORDA
Tipo	Publica: acceso libre	Privada o Federado. Necesario permisos.	Privada o Federado: necesitas permiso
Descripcion	Plataforma BC generica	Plataforma VC Modular	Plataforma distribuida
Usos	General. P2P y B2C	B2B en empresas.	Registros distribuidos para la industria financiera.
Administrador	No	1 adm en Privado Varios adm en Federado	1 (Privado) - Varios (Federado)
Smart Contract	Si	No. El usuario puede crearlo.	Si
Mecanismo de Consenso	Mineria basado en PoW. Lento y necesita de poner de acuerdo a todos los usuarios.	No. El usuario puede crearlo con niveles de transaccion.	Al no tener mineria necesita un Algoritmo de consenso donde Solo las partes involucradas
Confidencialidad	Transparente	Privadas	
Privacidad	No	Si	Si
Lenguaje de programacion	Solidity	Goland(Google), Java, Node.js	Kotlin
Propietario	Ethereum Fundation (DAO)	Linux Foundation	Empresa R3
Codigo abierto	Si	Si	Con restricciones
Moneda	Ether	No. El usuarios puede crearlo	No
Coste de Transaccion	Si	No	No
Registro	Publico	Privado	Privado

3.4. Tecnologías LPWAN

Las comunicaciones inalámbricas que abarca las tecnologías LPWAN ofrecen conectividades aplicadas a redes de áreas extensas (WAN) para conectar dispositivos sin grandes necesidades de ancho de banda y con un bajo consumo.

Están orientadas a redes celulares con dispositivos que pueden ser móviles ya que cada celda abarcar varios kilómetros

La Figura 15 muestra en qué lugar del espectro están las distintas tecnologías catalogadas por el ancho de banda y su rango de cobertura.

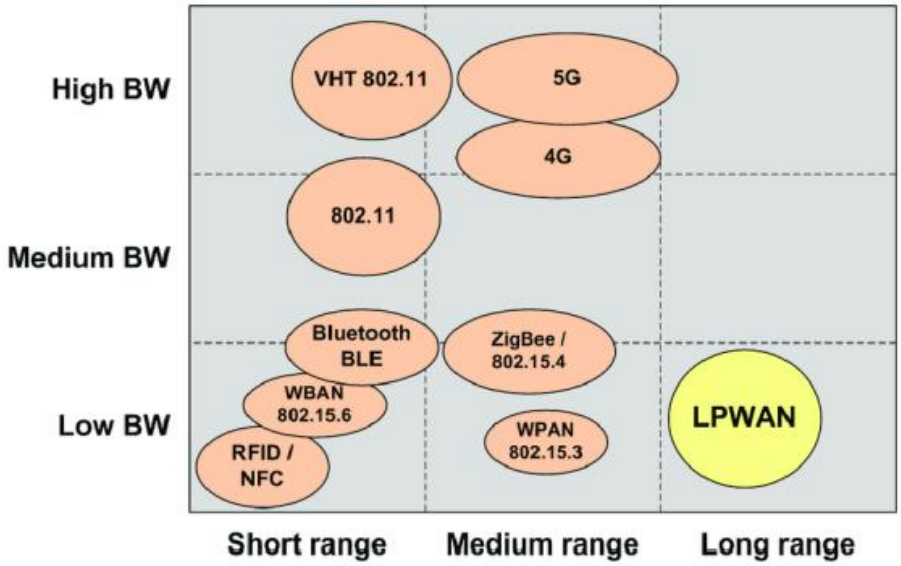


Figura 15. Cuadro comparativo entre tecnologia de conectividad

3.4.1. LoraWan

Es un protocolo estándar abierto patentada por Semtech Corporation. Posee una topología de red en estrella donde unos nodos hacen las tareas de gateway a otros nodos.

Tiene una modulación de espectro ensanchado o CSS (chip spread spectrum) que es un protocolo militar de principio del siglo veinte que permitía una separación optima entre las señales usando el factor de expansión. Esto servía para hacer frente a las interferencias y cubría largas distancias. Necesita un gateway de enlace formado por un modem multicanal para recibir mensajes por los diferentes canales.

Cubre un área extensa de baja potencia con “espectro no licenciado”. Sus ventajas son no estar bajo legislación gubernamental, su comunicación es bidireccional y sus dispositivos son de bajo consumo. Sus desventajas son tener grandes interferencias y mayor latencia que NB-IoT.

Además, debe ajustarse a las reglas del organismo regulador como el máximo de transmisiones que el emisor puede realizar, menos del 1% del duty cycle. Respecto a los niveles OSI, Lora cubre el nivel físico y LoraWan el nivel de enlace de datos o capa MAC.

3.4.2. SigFox

Es un protocolo propietario, no es abierto, creado por una empresa homónima. Posee una topología de red en estrella similar al LoRaWAN donde los dispositivos se conectan a las Base Stations que recopilan y reenvían a la nube SigFox[Figura 16] [56].

Figura 16. Entorno de una red tipo SigFox

Utiliza bandas de frecuencia no licenciadas para intercambiar mensajes limitados en cantidad tanto en subida como en bajada. Respecto a los niveles OSI Lora cubre desde el nivel físico al nivel de transporte.

No implementa mensajes de acuse de recibo, pero lo palia por tener el protocolo RF-TDMA y al duplicar las transmisiones por canales distintos.

3.4.3. NB-IoT

El protocolo NB-IoT fue lanzado por 3GPP a mediado del año 2016. Cubre entornos urbanos y permite mejorar la comunicación entre los dispositivos IoT que necesitan conectarse a una red móvil con una latencia y una tasa de datos baja.

Las operadoras necesitaron más tiempo para adaptar el software de las antenas e implementar el nuevo servicio, que otras soluciones como SigFox o LoRa.

El espectro de frecuencias, en la banda ISM de frecuencias entre los 790Mhz y 862Mhz, esta licenciado por lo que simplifica el hardware y el software aprovechando mejor la red. Esto produce ventajas como tener mejores coberturas, unos consumos de energía bajos y permite un ahorro de costes.

Se desarrollo es una simplificación del protocolo LTE teniendo en cuenta el tipo de dispositivos a los que iba dirigidos. Implementa un esquema de seguridad LTE donde la arquitectura troncal es EPC. Sus nodos tienen gran movilidad. Ofrece buena cobertura en interiores y tolerancia a la latencia

La Figura 17 muestra mediante un diagrama de características cuales son los puntos fuertes por tecnología.

Figura 17. Diagrama de las caracteristicas de SigFox, LoRa y NB.IoT.

A continuación, vemos en la Tabla 5 comparando las tres tecnologías.

	LORAWAN	SIGFOX	NB-IOT
MODULACION	FSK / CSS	GFSK / DBPSK	BPSK / QPSK
ANCHO DE BANDA DEL CANAL	125 kHz - 250 kHz	100 Hz	180 KHz /200 KHz
FRECUENCIA GRABAJO - MHz	868 (Europa) - 915 (USA) - 433 (Asia)	915 / 928	Licencias LTE: 700/ 800 / 900
TASA DE DATOS	Adaptativa: < 10,000 bps	< 100 bps	No
DIRECCIONALIDAD	Half-duplex		Half-duplex
MOVILIDAD	SI	SI	NOMADA
INTERFERENCIAS	Alta inmunidad		Baja
RUIDO	Alta inmunidad		
COBERTURA	Rural: 20 KM Urbano: 5 KM		Rural: 10 KM Urbano: 5 KM
SEGURIDAD	AES 128b		Encriptacion LTE
VENTAJAS	Inmune interferencias	Optimizada frec aleatoria al tx	Mejor cobertura
INCONVENIENTES	Alta latencia	Sin FEX (interferencias)	Sin Handoff(ruido)
TASA DE TX - kbps	50	0,1	200
THROUGHPUT - kbps	50	0,1	150
LATENCIA - ms	44105	10959	>10000
IMPLEMENTACION	Gateway	SigFox modem y network	SW upgrading
MENSAJES / DIA	Duty cycle	140 sms/dia	Ilimitado
MENSAJES (carga max)			
VENTAJAS	Tasa adaptativa	Alta fiabilidad	PSM eDRX
INCONVENIENTES	Tamaño de paquete	Seguridad	Perdida acuse de recibo
NUM EQUIPOS/CELDAS	<20,000	<20,000	<50,000

Tabla 5. Comparativa de las tecnologias LPWAN.

3.5. Protocolos de comunicación

Los protocolos de comunicación disponibles para aplicaciones de IoT sirven para conectar una gran cantidad de dispositivos cotidianos entre sí. Gracias a internet y los smartphones estos sistemas permiten controlar dispositivos tipo SoC o SBC que hemos visto anteriormente, de una forma sencilla, segura y rápida.

En el ámbito del corto alcance hay numerosas tecnologías como MQTT, CoAP, AMQP, XMPP, DDS, LwM2M…aunque nos centraremos en las dos primeras.

Estos mecanismos deben cumplir ciertos requisitos para realizar tareas como la necesidad de conectar un gran número de sensores, actuadores y servidores o puntos centrales, la seguridad y la simultaneidad de esas comunicaciones. La escalabilidad de la solución, la interoperabilidad entre los dispositivos que vayamos añadiendo a la red y la facilidad de acceder a cada dispositivo.

La figura 18 muestra en que posición en referencia al sistema OSI se encuentran los protocolos que vamos a analizar.

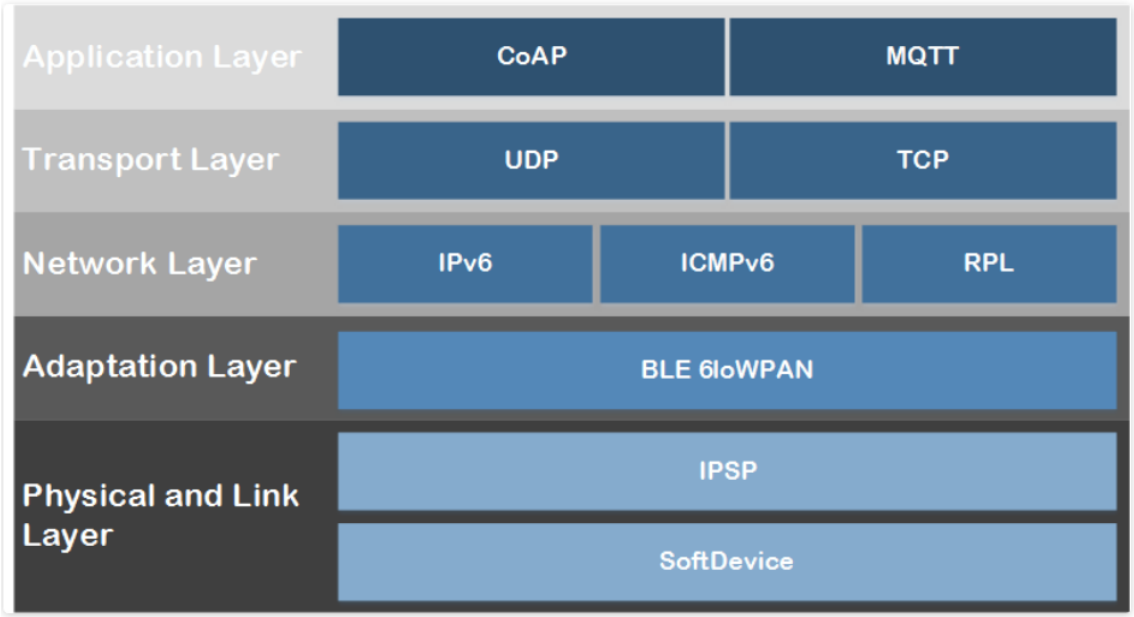


Figura 18. Sistema OSI de los protocolos CoAP y MQTT

3.5.1. CoAP

Es un protocolo cliente-servidor no estándar que fue lanzado a mitad del año 2013 por el grupo IETF.

El objetivo era crear un protocolo ligero y permitir una conexión permanente entre nodos y redes de baja potencia sobre dispositivos de baja capacidad.

CoAP se creó modificando el protocolo HTTP: se reduciendo las cabeceras y se añadió multicast, soporte de UDP y seguridad. Y como HTTP es un protocolo RESTful donde los recursos y los identificadores se tratan a través de una API. Además, se limita la fragmentación de mensajes para tener menor sobrecarga.

Distingue mensajes que requieren confirmación de los que no los necesitan. El intercambio de solicitudes y respuesta es asíncrono. Esto permite cubrir las necesidades de los dispositivos de bajo consumo, tener un funcionamiento con bajo ruido y una flexibilidad de comunicación con varios dispositivos.

Un nodo cliente puede comandar a otro nodo vía un paquete CoAP y el servidor CoAP lo interpretará. Es decir, el nodo extrae la información del paquete, y decidirá qué acción realizar.

Aparecen las subscripciones bajo demanda para monitorizar recursos mediante publicación y suscripción.

El URI se utiliza para que el cliente pueda descubrir recursos.
La seguridad se logra por construirse sobre el DTLS para lograr la integridad y la confidencialidad de los mensajes.

La codificación binaria del código reduce la sobrecarga de la red, aunque para solucionar problemas de red se necesita un analizador de protocolo.

El código de respuesta marca la capacidad de copiar en caché las respuestas de CoAP, no como en HTTP que depende del método de solicitud.

La figura 19 muestra cual es el mecanismo que implementa CoAP.

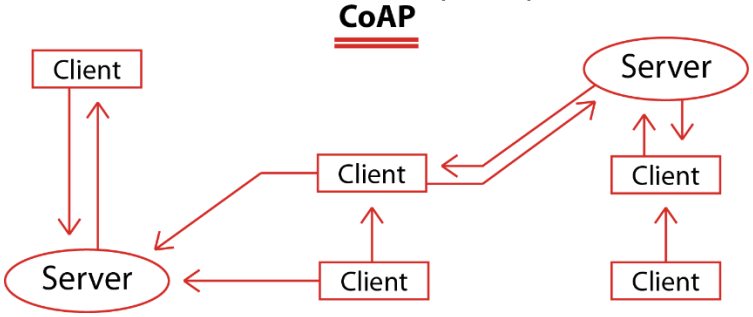


Figura 19. Mecanismo del protocolo CoAP

3.5.2. MQTT

El protocolo de *transporte de telemetría de cola de mensajes* (MQTT) es de código abierto con el fin de conectar dispositivos empotrados. Fue creado en el 1999 y estandarizado cuatro años más tarde por OASIS.

