Estado del arte y diseño de los componentes del caso de uso infraestructuras cloud

MLEDGE - Aprendizaje automático en la nube y en el borde

(Cloud and Edge Machine Learning)

Junio de 2024

Información sobre el entregable

Nombre del documento: Estado del arte y diseño de los componentes del caso de uso de infraestructuras cloud

Versión actual: 1.0

Proyecto: MLEDGE - Aprendizaje automático en la nube y en el borde (Cloud and Edge Machine Learning)

Paquete de trabajo: P5 - Implementación del caso de uso de infraestructuras cloud

Tareas: El entregable es resultado del trabajo en los diversos componentes técnicos:
- A5.1: Diseño de los componentes del caso de uso de infraestructuras cloud

Entregable: E5.1 – Estado del arte y diseño de los componentes de infraestructuras cloud

Autores: Iker Ceballos Rodríguez (Acuratio), David Márquez Mínguez (Acuratio)

Revisores: Santiago Andrés Azcoitia (IMDEA), Nikolaos Laoutaris (IMDEA)

Historial de Versiones

Versión	Fecha	Resumen de modificaciones
Version 1.0	30-06-2024	Versión inicial del documento

Índice

Índice 3
Introducción 4
1. Definición del problema y objetivos 6
1.1. Contexto 6
1.2. Objetivos 8
1.3. Situación actual 8
1.4. La solución 10
1.5. Casos de uso 12
2. Especificación de requisitos 14
2.1. Metodología 14
2.2. Requerimientos funcionales 14
2.3. Requerimientos no funcionales 17
3. Arquitectura de la plataforma FLaaS 18
3.1. Descripción de la arquitectura 18
4. Diseño detallado de la solución 20
4.1. Capa de Ejecución 20
4.2. Capa de Comunicaciones 21
4.3. Capa de Coordinación 21
4.4. Capa de Gobernanza 22
4.5. Capa de Costes 24
5. Diseño detallado de los demostradores 25
5.1. Caso de uso de optimización de costes cloud 25
5.2. Caso de uso de mejora de protocolos de FL 26

Introducción

En diciembre de 2022 fue adjudicado a IMDEA Networks el proyecto “MLEDGE - Aprendizaje automático en la nube y en el borde (Cloud and Edge Machine Learning)” (REGAGE22e00052829516, en adelante el ‘Proyecto’