El sistema utilizado es de publicación-suscripción. Al utilizar el protocolo TCP aplica un reconocimiento de sesión continuo. La comunicación entre los clientes y el bróker es bidireccional con acuses de recibo.

Esta optimizado para dispositivos con bajos requerimientos a nivel de ancho de banda y de consumo energético. Además, está enfocado a redes poco confiables donde el software liviano debe garantizar cierta seguridad y confiabilidad.

La topología es en estrella con el bróker en el centro para gestionar los mensajes que se intercambia con los “clientes”. Ejemplos de bróker son Mosquitto o RabbutMQ. Incluye el software NodeRed para instalar en los nodos para comunicarse con el servidor broker.

La [Figura 20] muestra cual es el mecanismo que implementa MQTT.

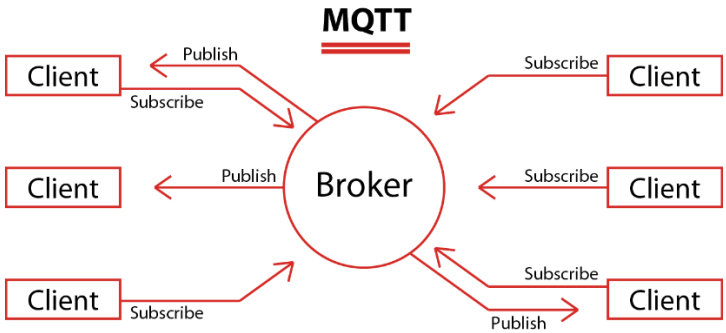


Figura 20. Mecanismo del protocolo MQTT

Un topic, equivalente al asunto en un correo electrónico, es el tema donde se suscriben los clientes para recibir los mensajes y al que se envían los mensajes.

La arquitectura se basa en eventos donde cada mensaje lo recibe solo aquellos nodos que se hayan suscrito a una publicación concreta. El emisor no sabe a quién dirige el mensaje, para ello está el broker, que lo reenviará a aquellos que están suscritos.

Los clientes son dispositivos que publican etiquetas, mensajes cortos, y/o se suscriben a un topic. El formato de los mensajes, generalmente cortos, es libre y abarca cualquier tipo de información, desde la temperatura a la presión. Los mensajes MQTT son asíncronos y su estructura es:

- Encabezado fijo (2 bytes) obligatorio en todos los mensajes.
- Encabezado variable (4 bits) opcional
- Mensaje o carga útil / payload (ideal <256MB; real <2 o 4 kB)

La especificación deja libertad al programador para decidir como implementar ciertos parámetros según sus necesidades.

- El tipo y el tamaño máximo.
- QoS. Hay tres niveles de calidad de servicio. Aunque a mayor calidad menor rendimiento.
    - QoS 0: como máximo una vez. puede que no se entregue.
    - QoS 1: al menos una vez. garantiza la entrega, pero puede que duplicados.
    - QoS 2: exactamente una vez. garantiza que llegará una vez el mensaje.
- Dos soluciones de seguridad que añaden carga a la red. Por un lado, tenemos la opción de incluir usuario y contraseña en los datos enviados. Y por otro soporta cifrado mediante SSL.

Para ver en más detalle cómo se implementa este protocolo podemos comparar la [Figura 21] con el siguiente código:

- /editorial maker/electrónica/placas desarrollo/arduino
- /casa/+/temperatura: el + sustituye cualquier nivel
- /casa/#: el # suscribe todo lo que hay debajo

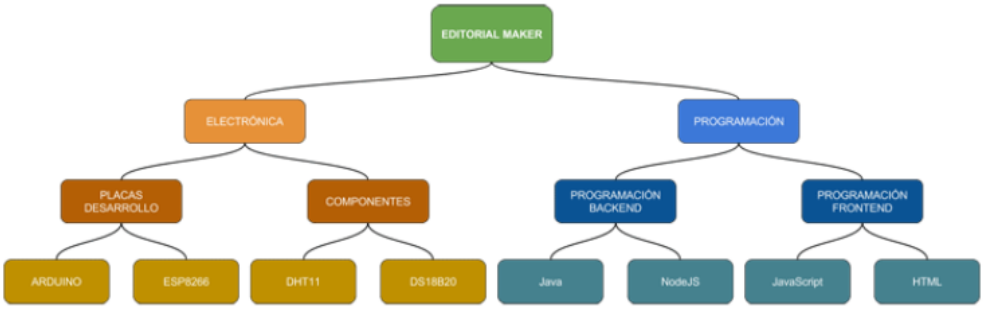


Figura 21. Ejemplo de estructura en arbol de una red MQTT

Procedimiento de conexión:

- Cada cliente MQTT abre una conexión permanente TCP con el Broker.
- Para instalar MQTT en una RBP se necesita un servidor Eclipse Mosquitto. Este broker es el mediador de mensajes que incluye el protocolo MQTT para que esté en constante disponibilidad. Los pasos a seguir son:

    1. Fijar una misma IP.
    2. Instalar Broker MQTT y un cliente MQTT en Raspberry Pi
    3. Crear *topics, mosquitto_sub y mosquitto_pub.*
    5. Instalar librerías para cargar en las SoC los clientes MQTT.

La diferencia con HTTP es que no es petición/respuesta.

Hay varios puntos negativos. No se permite la transferencia de dispositivo a dispositivo ni realizar *multicast.* Las aplicaciones que usan MQTT suelen ser relativamente lentas.

A continuación, veremos en la tabla 6 donde se compara ambos protocolos.

Características	MQTT	CoAP
Protocolo base	TCP	UDP
Modelo usado para la	Publicación-Suscripción	Pregunta-Respuesta Publicación-Suscripción
Nodo de comunicación	M:N	1:1
Consumo de energía	Mayor que CoAP	Menor que MQTT
Fiabilidad	3 Niveles de calidad de servicio QoS 0: Entrega no garantizada QoS 1: Confirmación de entrega QoS 2: Doble confirmación de entrega	Mensajes confirmables Mensajes no confirmables ACKs Retransmisiones
Número de tipos de mensaje usados	16	4
RESTful	No	Sí
Tamaño cabecera	2 Bytes	4 Bytes
Tipos de mensaje	Asíncrono	Asíncrono & Síncrono
Implementación	Fácil de implementar Complejo para añadir extensiones	Pocas librerías existentes y soporte
Seguridad	No definida	DTLS o IPSec
	Puede utilizar TLS/SSL	
Otros	Útil para conexiones en localizaciones remotas Sin gestión delerror	Baja saturación Baja latencia Problemas en NAT

Tabla 6. Comparativa de los protocolos de comunicacion

3.6. Cloud Computing

Hay muchas empresas que ofrecen servicios de computación en la nube. Con cloug computing en concreto nos referimos al conjunto de servicios que se implementa, despliegan y se usan a través de internet y que sustituyen total o parcialmente a los servidores insitu tradicionales.

Las herramientas ofrecidas han permitido la aparición de startups que han aprovechado las ventajas que han introducido para desplegar servicios complejos bajo demanda y reducir tanto tiempos como costes de implementación, licencias o mantenimiento. Permite una rentabilidad mayor, unos gastos fijos, protección frente a imprevistos y accesibilidad.

Para nuestra solución nos centraremos en los que ofrecen servicios de BC en la nube. Estos servicios son altamente demandados y entre las empresas que lo ofrecen encontramos a Oracle, Microsoft, Amazon, IBM o Google.

3.6.1. Azure

La fortaleza que ofrece Microsoft es la integración con otras aplicaciones de Microsoft. Por ejemplo, disponer de Active Directory y W10 tienen numerosas ventajas, permite gestionar todo en remoto y evita tener un servidor *on premise* con el consiguiente ahorro de costes.

Al integrar en la nube soluciones como Azure Sharepoint, Enterprise Mobility Security o Office 365 entre otras permite ofrecer seguridad, administración, escalar, autogestión de datos, como de aplicaciones y en infraestructuras.

Plataformas como Azure Blockchain Workbench y Azure Blockchain Development Kit permiten conectar con otros servicios de Azure y facilitador la creación de aplicaciones BC para integrarlo en la anterior. Ethereum on Azure y Hyperledger Fabric on Azure para alojar tu red de BC pública y privada respectivamente. En este último caso aparece el AKS (Azure Kubernetes Service) donde se puede implementar y configurar una red Hyperledger Fabric en Azure.

3.6.2. Google Cloud

La plataforma que Google ofrece a sus clientes es la misma con los mismos servicios que utiliza internamente para sus desarrollos como YouTube o Google Maps.

Abarca numeroso servicios de todo tipo entre los que podemos destacar los siguientes:

- Compute Engine para tener una infraestructura como servicio de máquinas virtuales configurables.
- BigQuery para almacenar y analizar datos de forma escalable y segura.
- Cloud IoT Core para conectar, administrar y transferir datos procedentes de dispositivos de manera fácil y segura.

En particular hay integraciones de aplicaciones en la Google Cloud y BC con Ethereum como se observa en [57]. En dicha plataforma podemos encontrar los servicios necesarios para empezar a implementar redes públicas o privadas.

- Para las públicas tenemos el Ethereum Developer Kit [58].

- Para las privadas podemos utilizar la Hyperledger Fabric and Composer que es un entorno para desarrollar aplicaciones relativos a BC privados [59].

3.6.3. AWS

Por último, analizamos como Amazon proporciona sencillas y numerosas soluciones para implementar y desplegar proyectos muy variados.

Su catálogo de servicios BC cubre proyectos tanto de redes públicas como de redes privadas de código abierto, Ethereum y Hyperledger Fabric respectivamente, mediante la plataforma **Amazon Managed Blockchain (AMB)**.

AMB permite crear y administrar este tipo de redes mediante aplicaciones. Es autoescalable, es decir, cualquier miembro de la red puede agregar un nodo especificando su CPU y memoria según sea la carga necesaria. Facilita al desarrollador el no tener que ocuparse de la provisión el hardware y configurar la red además de estar pendiente de los cambios, agregar nuevos miembros o solicitudes de transacción.

Una vez en funcionamiento el servicio administra sus certificados, extrae métricas operacionales y por medio del API de votación de Managed Blockchain [Figura 22] permite votar para agregar o eliminar a miembros de su red. La herramienta Key Management Service (KMS) hace posible crear los certificados de su red que sirve básicamente para evitar la necesidad de crear un almacenamiento seguro de claves.

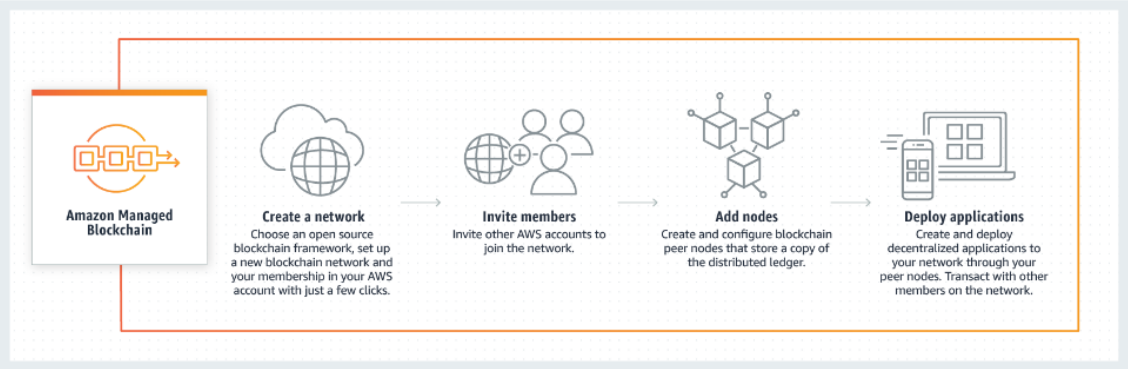


Figura 22. Esquema del funcionamiento de la AMB de Amazon

Con ayuda de la herramienta Amazon Quantum Ledger Database (QLDB), que no deja de ser una base de datos que alberga una copia inmutable, es posible replicar y analizar los datos externamente mediante el lanzamiento de consultas para observar tendencias y tomar decisiones. Por otro lado, añade fiabilidad por hospedar el historial de todas las transacciones de la red BC gracias al registro de cambios.

4. Selección

Una vez analizadas las distintas tecnologías en el capítulo anterior, en este partiremos de la figura 23 para explicar las razones por las que lo hemos elegido solución final.

Las tecnologías seleccionadas fueron elegidas teniendo en cuenta disponibilidad, comunidad de desarrollo, previsión de futuro y compatibilidad entre las distintas soluciones.

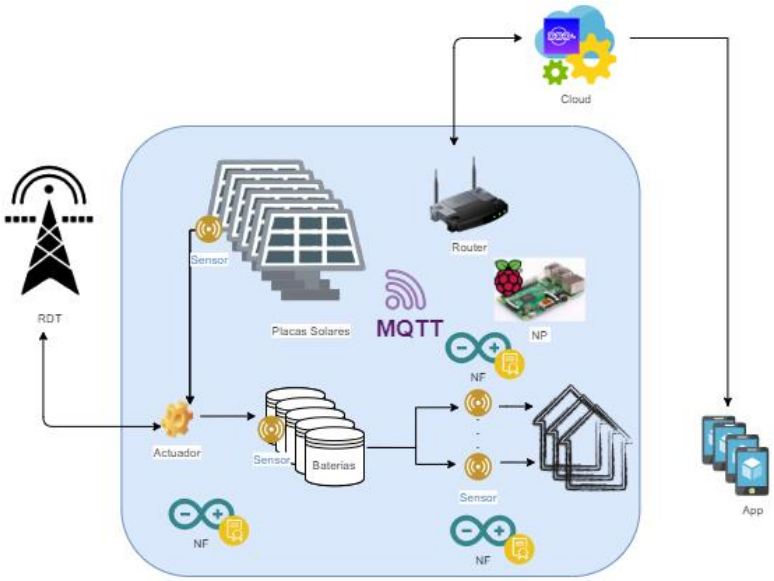


Figura 23. Estructura general de la solucion del proyecto

- La topología de red: árbol.

- Los dispositivos físicos a desplegar serán:
  - SoC: Arduino.
  - SBC: Raspberry Pi.

- Los dispositivos físicos no analizados a desplegar serán:
  - Sensores para extraer datos.
  - Actuadores para ejecutar las instrucciones.
  - Paneles solares monocristalinos fotovoltaicos
  - Inversores de autoconsumo para transformar la energía.
  - Baterías de litio con reguladores de carga.

- A nivel de software:
  - Comunicación: MQTT.
  - Cifrado: SHA256.
  - Tecnología publica: Ethereum.
  - Cloud Computing: Amazon AWS.
  - Algoritmo de consenso: PoS.
  - Smart Contract: Enerken.

4.1. Topología de red

Entre las topologías analizadas para la red descartamos de entrada la topología de estrella. Esta topología impide a los SoC comunicarse con el router directamente, sino que necesitan un nodo intermedio que es el SBC para ello.

La topología de malla también se descarta. Aunque es posible implementarla, en comparación con la topología en árbol sería más costoso desplegar varios router y SBCs para una red tan pequeña, que únicamente necesitaría desplegar una unidad de cada.

Por lo tanto, la topología elegida será de árbol [Figura 24].

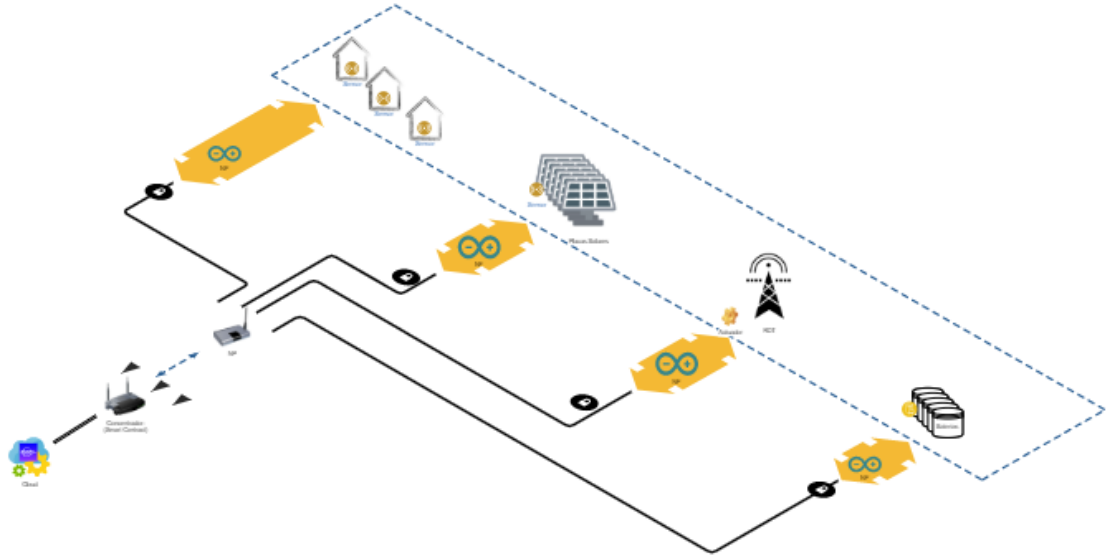


Figura 24. Topologia en arbol de la red del usuario

Como se observa en la figura, la estructura está compuesta por un router wifi, un único SBC y varios SoC de los que dependen los sensores y actuadores.

4.2. Dispositivos físicos analizados

Entre los dispositivos analizados en el rango de SoC y SBC elegimos Arduino y Raspberry Pi.

Las razones en ambos casos son similares:

- Comunidad. Son las más extendidas y con mayor comunidad por detrás.
- Futuro. Una comunidad fuerte asegura un futuro a medio-largo plazo.
- Simplicidad. La comunidad ofrece una larga lista de bibliotecas, tutoriales e IDEs que ayudan tanto para la implementación como en aparición de problemas.

4.3. Dispositivos físicos no analizados

Entre los dispositivos que componen la red, y no hemos analizado en profundidad, se encuentran los paneles solares, las baterías, los actuadores y los sensores.

Paneles solares. Se ha optado por una solución de paneles solares de tipo fotovoltaicos monocristalinos y de propiedad común a la comunidad de vecinos. Hay varios motivos:

- El espacio. Es posible que cada vecino tenga un balcón para alojarlo.
- La eficiencia. No todas las casas tienen acceso a la luz solar directa.
- El coste. Es más económico concentrar el despliegue de cableado, inversores de corriente y el número de sensores en un único punto. Esto reduce la longitud de metros de cables, mano de obra y cantidad de sensores e inversores. Las placas solares necesitan inversores de autoconsumo para transformar la energía de corriente continua (CC) en corriente alterna (CA) y un bucle de enganche de fase para sincronizar la fase y la frecuencia para poder utilizarla.

Las baterías, al igual que las placas solares y por motivos similares serán compartidas por toda la comunidad y situados en un único lugar. Al ser la parte más cara de la instalación, con una vida aproximada de 15 años, se elegirán las de tipo litio que son más rentables a largo plazo. A las baterías se le debe añadir un regulador de carga para controlar la corriente de entrada para así evitar que se sobrecargue.

Los sensores y los actuadores deben cumplir una serie de características como bajo consumo, coste económico, cobertura en entornos cerrados, ancho de banda baja, baja latencia y escalabilidad. El sensor elegido [Figura 25] es el pequeño transformador desmontable SCT-013. Sirve para realizar mediciones de corriente eléctrica y evita modificaciones intrusivas en los conductores que se desean monitorizar.

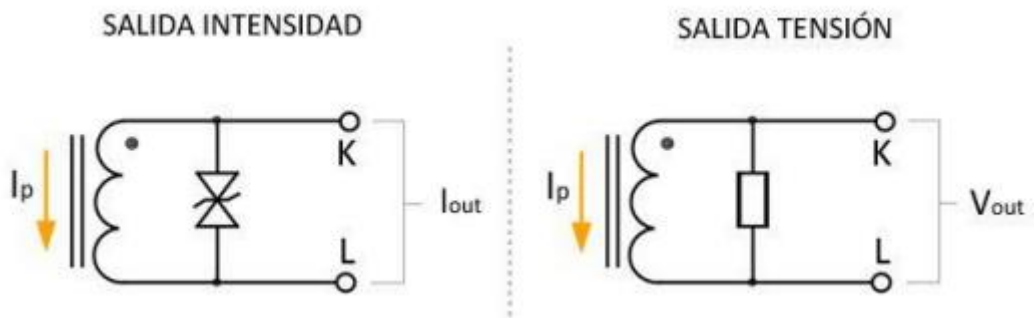


Figura 25. Funcionamiento del sensor de la red

4.4. Comunicación

Para nuestro proyecto necesitamos aquellas tecnologías de corto alcance ya que entorno objetivo es una urbanización. Por ello queda excluido las tecnologías LPWAN.

Entre las tecnologías de corto alcance para cubrir una red de sensores y dispositivos IoT, hemos analizado dos: CoAP y MQTT.

Para nuestro entorno hemos elegido MQTT como el protocolo porque cumple mejor las necesidades siguientes:

- La comunicación M:N. Teniendo en cuenta como hemos definido la red a nivel de topología el sistema suscriptores-publicadores es ideal. Aunque CoAP también es capaz de cumplir ese requisito, el hándicap es que está enfocado para redes 1:1.

- Calidad de servicio. MQTT permite la opción de configuración a nivel de calidad de servicio, CoAP no tiene lo tiene.

- Sencillez. A la hora de implementar la solución es muy asequible gracias a las numerosas librerías compatibles con el equipamiento elegido, Arduino y Raspberry Pi. En este aspecto CoAP no tiene tal potencia.

- Seguridad. Ofrece distinto niveles de seguridad con usuario y clave de acceso y de implementar cifrado TLS por ejemplo [60] [61].

Hay numerosos proyectos en internet donde es posible ver como una vez se despliega el mediador *Mosquitto* [62] que contiene el protocolo MQTT en la Raspberry, es posible implementa los protocolos TLS o SSL para añadir seguridad.

4.5. Blockchain Publica

Hyperledger Fabric está orientado a organizaciones que requieran una mayor privacidad mediante control de acceso. Está pensado para empresas que solo le interese compartir datos comerciales con su banco, sus clientes o sus proveedores, y no con su competencia.

En nuestro caso nos interesa entrar en una amplia comunidad sin restricción de acceso y no nos importa que otros usuarios sepan de nuestro negocio. Es más, nuestro objetivo es poder conectarnos a la máxima cantidad de usuarios: otras empresas, prosumidores o asociaciones de vecinos en general. Nuestro interés es tener mayor espectro de intercambio para abarcar no solo el mercado de energía sino cualquier otro servicio en el que emplear nuestros tokens. Por ello Hyperledge como Corda quedan excluidas.

La única red que permite utilizar el algoritmo de consenso PoS es Ethereum. Además, como hemos analizado está enfocado a redes de blockchain transparentes. Un ejemplo donde puede ser útil es en un ayuntamiento donde es el contribuyente esté interesado en comprobar la nómina de los concejales o los contratos de los servicios públicos como el servicio de limpieza o de alumbrado.

4.5.1. Cifrado

Entre los cifrados analizados es el Secure Hash Algorithms (SHA) el que se empleará ya que es el utilizado por la nube de Amazon.

4.5.2. Algoritmo de consenso

Para asentar las bases para un consumo responsable y comprometido con el medio ambiente evitaremos utilizar el algoritmo de minado que como hemos visto anteriormente necesita grandes recursos computacionales lo que implica una gran cantidad de energía.

Por ello descartamos el PoW y utilizaremos el algoritmo Power of Stake, que en vez de minar crea consensos entre los nodos de la red.

4.5.3. Ethereum, Smart Contract y Tokens

La energía es una mercancía, pero actualmente está limitada y es accesible únicamente a un mercado de grandes empresas. Igual que pasaba hace años con la bolsa. Al igual que entonces si se logra tokenizar la energía, como pasó con las acciones, se hace accesible a pequeños inversores, democratizando el sector. Además, permite aumentar el mercado, la competencia y la circulación.

Partimos de que Ethereum no es una criptomoneda sino una plataforma descentralizada que permite crear aplicaciones distribuidas o DAPPs a través de la ejecución de los *smart contract*. Y una criptomoneda es dinero digital distribuido en blockchain descentralizado. La moneda digital que se usa en la plataforma Ethereum es el Ether.

A diferencia de una criptomoneda, como puede ser el Ether o el Bitcoin que tienen su propia blockchain, un token está basado en la blockchain de una criptomoneda ya existente. Es decir, por ejemplo, EOS es un token existente que opera encima de la blockchain de Ethereum. Hoy en día hay según la web Eidoo más de mil tokens diferentes basado en ERC20 y circulado por Ethereum. EOS, Tron o Binance Coin son las más famosas.

Un *Token* en la blockchain representa un activo digital susceptible de ser comercializado, es decir, es la representación de un bien material o inmaterial. Un símil seria las fichas de un casino. Dichas fichas simbolizan dinero fiduciario para usar únicamente en los distintos juegos de azar del propio casino, pero no fuera del recinto. O por ejemplo las acciones de una empresa en bolsa que es

un título valor que los accionistas pueden intercambiar en el mercado bursátil y representa una participación en la compañía (se emiten en el mercado primario y después se negocian en el mercado secundario). Pueden representar desde objetos físicos como el oro (Digix), como intangibles en el futuro como acciones o bonos.

A partir de la creación de Ethereum han aparecido diversos tokens donde el funcionamiento lo define únicamente el programador. Entre los aspectos a programar se encuentra como va a ser la oferta inicial de monedas, como serán repartidas, el nombre, el símbolo…todo esto se define en el smart contract.

El token ERC20 (Ethereum Request for Comments) se diferencia de otros tokens en que se ajusta completamente al token estándar de Ethereum. Es un subconjunto de tokens en Ethereum que utiliza un conjunto de reglas básicas, funciones y eventos para la creación, distribución e intercambio de los tokens. De facto es tomado como el estándar, la referencia para el resto, lo que le convierte en el modelo más utilizado en dicha plataforma para crear tokens propios e independientes.

Así pues, si el programador al crear un nuevo token integra todas las funciones del estándar ERC20, conseguirá una serie de ventajas. Las principales son:

- Desarrollo sencillo y rápido. No necesita grandes conocimientos de programación por lo tanto su implementación no lleva mucho tiempo.
- Se encuentran apoyadas por una infraestructura ya creada, Ethereum, en vez de implementar otra plataforma completamente nueva.
- Una perfecta interacción con otros smart contract.
- Esta desvinculado del valor del Ether.
- Cometer menos errores al implementarlo porque define todas las reglas a seguir en la infraestructura de Ethereum.
- Posibilidad de trabajar con Dapps que usan el estándar ERC20.
- Ser fácilmente intercambiable.
- Interactuar con otros tokens que implemente el mismo estándar.

Además, los tokens son rastreables en la blockchain, al igual que el Ether o el Bitcoin, porque los smart contract de estos tokens son desarrollados sobre la cadena de bloques Ethereum.

El desarrollador es capaz de predecir la forma de interactuar y como se transferirán su token. Los tokens que no usan todas las funciones del ERC20, se consideran parcialmente compatibles.

La Máquina Virtual Ethereum (EVM) esta desplegado en miles de ordenadores, que llamamos nodos, conectados a través de internet y que forman la red de Ethereum. Si lo vemos como si fuera una “única” maquina podemos afirmar que es el lugar donde se ejecutan todos *smart contract.* Es decir, todos los cálculos compartidos que describen la distribución y los movimientos de los tokens.
El Gas es la unidad para medir el trabajo realizado en Ethereum. Por ejemplo, si un smart contract necesita ejecutar cinco instrucciones diferentes tendrá coste

computacional en Gas menor que otro contrato de diez instrucciones. Al estar separado del valor del Ether se mantiene estable y ofrece estabilidad al sistema.

El desarrollador debe incorporar un conjunto de funciones de los estándares de ERC20 en su contrato inteligente para cumplirlo. En concreto debe cumplir nueve métodos y dos eventos.

Los métodos son los siguientes:

- *Name* (opcional): Nombre del token.
  - `function name() public view returns (string)`
- *Symbol* (opcional): Símbolo del token.
  - `function symbol() public view returns (string)`
- *Decimals* (opcional): El número de decimales que utiliza el token.
  - `function decimals() public view returns (uint8)`
- *TotalSupply*: Suministro total de tokens que existirán.
  - `function totalSupply() constant returns (uint256 totalSupply)`
- *BalanceOf*: Saldo de la cuenta del propietario.
  - `function balanceOf(address _owner) constant returns (uint256 balance)`
- *Transfer*: Transferencia a…
  - `function transfer(address _to, uint256 _value) returns (bool success)`
- *TransferFrom*: Transferencia desde…
  - `function transferFrom(address _from, address _to, uint256 _value) returns (bool success)`
- *Approve*: Permite la retirada de fondos.
  - `function approve(address _spender, uint256 _value) returns (bool success)`
- *Allowance*: Devuelve la cantidad que se puede retirar.
  - `function allowance(address _owner, address _spender) constant returns (uint256 remaining)`

Los eventos son:

- *Transfer*: Activado cuando se transfieren los tokens.
  - `event Transfer(address indexed _from, address indexed _to, uint256 _value)`
- *Approval*: Activado siempre que se aprueba la transferencia.
  - `event Approval(address indexed _owner, address indexed _spender, uint256 _value)`

Además de las funciones estándar es posible crear personalizadas para habilitar más funcionalidades:

- Compra y venta automática: vinculado al valor de otro token.
- Recarga automática: de Gas para pagar a los mineros sus transacciones.
- Agregar una ‘casa de moneda’ central: para cambiar el número de tokens en circulación.
- Tokens des/congelados: de un usuario cuando un organismo regulador te lo indique.
- Proof-of-Work: para ejecutar una «minería fusionada» con Ethereum.

4.6. Cloud Computing

Entre todas las soluciones estudiadas vemos que Amazon es quien ofrece un modelo más versátil, escalable y económico. Además, es el mayor proveedor del mundo actualmente para alojar aplicaciones blockchain.

A continuación, describiremos los tres grandes módulos elegidos para cubrir las necesidades de la solución.

El módulo AWS IoT, descrito anteriormente, permite conectar de forma bidireccional y segura los dispositivos (sensores, actuadores, SoC, SBC) a la nube AWS. Mediante submódulos recopila, almacena y analiza todos los datos de telemetría y permite configurar y mantener la red de dispositivos IoT.

- AWS IoT Core [figura 26]: es un servicio administrado que permite realizar un seguimiento y conectar a los dispositivos mediante el protocolo de comunicación ligero MQTT que, como comentamos, está especialmente diseñado para tolerar conexiones intermitentes, minimizar la huella de código en los dispositivos y reducir los requisitos de ancho de banda de la red.

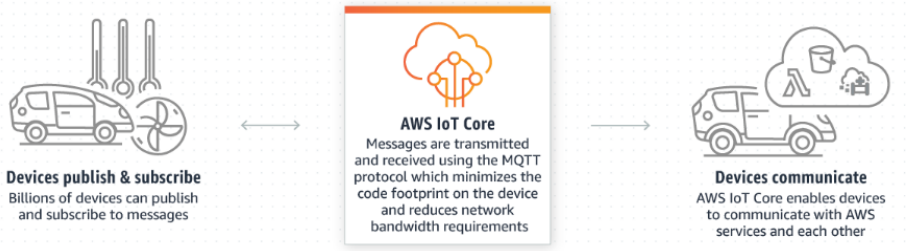


Figura 26. Ejemplo de AWS IoT Core

- AWS IoT Device Management [figura 27]: facilita las tareas de registrar, organizar, monitorizar y administrar de forma remota dispositivos IoT a escala.

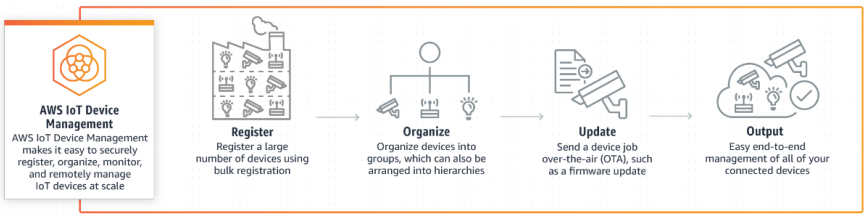


Figura 27. Ejemplo de AWS IoT Device Management

- AWS IoT Events [figura 28]: es un servicio administrado que monitoriza continuamente los datos proveniente de sensores compatibles con IoT que envían datos de telemetría; facilita las tareas de detección de eventos y activa la respuesta automática a estos eventos mediante instrucciones simples como “if-then-else”

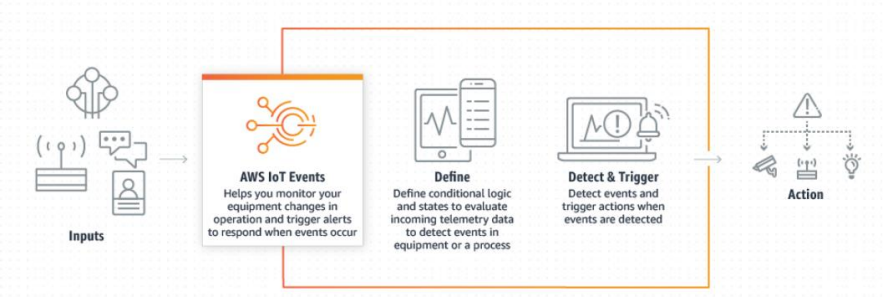


Figura 28. Ejemplo de AWS IoT Events

El módulo Amazon SageMaker [figura 29] ofrece un servicio administrado con la capacidad de crear, entrenar e implementar modelos de aprendizaje automático. Es decir, ofrece la posibilidad de implementar Machine Learning. Y el módulo Amazon Augmented AI (Amazon A2I):es un servicio que facilita la creación de los flujos de trabajo necesarios para que las personas revisen las predicciones del aprendizaje automático e intervengan cuando un modelo no es capaz de realizar predicciones de alta confiabilidad.

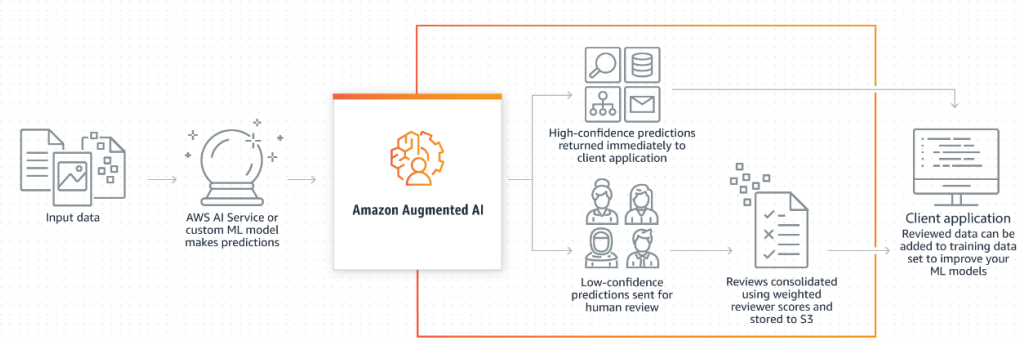


Figura 29. Ejemplo del módulo Amazon SageMaker

El módulo Amazon Managed Blockchain (AMB) [figura 30]: es un servicio administrado que facilita la creación y administración de redes de blockchain escalables mediante el uso de los marcos de código abierto como Ethereum. Configurar el funcionamiento de las políticas analizadas anteriormente, el servicio ajusta su escala automáticamente.

La tarea de administrar y mantener los certificados realiza un seguimiento de las métricas operacionales como, y permite replicar una copia inmutable de su actividad de red de cadenas de bloques a Amazon Quantum Ledger Database (QLDB), una base de datos de libro mayor completamente administrada. Esto le permite analizar fácilmente la actividad de la red de manera externa y obtener información acerca de las tendencias.

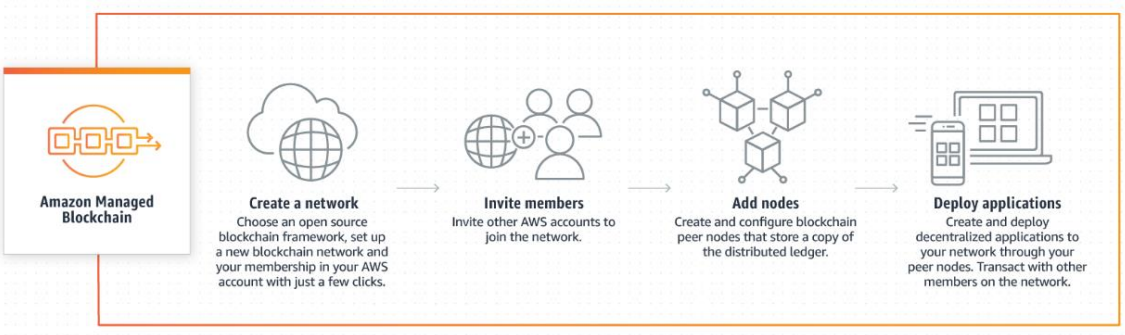


Figura 30. Ejemplo de Amazon Managed Blockchain

5. Modelo

Como se explicó anteriormente vamos a describir cómo utilizar BlockChain para generar un mercado distribuido secundario de energía donde cualquier usuario de la red Ethereum (BC) sea capaz de participar adquiriendo energía u obteniendo una recompensa por la energía generada.

Los clientes y proveedores serán únicamente los usuarios de la red de Ethereum [figura 31] con los que poder pagar mediante el token Enerken según unas directivas dadas al aplicativo en la nube.

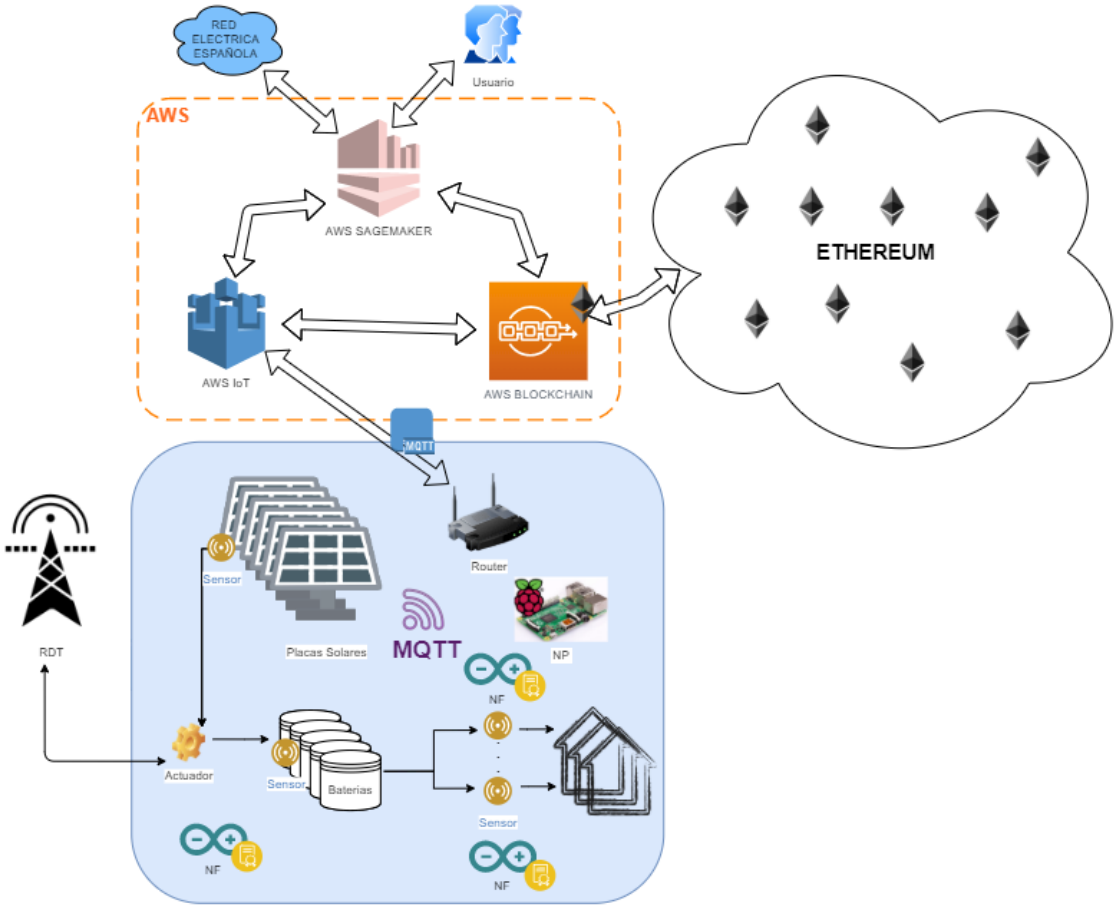


Figura 31. Red del proyecto

En este capítulo se describirá la solución elegida y se realizará una selección a distintos niveles:

- Tecnologías.
- Protocolos.
- Algoritmos.
- Comunicación.
- Blockchain.

5.1. Funcionamiento: sensores

Primeramente, veremos cómo extraer los datos que necesitamos para implementar la solución.

La estratégica distribución de los sensores [Figura 32] servirá para obtener la telemetría de la energía generada, consumida y transmitida.

El mercado de componentes electrónicos y de desarrollo software actualmente ofrece una variada oferta y con precios asequibles. Esto permite un rápido ciclo de adopción.

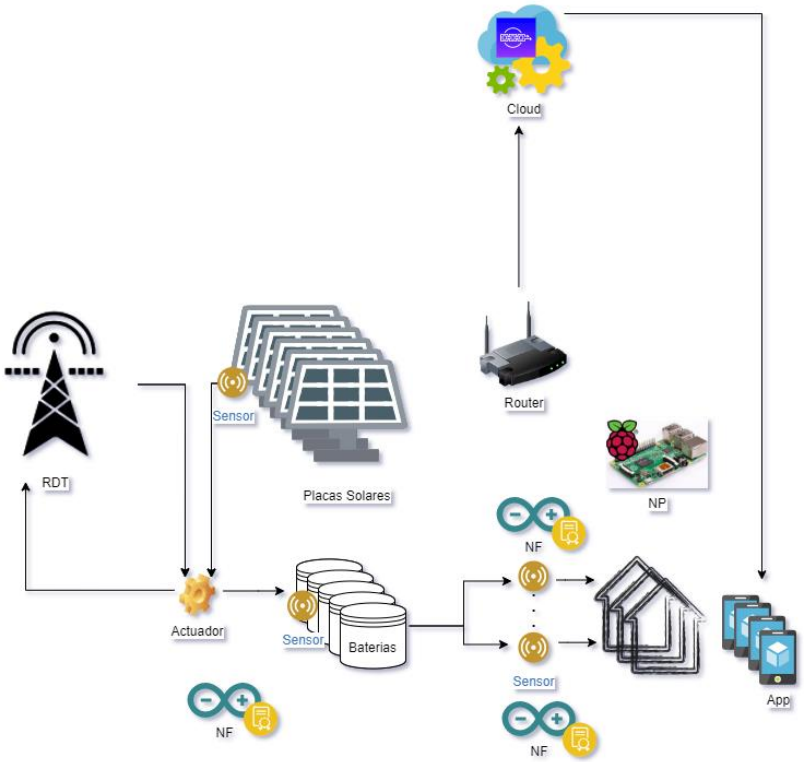


Figura 32. Distribución de los sensores en la red

Para realizar la lectura de los consumos en las baterías, en las placas solares y en casa de cada usuario utilizaremos el mismo sensor [Figura 33] de corriente no intrusivo:

Figura 33. Detalle del sensor de corriente

5.1.1. Baterías

Los NF se interesan por el estado de las baterías [Figura 34].

- Si están por debajo de un 20% porcentaje, lanza la petición de comprar electricidad.
- En caso contrario no hacer nada.

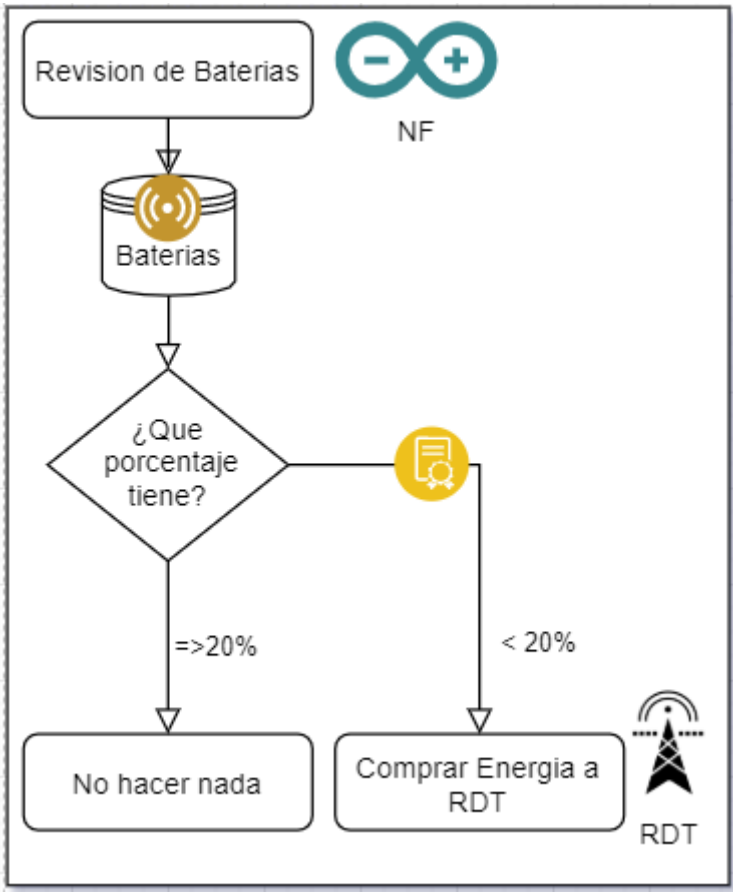


Figura 34. Modelo de decisión del funcionamiento de las baterías

5.1.2. Placas solares

Los sensores se instalarán:

- En las placas solares.
- En las baterías.
- A la entrada de la casa de cada usuario de la red.

Una vez obtenido la energía como se observa en la **¡Error! No se encuentra el origen de la referencia.**35:

1- El sensor preguntará al NF que hacer.
2- El NF preguntará al sensor de las baterías si están llenas:
   a. Si lo está, contactará con el NP para realizar el intercambio con la RDT energía por tokens.
   b. Si no lo está, deberá consultar con el SBC si venderla al RDT o almacenarlo según las políticas prefijadas.
3- En el caso de venderla se firmará un smart contract directamente con el NF y lanzará la petición de tarea al actuador para realizar el intercambio.

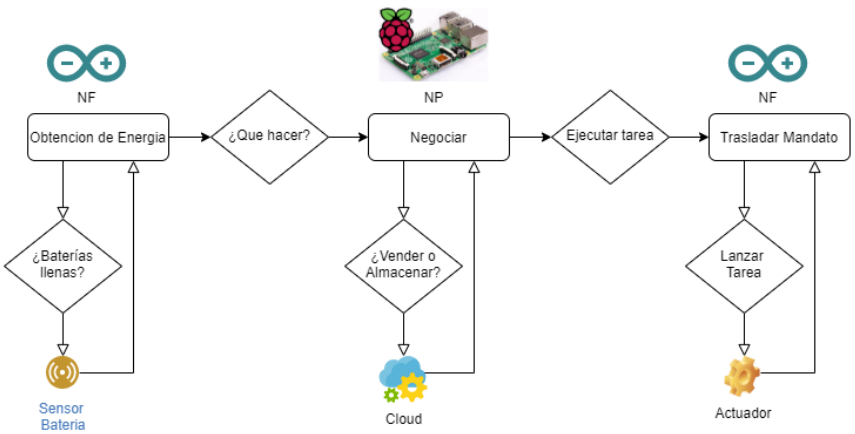


Figura 35. Modelo de decisión del funcionamiento de las placas solares

5.1.3. Usuario

Los sensores situados a la entrada de la red eléctrica privada de cada usuario realizaran varias tareas [Figura 37].

Cada vez que el usuario consuma energía contactara con el NF para registrar el consumo. A posteriori el NF irá publicando la información que le llegará al bróker situado en el NP que posteriormente mandará al Cloud para registrar los consumos de cada cliente y firmar el contrato inteligente cada vez.

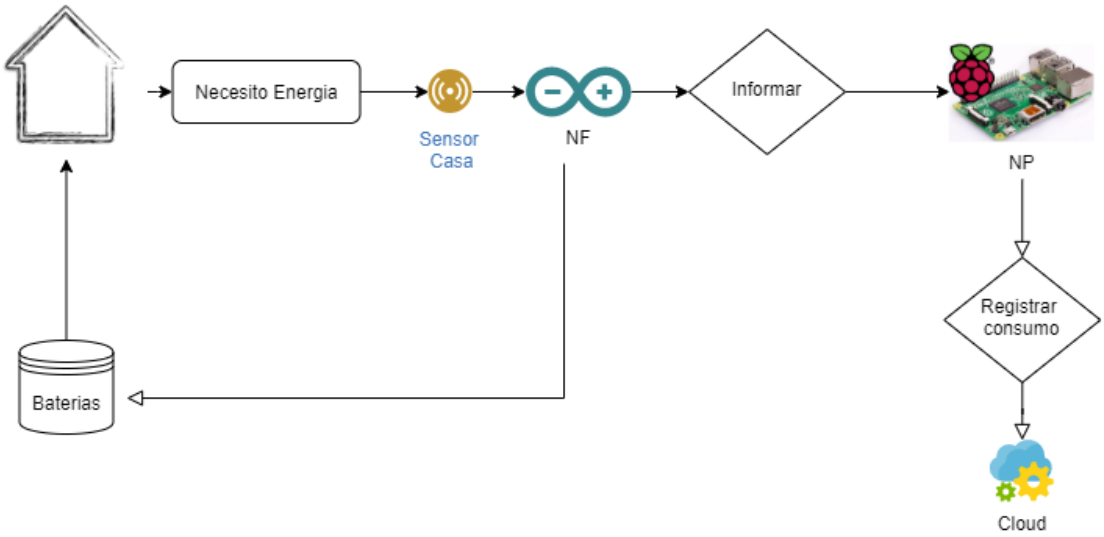


Figura 36 Modelo de decisión del funcionamiento del usuario final

5.1.4. Actuador

Básicamente es un interruptor al que se le indicará por medio del NF dejar pasar la corriente por el cable entre la RDT y las baterías o entre las placas solares y el RDT:

- Si se compra electricidad abrirá el circuito que comunica la RDT y las baterías para su recarga.
- Si se vende electricidad abrirá el circuito de salida desde las placas solares a la RDT.

5.2. Funcionamiento: cloud

En este apartado desarrollaremos, como vimos anteriormente, la utilización de tres grandes módulos que proporciona la nube de Amazon [Figura 37].

En la siguiente figura presentamos como se implementará la cloud y como será los flujos de comunicación que presentaremos en los siguientes apartados.

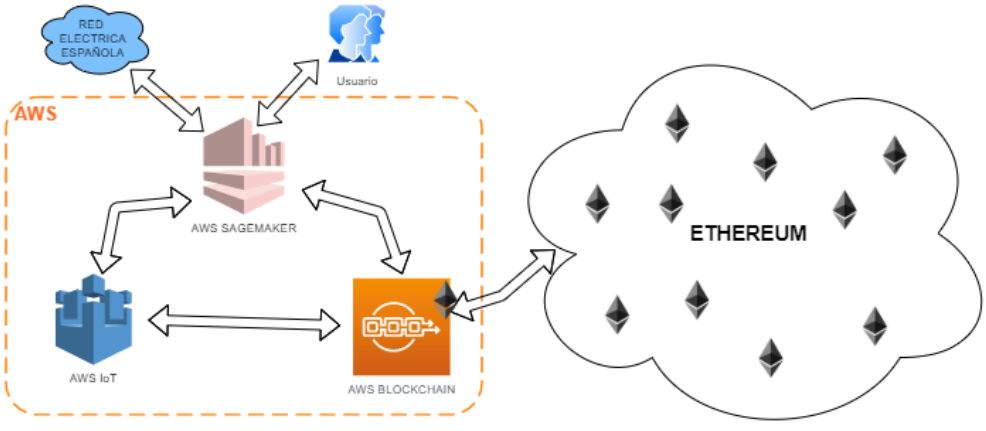


Figura 37. Relación de la nube con Ethereum, web REE y usuario final

5.2.1. AWS IoT

El módulo AWS IoT [figura 38] está conectado simultáneamente con el router de la red de la sede, con el módulo AWS SageMaker y con el módulo AWS BlockChain.

Amazon ofrece dentro del módulo distintas funciones que divide en submódulos.

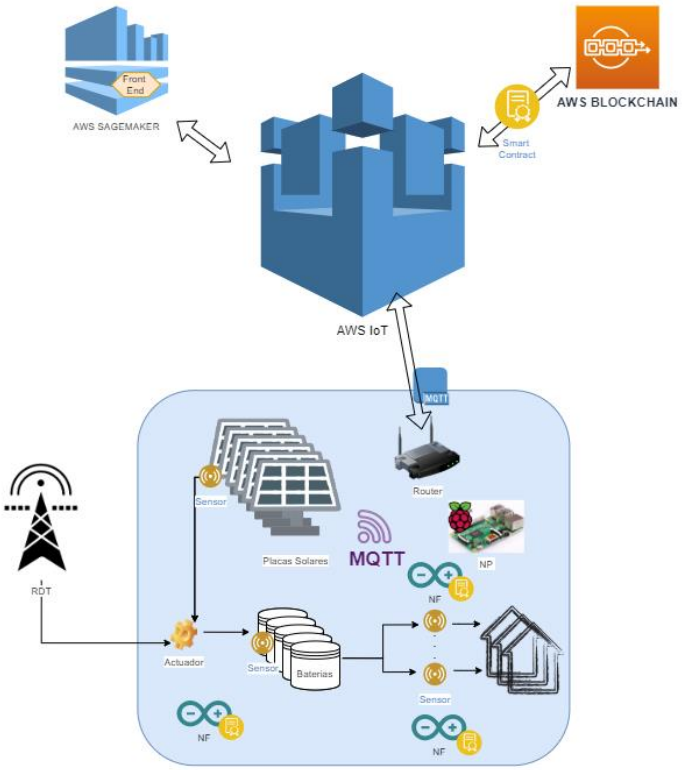


Figura 38. Modulo AWS IoT

Primeramente, utilizaremos el submodulo AWS IoT Core.

Este servicio permite registrar la RBP como si fuera un objeto más en la nube AWS. Ofrece las herramientas para crear un certificado para autenticarse junto a las políticas asociadas. Permite añadir seguridad mediante conexiones TLS. Facilita crear una serie de reglas donde se definen las consultas y acciones a realizar cada vez que se recibe un mensaje MQTT. Estos mensajes son los topics del protocolo de comunicación MQTT.

La implementación de este protocolo seguiría la lógica de la [Figura 39] y los topics se catalogarían de la siguiente manera:

- /Raspberry Pi/ARDUINO 1/Sensor Casa *x*
- /Raspberry Pi/ARDUINO 2/Sensor Placa Solar *x*
- /Raspberry Pi/ARDUINO 3/Sensor Bateria *x*
- /Raspberry Pi/ARDUINO 4/Sensor RTB *x*
- /Raspberry Pi/ARDUINO 4/Actuador RTB *x*

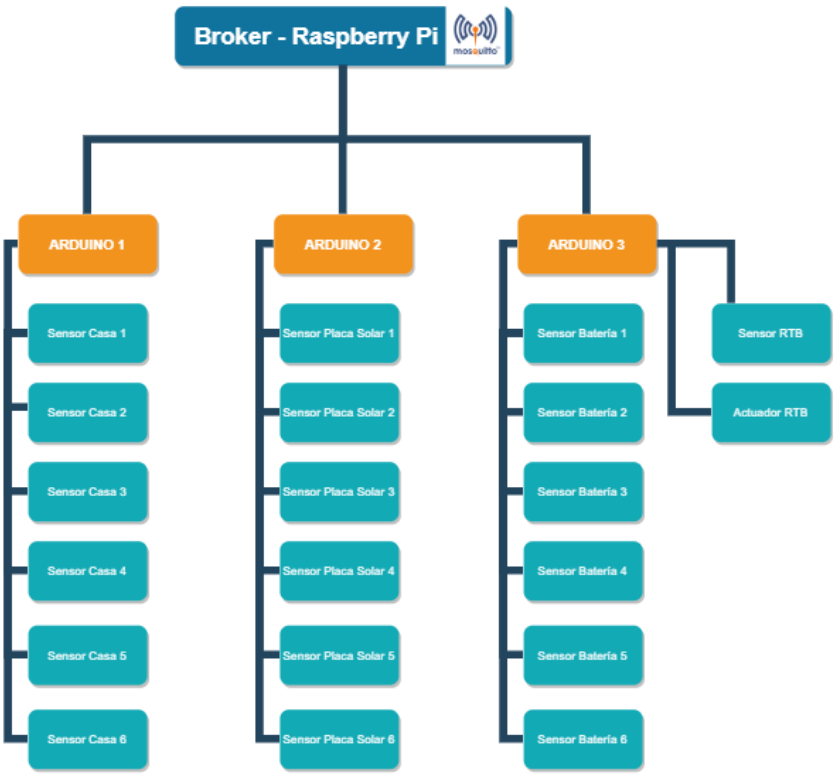


Figura 39. Implementacion del protocolo MQTT

Una vez conectada la red IoT con seguridad TLS y definido el protocolo de comunicación MQTT es necesario utilizar el servicio AWS IoT Device Management para registrar, organizar, monitorizar y administrar de forma remota todos los dispositivos IoT que tenemos desplegados en la red.

Registraremos los dispositivos jerárquicamente como indicamos a continuación y a medida que crezca la red se podrá fácilmente incorporar nuevos sensores o actuadores:

- Sensores Casa
- Sensor Placa Solar
- Sensor Batería
- Sensor RTB
- Actuador RTB.

Las otras funciones que ofrece AWS IoT Device Management, una vez registrados y organizados, son las de monitorizar y administrar. Esto es posible mediante operaciones remotas como actualizaciones de software o reinicio del dispositivo. Permiten realizarlas tanto en un único dispositivo como en todos a la vez y a la vez recibir en tiempo real el estado de dichos trabajos a medida que se implementan en los dispositivos.

El siguiente paso es obtener los datos de telemetría mediante el servicio AWS IoT Device. Este servicio monitoriza continuamente los datos proveniente de sensores. Los objetivos son detectar de eventos y en caso necesario activar una respuesta automática. En nuestro caso lo utilizaremos para enviar todos los datos al módulo Amazon SageMaker.

5.2.2. Amazon SageMaker

El módulo Amazon SageMaker es un servicio administrado que permite crear, entrenar e implementar modelos de aprendizaje automático de forma rápida. Una vez entendida su potencia lo utilizaremos como el cerebro de todas las operaciones, tanto de la parte de IoT como de la parte la parte blockchain. Será la piedra angular y donde reside toda la inteligencia de la solución.

Las tareas las descomponemos en tres partes:

- Recepción de datos.
- Análisis de datos.
- Ejecutar las tareas.

En la figura 40 observamos como está relacionado con las diferentes partes de la solución que desgranaremos en los siguientes párrafos.

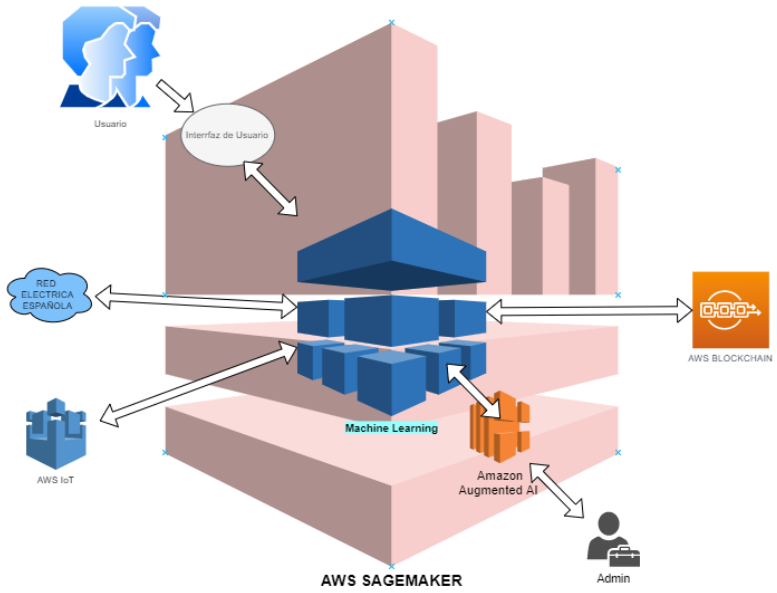


Figura 40. Modulo AWS SageMaker

El módulo SageMaker recibirá datos desde dos procedencias, una interna y otra externa. Desde el interior de la solución recibirá datos del módulo AWS IoT, desde donde recopilará todos los datos de telemetría obtenidos de los usuarios. El medio externo es la web de Red Eléctrica Española, [https://www.esios.ree.es/,](https://www.esios.ree.es/) donde obtendremos la información publicada de la operación del sistema eléctrico español sobre los costes de la electricidad en tiempo real.

Una vez en nuestro poder de los datos necesarios, el siguiente paso es analizar los datos en conjunto, hacer previsiones del consumo de nuestra red y utilizar las herramientas de machine learning y de IA para crear modelos de toma de decisión.

Todo este flujo de trabajo lo analizaremos mediante la herramienta Amazon Augmented AI que nos permitirá observar, revisar y en caso de ser necesario modificar los modelos creados por el machine learning.

En este módulo implementaremos toda las funciones explicadas anteriormente:

- Implementar un frontend para que los usuarios finales interactúen y en la parte dinámica de javascript, utilizar una librería que se llama web3 para interactuar con el Smart contract
- La correlación energía-token estará relacionada con la energía generada mediante un modelo económico y una función que fluctuará dependiendo de la oferta del mercado y de la demanda de los usuarios.
- El intercambio con los usuarios y las descritas en el punto 5.1 sobre el funcionamiento del usuario final [figura 41], de las baterías [figura 42], de las placas solares y los actuadores.

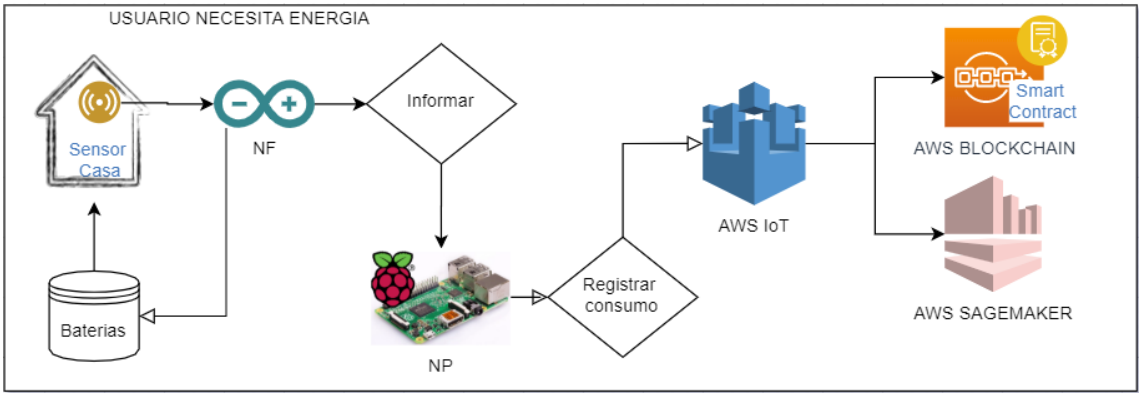


Figura 41 Modelo de decisión: funcionamiento del usuario final

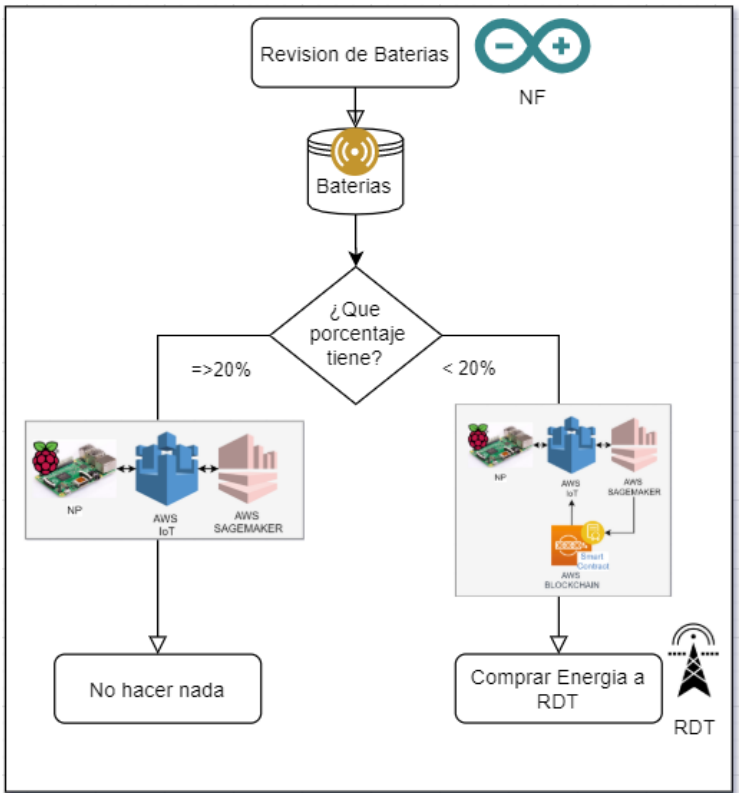


Figura 42. Modelo de decisión del funcionamiento de las baterías

5.2.3. Amazon Managed Blockchain

Por último, el módulo Amazon Managed Blockchain [figura 43] lo utilizaremos para implementar la parte de blockchain de la solución.

En el módulo AMB es donde se implementa los diagramas de decisión y todas las funciones de los *smart contracts* y se configura el despliegue de un nodo de la red de Ethereum en la Raspberry Pi.

El propio modulo te proporciona el nodo Ethereum creado y la gestión

1- Crearemos un nodo Ethereum. Se debe completar la plantilla de AWS CloudFormation para lanzar el nodo de la red de Ethereum publica usando la plataforma docker-local en una única instancia de Amazon EC2.
2- Configuraremos el *smart contract.* Veremos el detalle en el siguiente apartado.

Los usuarios de AWS pueden crear su nodo lanzando instancias que ejecutan la red Ethereum, y que les permite elegir entre implementar una instancia EC2 público o privado.

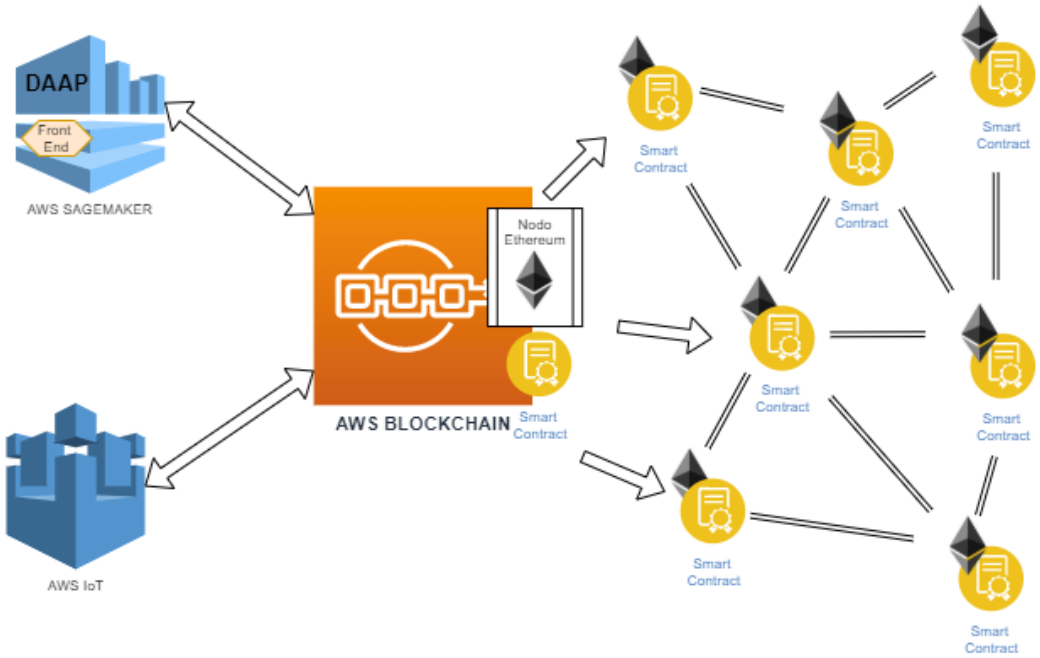


Figura 43. Modulo AWS Blockchain

5.3. Funcionamiento: smart contract

Una vez tengamos la infraestructura implementada, debemos desarrollar la lógica de negocio en un *smart contract*.

Para este proyecto únicamente utilizaremos un único smart contract para cargar todas las funciones necesarias. Este SC está basado en el estándar ERC20 para hacerlo compatible con la mayoría de las aplicaciones de la red Ethereum.

Las dos gestiones fundamentales que debe cubrir nuestro SC son:

- Gestionar la energía. Para registrar todo lo relacionado con la energía:
  - Se ha producido un kilovatio.
  - Se ha consumido un kilovatio.
  - Relación entre 1kw y 1 Enerken.

- Gestionar los tokens. Para transferir tokens cuando se compre o se venda electricidad, consultar balances, etc...

Las anteriores descripciones se traducen en el siguiente código:

- Definición del token.
  - Nombre: *enerken.*
  - Cantidad emitida: un billón de unidades de *enerken.*
  - La relación: 1W equivale a 1 *enerken*.
  - El código del *smart contract* [figura 44]

```
string public symbol = ENK;
string public  name = enerken;
uint8 public decimals = 18;
uint _totalSupply = 1000000000;
```

Figura 44. Definición del Token

- Comprar de electricidad a la red RDT
  - El código del *smart contract* [figura 45].

```
function _mint(address account, uint256 amount) internal {
   require(account != 0);
   _totalSupply = _totalSupply.add(amount);
   _balances[account] = _balances[account].add(amount);
   emit Transfer(address(0), account, amount);
 }
```

Figura 45. Compra de electricidad

- Vender electricidad a un usuario o a la RDT.
  - El código del *smart contract* [figura 46].

```
function transfer(address to, uint tokens) public returns (bool success) {
    balances[msg.sender] = balances[msg.sender].sub(tokens);
    balances[to] = balances[to].add(tokens);
    emit Transfer(msg.sender, to, tokens);
    return true;
}
```

Figura 46. Venta de electricidad

- Regalar un kilovatio.
    - El código del smart contract [figura 47].

```
// Regalar energy
   function () public giveable(address spender, uint256 addedValue) public returns (bool)  {
    require(spender != address(0));

    _allowed[msg.sender][spender] = (_allowed[msg.sender][spender].add(addedValue));
    emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
    return true;
  }
```

Figura 47. Regalar electricidad

- Consultar balance.
    - El código del smart contract [figura 48].

```
function balanceOf(address tokenOwner) public view returns (uint balance) {
    return balances[tokenOwner];
}
```

Figura 48. Consultar el balance de tokens

Así pues, como acabamos de ver con las funciones descritas podríamos realizar tareas de compra, venta, consulta o donación de electricidad. Un ejemplo de funcionamiento de este smart contract según la lógica implementada en el aplicativo seria en caso de sobrante y venta de electricidad:

1- Registrar la venta de 1KW que se he generado.
2- Realizar la transferencia de 1 tokens hacia dicho usuario.

6. Conclusiones

En **este proyecto** se ha desarrollado un estudio sobre el despliegue de energías renovables dentro de una urbanización donde se despliega una red de dispositivos IoT. Mediante una aplicación en la nube, a la que se aplica *machine learning*, se gestiona de manera automática el intercambio de energía eléctrica a partir de un *smart contract* en la red de blockchain pública Ethereum. De esta forma conseguimos *tokenizar* la energía.

El **blockchain permite la *tockenizacion*** de la energía. Esto significa la liberación del mercado que hasta ahora ha sido copado por las grandes eléctricas. Es un caso parecido a la creación de acciones en la bolsa que democratizó en parte el sector financiero. La potencia que ofrece este cambio es fundamental para el devenir del mercado eléctrico donde los precios a pagar ya no dependerán únicamente de un oligopolio centralista, sino que permite la incorporación de los pequeños prosumidores.

Los **beneficios al usuario final** son numerosos. Con la *tokenizacion* de su energía consigue ser dueño de la energía que genera, la independencia energética y obtiene una herramienta para negociar sus tarifas. Además, contribuye a reducir la huella ecológica contribuyendo a una sociedad más ecológica y al *New Green Deal.*

El **coste de la implementación de esta red** a nivel de usuario final no es muy elevado. Como hemos visto los dispositivos físicos a desplegar, a parte de las baterías y las placas solares que serán amortizados en pocos años, como los SoC, el SBC, el actuador y los sensores son de bajo costo. La parte software utiliza una red pública como es Ethereum que es gratuita y contratar parte de soluciones en la nube a Amazon tiene un coste mensual reducido. Estas dos puntos no exige un gran desembolso para una comunidad de vecinos si lo comparamos con la instalación de un gimnasio o la contratación de un servicio personal de seguridad con horario 24/7.

Futuros desafíos. Con este trabajo hemos logrado convertir únicamente una parte de toda la red eléctrica en inteligente y para futuras líneas de desarrollo de este proyecto se podrían considerar el despliegue de tecnologías como blockchain, IoT y machine learning al resto de etapas. Es decir, que las TIC llegasen a las centrales generadoras, las líneas de transporte, las centrales de transmisión, las redes de reparto y las líneas de distribución. Algunas medidas necesarias:

- Proponer un acuerdo entre la administración pública, los prosumidores y las compañías eléctricas para incorporar sensores a lo largo de la red eléctrica para monitorizarla y desplegar tecnologías para obtener energía renovable a lo largo de la red con el fin de eliminar las energías fósiles.

- Priorización la energía según el tipo de suministro: mediante el marcado de priorización a incluir en la cabecera de los paquetes de energía. Esto se utilizará para que llegue la energía eléctrica a un hospital antes que al alumbrado público en caso de emergencia.

- Método de transmisión: para monitorizar los datos es necesario trasladar la información al centro de datos. Para ello planteo tres alternativas: crear una red de fibra óptica con el alto gasto que conlleva; mediante un sistema sin cables con la limitación de interferencias, ruidos, espectro y velocidades que tiene; o reutilizar el propio cable eléctrico de cobre con las limitaciones de velocidades, espectro y alcance.

- Métricas: crear un sistema que mida el nivel de calidad, la eficiencia y la cantidad de energía eléctrica que se consume.

Otras líneas de mejoras es implementar la red con otras tecnologías u otros protocolos como las propuestas en el proyecto que no fueron seleccionadas. También sería posible incluir otras tecnologías como las redes de carga de los coches eléctricos. Incluso sería viable contemplar proyectos más ambiciosos a nivel local, estatal e incluso internacional.

7. Glosario

- 3GPP: 3rd generation partnership project. Es una colaboración de grupos de asociaciones de telecomunicaciones con el objetivo de asentar especificaciones de tercera generación.
- Amazon Managed Blockchain: AMB. Plataforma de Amazon sobre la que se despliega proyectos de BC.
- Amazon SageMaker. Modulo de AWS
- API: App Programming Interface. Es la interfaz con la que se permite programar las aplicaciones.
- Arduino: Plataforma abierta para realizar proyectos electrónicos de código abierto.
- AWS IoT Core. Submódulo de AWS IoT que permite registrar SBC, ofrece las herramientas para crear un certificado, añade seguridad e implementa la lógica de MQTT.
- AWS IoT Device Management. Submódulo de AWS IoT que registrar, organizar, monitorizar y administrar de forma remota todos los dispositivos IoT que tenemos desplegados en la red
- AWS IoT Device. Submódulo de AWS IoT que permite monitorizar la telemetría, detectar eventos y activar una respuesta automática.
- AWS IoT: Modulo de AWS que permite desplegar y gestionar dispositivos IoT mediante el uso de submódulos.
- AWS: Amazon Web Services. Plataforma de Amazon implementada en la nube y ofrece numerosos servicios integrales a nivel global.
- Big Data: este sistema permite procesar de forma masiva datos en un sistema distribuido en tiempo real.
- BlockChain: Cadena de Bloques. Es una red tipo peer-to-peer que mediante unos algoritmos introduce seguridad a las transacciones por internet de una manera abierta, descentralizada, consistente, certificada y transparente.
- Broker: nodo central en una red basada en el protocolo MQTT.
- Bytecode: es el tipo de instrucciones que entiende la EVM.
- Cloud Computing: La computación en la nube referencia a la tecnología que posibilita la capacidad computacional y el uso de multitud de servicios, archivos e información desde Internet alojado en la nube.
- CoAP: Constrained Application Protocol. Es un protocolo de comunicación cliente-servidor.
- DTLS: Datagram Transport Layer Security. Es un protocolo que proporciona privacidad en las comunicaciones para protocolos de datagramas y es utilizado por CoAP.
- Edge Computing: la computación en el borde permite acercar el procesamiento y tratamiento de los datos recopilados en el extremos de la red.
- EPC: Evolved Packet Core: Protocolo para el intercambio de paquetes que utiliza NB-IoT.
- Ethereum. Red BC publica que utiliza SC para sus transacciones.

- Hash: es un algoritmo de cifrado o de encriptación que utiliza una clave de cifrado para transmitir información privada públicamente por internet y evitar que otros accedan al mensaje fácilmente.
- Hyperledge. Red BC privada que utiliza SC para sus transacciones.
- Hyperledger Fabric es una plataforma de código abierto para crear programas que funcionen en la plataforma de blockchain Hyperledger.
- IDE: Integrated Development Environment. Es un entorno de desarrollo integrado compuesto por un lenguaje de programación, las herramientas para transmitir código y un bootloader.
- IoE; Internet of Energy. Subgrupo de IoT aplicado a la energía.
- IoT: Internet of Things. Es una de las principales tecnologías habilitantes de la SC y ha sido palanca para el desarrollo de nuevos modelos de negocio, denominado Industria 4.0.
- LoraWan: protocolo estándar abierto encuadrado en las tecnologías LPWAN.
- LPWAN: Low Power WAN. tecnologías que ofrecen conectividades en redes de áreas extensas.
- Machine Learning
- Máquina virtual de Ethereum: EVM. Alojado en cada nodo de la red de Ethereum para implementar y ejecutar los SC.
- Mosquitto: es un tipo de broker que se utiliza para implementar MQTT.
- MQTT: Message Queue Telemetry Transport. Es un protocolo de comunicación publicación-suscripción.
- NB-IoT: protocolo estándar abierto encuadrado en las tecnologías LPWAN.
- NodeRed: es un software que se instala en los nodos de una red MQTT para comunicarse con el broker.
- Nodo Primarios: NP. Dispositivos SBC que al poseer un microprocesador permite ejecutar tareas pesadas de procesamiento.
- Nodos Primarios: NP. Dispositivos SoC optimizados para automatizar tareas e interactuar con los sensores y los actuadores.
- Proof of Stake: PoS. Algoritmo de consenso que utiliza las validaciones de los nodos para determinar si un bloque es o no correcto.
- Proof of Work: PoW. Algoritmo de consenso que utiliza la prueba de trabajo para determinar si un bloque es o no correcto.
- Prosumidor: Individuo que produce el mismo bien o servicio que consume.
- Raspberry Pi: RBP. Dispositivos SBC.
- RDT: Red de distribución y Transporte.
- Remix: es el IDE para Solidity.
- RF-TDMA: Random Frecuency – Time Division Multiple Access: protocolo utilizado por SigFox para sus comunicaciones.
- SBC: Single Board Computer. Dispositivos formados fundamentalmente por la memoria, entradas/salidas y un microprocesador.
- SEP Sistema de Potencia Eléctrica: engloba las instalaciones y los equipos para generar, transportar y distribuir la energía eléctrica.
- SigFox: protocolo propietario encuadrado en las tecnologías LPWAN.
- Smart Buildings: mediante la introducción de las nuevas tecnologías la idea es eficientar el consumo, integrar un sistema de control y

automatización de sistemas como la seguridad, la iluminación o la climatización.

- Smart City: Ciudad Inteligente. Se consigue modificando los dispositivos de cualquier ámbito actuales e introduciendo nuevos para obtener datos que ayuden a automatizar las ciudades.
- Smart Contract: SC. Es el Código que crea un contrato por el que se ejecuta la aplicación para el intercambio de transacciones en la red BC.
- Smart Devices: dispositivos conectados a internet que aportan datos a una plataforma que los utiliza para automatizar tareas y tomar decisiones.
- Smart Environment: mediante la introducción de las nuevas tecnologías se centra en los ámbitos de energía, agua, residuos y el medio ambiente.
- Smart Governance: es la nueva forma de dirigir una ciudad utilizando las nuevas tecnologías permitiendo a los ciudadanos que aporten ideas, información sobre eventos y participen en la toma de decisiones de manera digital.
- Smart Grid: Red Eléctrica Inteligente. Introduce sistema inteligente a la red tradicional y cambiar el paradigma del sistema actual.
- Smart Mobility: mediante la introducción de las nuevas tecnologías tiene el objetivo de gestionar el tráfico, el sistema de aparcamiento, reducir el impacto ambiental, mejorar la planificación de los medios de transporte y la movilidad de las personas.
- Smart Utilities: mediante la introducción de las nuevas tecnologías se focalizan en abaratar costes de operación y distribución del gas, el agua y la electricidad.
- SoC: System on a chip. Son dispositivos autómatas formados principalmente por un microcontrolador.
- Solidity: es el lenguaje sobre el que se programa los SC.
- Token: moneda electrónica que es utilizada para intercambiar bienes y servicios y funcionan por encima de una red de BlockChain.
- Topic: es el tema donde se suscriben los clientes para recibir los mensajes y al que se envían los mensajes en una red MQTT.

8. Bibliografía

[1] P. Pai., «Smart City Services - Challenges and Approach,» IEEE, 2019.

[2] W. Z. Hessam Moeini, "Toward Data Discovery in Dynamic Smart City Applicacions," IEEE, 2019.

[3] L. Wei, «Based On Big Data Technology Analysis On The Mode And Countermeasures Of Smart City Construction Operation Management,» IEEE, 2018.

[4] P. Sadhukhan, «An IoT based Framework for Smart City Services,» IEEE, 2018.

[5] B. Rusti, «Deploying Smart City components for 5G network slicing,» IEEE, 2018.

[6] Y. K. Tomov, «itcoin Evolution of Blockchain Technology,» IEEE, 2019.

[7] T. Clohessy, «Smart City as a Service (SCaaS): A Future Roadmap for E-Government Smart City Cloud Computing Initiatives,» IEEE, 2014.

[8] Q. Chen, «A Survey on an Emerging Area Deep Learning for Smart City Data,» IEEE, 2019.

[9] J. d. Andalucia. [En línea]. Available: https://www.andaluciasmart.andaluciaesdigital.es/moad. [Último acceso: 04 2020].

[10] B. S. Baños. [En línea]. Available: https://empresas.blogthinkbig.com/smartcity/. [Último acceso: 04 2020].

[11] D. F. d. Gipuzkoa. [En línea]. Available: https://www.gipuzkoa.eus/es/web/ekonomia/programas-y-ayudas/proyectos-estrategicos/smart-mobility-industry. [Último acceso: 04 2020].

[12] S. U. Summit. [En línea]. Available: https://www.smartutilitysummit.com/. [Último acceso: 04 2020].

[13] T. Crystal. [En línea]. Available: https://www.thecrystal.org/about/awards/. [Último acceso: 04 2020].

[14] U. d. Alicante. [En línea]. Available: https://web.ua.es/es/smart/smart-environment-un-entorno-de-calidad-de-vida.html. [Último acceso: 04 2020].

[15] P. Feng, «Big Data Analysis of E-Commerce Based on the Internet of Things,» IEEE, 2019.

[16] Y. L. a. P. P. E. Saksonov, «Structural Synthesis of the IoT System for the Fog Computing,» 2019.

[17] Alotaibi, «Utilizing Blockchain to Overcome Cyber Security Concerns in the Internet of Things: A Review,» 2019.

[18] IFTTT. [En línea]. Available: https://ifttt.com/. [Último acceso: 04 2020].

[19] K. H. a. H. Suzuki, «Cooperation Between Heterogeneous IoT Devices Using iHAC Hub,» IEEE, 2019.

[20] B. S. Baños. [En línea]. Available: https://empresas.blogthinkbig.com/que-es-iot/. [Último acceso: 04 2020].

[21] J. F. Mora, «Panorámica Histórica de la Ingeniería Eléctrica. El Sistema Eléctrico de Potencia,» 2016.

[22] O. Waxman, «https://time.com/5698700/current-war-real-history/,» *Time,* 2020.

[23] A. Gonzalez, Gestion de la energia en una red inteligente, 2012.

[24] S. N. Adel Nazemi, «Challenges and Opportunities of the Integration of IoT and Smart Grid in Iran Transmission Power System,» IEEE, 2017.

[25] M. L. Xu Xinhua, «The vision of smart grid 2.0,» IEEE, 2014.

[26] M. Y. Junwei Cao, «Energy Internet – Towards Smart Grid 2.0,» IEEE, 2013.

[27] J. M. Hossein Shahinzadeh, «Internet of Energy (IoE) in Smart Power Systems,» IEEE, 2019.

[28] J. M. Hossein Shahinzadeh, «Internet of Energy (IoE) in Smart Power Systems,» IEEE, 2019.

[29] A. A. M. a. R. F. M. F. Lima, «IoT Energy Retrofit and the Connection of Legacy Machines Inside the Industry 4.0 Concept,» IEEE, 2019.

[30] A. N. R. S. K. a. S. A. M. S. Gayathri, «Battery Condition Prognostic System using IoT in Smart Microgrids,» IEEE, 2018.

[31] U. Europea. [En línea]. Available: https://ec.europa.eu/energy/topics/technology-and-innovation/energy-storage/batteries_en#set-plan-action. [Último acceso: 04 2020].

[32] O. A. V. R. a. D. M. I. H. Gong, «Real Time Operation of Smart Homes with PV and Battery Systems under Variable Electricity Rate Schedules and Transactive Power Flow,» IEEE, 2018.

[33] L. P. a. V. R. K. Nalinaksh, «An Internet of Things solution for real- time identification of electricity theft and power outrages caused by fault in distribution systems,» IEEE, 2018.

[34] C. D. C. G. F. a. S. C. M. R. Morello, «A Smart Power Meter to Monitor Energy Flow in Smart Grids: The Role of Advanced Sensing and IoT in the Electric Grid of the Future,» IEEE, 2017.

[35] Q. W. Y. T. a. M. N. Z. Liu, «The Real-Time Co-Simulation Platform with Hardware-in-Loop for Cyber-Attack in Smart Grid,» IEEE, 2018.

[36] Y. K. Tomov, «Bitcoin: Evolution of Blockchain Technology,» IEEE, 2019.

[37] H. T. T. H. Junfeng Xie, «A Survey of Blockchain Technology Applied to Smart Cities: Research Issues and Challenges,» IEEE, 2019.

[38] D. Bowden, «Towards Secure and Smart Healthcare in Smart Cities Using Blockchain,» IEEE, 2018.

[39] E. S. K. J. G. S. a. J. W. J. S. J. Pee, «Blockchain based smart energy trading platform using smart contract,» IEEE, 2019.

[40] A. R. Julija Golosova, «Review of the Blockchain Technology in the Energy Sector,» IEEE, 2019.

[41] Ramón Gallart Fernández. [En línea]. Available: https://www.smartgridsinfo.es/comunicaciones/redes-inteligentes-entornos-rurales. [Último acceso: 05 2020].

[42] U. N. W. A. A. M. Zafar, «Applications of ZigBee in Smart Grid Environment: a reviewR.,» IEEE, 2015.

[43] [En línea]. Available: https://aprendiendoarduino.wordpress.com/2016/12/11/que-es-arduino-2/. [Último acceso: 05 2020].

[44] L. Llamas. [En línea]. Available: https://www.luisllamas.es/comparativa-esp8266-esp32/. [Último acceso: 06 2020].

[45] B. Santana. [En línea]. Available: https://es.ign.com/raspberry-pi/153948/feature/raspberry-pi-4-para-que-sirve-y-que-podemos-hacer-con-el. [Último acceso: 06 2020].

[46] Ionos. [En línea]. Available: https://www.ionos.es/digitalguide/servidores/know-how/un-vistazo-a-proyectos-basados-en-raspberry-pi/. [Último acceso: 06 2020].

[47] A. Rosic. [En línea]. Available: https://blockgeeks.com/guides/proof-of-work-vs-proof-of-stake/. [Último acceso: 05 2020].

[48] [En línea]. Available: https://mundotokens.com/proof-of-work/. [Último acceso: 05 2020].

[49] [En línea]. Available: https://academy.bit2me.com/que-es-proof-of-work-pow/#Caracteristicas_del_protocolo_PoW. [Último acceso: 05 2020].

[50] [En línea]. Available: https://academy.bit2me.com/que-es-proof-of-stake-pos/. [Último acceso: 05 2020].

[51] [En línea]. Available: https://academy.bit2me.com/que-es-una-dao/. [Último acceso: 05 2020].

[52] [En línea]. Available: https://www.miethereum.com/smart-contracts/solidity/. [Último acceso: 05 2020].

[53] A. Aguilar. [En línea]. Available: https://medium.com/everis-blockchain/protocolo-de-votaciones-en-ethereum-53c1c816331d. [Último acceso: 05 2020].

[54] [En línea]. Available: https://wiki.hyperledger.org/display/INTERN/Project+Plan%3A+Raspberry+Pi+Indy+Agent. [Último acceso: 05 2020].

[55] P. S. Martin Valenta. [En línea]. Available: http://explore-ip.com/2017_Comparison-of-Ethereum-Hyperledger-Corda.pdf. [Último acceso: 05 2020].

[56] J. S. Miranda, «http://openaccess.uoc.edu/webapps/o2/bitstream/10609/94946/6/jsaizmiTFM0619memoria.pdf,» 2019.

[57] G. Cloud. [En línea]. Available: https://cloud.google.com/blog/products/data-analytics/building-hybrid-blockchain-cloud-applications-with-ethereum-and-google-cloud. [Último acceso: 05 2020].

[58] G. E. Kit. [En línea]. Available: https://console.cloud.google.com/marketplace/details/techlatest-public/ethereum-developer-kit?filter=category%3Adeveloper-stacks&id=f2f1c909-e1b9-4573-8afd-097e0ced1653. [Último acceso: 05 2020].

[59] G. Hyperledge. [En línea]. Available: https://console.cloud.google.com/marketplace/details/click-to-deploy-images/hyperledger-fabric-and-composer. [Último acceso: 05 2020].

[60] S. e. Raspberry. [En línea]. Available: https://dzone.com/articles/mqtt-security-securing-a-mosquitto-server. [Último acceso: 05 2020].

[61] Steve. [En línea]. Available: http://www.steves-internet-guide.com/mosquitto-tls/. [Último acceso: 05 2020].

[62] S. ELECTRONICS. [En línea]. Available: https://www.digikey.es/es/maker/projects/send-and-receive-messages-to-your-iot-devices-using-mqtt/39ed5690cc46473abe8904c8f960341f. [Último acceso: 05 2020].

9. Anexos

9.1. Smart Contract

Parte 1/2

```
pragma solidity ^0.4.21;
import "./EIP20Interface.sol";
import "../../math/SafeMath.sol";

contract ERC20Token is ERC20Interface {

    using SafeMath for uint; //use our safe math library
    uint256 constant private MAX_UINT256 = 2**256 - 1;
    mapping(address => uint) balances;
    mapping(address => mapping(address => uint)) allowed;

    string public symbol = ENK;
    string public  name = enerken;
    uint8 public decimals = 18;
    uint _totalSupply = 1000000000;

    constructor (
        uint256 _initialAmount,
        string _tokenName,
        uint8 _decimalUnits,
        string _tokenSymbol

    ) public {
        balances[msg.sender] = _initialAmount;
        _totalSupply = _initialAmount;
        name = _tokenName;
        decimals = _decimalUnits;
        symbol = _tokenSymbol;
    }

    function totalSupply() public view returns (uint) {
        return _totalSupply;
    }

    function balanceOf(address tokenOwner) public view returns (uint balance) {
        return balances[tokenOwner];
    }

   function transfer(address to, uint tokens) public returns (bool success) {
        balances[msg.sender] = balances[msg.sender].sub(tokens);
        balances[to] = balances[to].add(tokens);
        emit Transfer(msg.sender, to, tokens);
        return true;
    }

    function approve(address spender, uint tokens) public returns (bool success) {
        allowed[msg.sender][spender] = tokens;
        emit Approval(msg.sender, spender, tokens);
        return true;
    }

    function transferFrom(address from, address to, uint tokens) public returns (bool success) {
        balances[from] = balances[from].sub(tokens);
        allowed[from][msg.sender] = allowed[from][msg.sender].sub(tokens);
        balances[to] = balances[to].add(tokens);
        emit Transfer(from, to, tokens);
        return true;
    }

    function allowance(address tokenOwner, address spender) public view returns (uint remaining) {
        return allowed[tokenOwner][spender];
    }
```

Parte 2/2

```
// Regalar energy
   function () public giveable(address spender, uint256 addedValue) public returns (bool)  {
    require(spender != address(0));

    _allowed[msg.sender][spender] = (_allowed[msg.sender][spender].add(addedValue));
    emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
    return true;
  }

// Fallback function. Don't accept ETH
   function () public payable {
       revert();
    }
// Aumentar la cantidad permitida de gasto de token
  function increaseAllowance(address spender, uint256 addedValue) public returns (bool)  {
    require(spender != address(0));

    _allowed[msg.sender][spender] = (_allowed[msg.sender][spender].add(addedValue));
    emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
    return true;
  }

// Decrementar la cantidad permitida de gasto de token
  function decreaseAllowance(address spender, uint256 subtractedValue) public returns (bool)  {
    require(spender != address(0));

    _allowed[msg.sender][spender] = (_allowed[msg.sender][spender].sub(subtractedValue));
    emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
    return true;
  }

// Crear la cantidad de token de una cuenta dada
 function _mint(address account, uint256 amount) internal {
    require(account != 0);
    _totalSupply = _totalSupply.add(amount);
    _balances[account] = _balances[account].add(amount);
    emit Transfer(address(0), account, amount);
  }

// Elimina la cantidad de token de una cuenta dada
 function _burn(address account, uint256 amount) internal {
    require(account != 0);
    require(amount <= _balances[account]);

    _totalSupply = _totalSupply.sub(amount);
    _balances[account] = _balances[account].sub(amount);
    emit Transfer(account, address(0), amount);
  }

// Elimina la cantidad de token de una cuenta dada
 function _burnFrom(address account, uint256 amount) internal {
    require(amount <= _allowed[account][msg.sender]);

    _allowed[account][msg.sender] = _allowed[account][msg.sender].sub(amount);
    _burn(account, amount);
  }

}
```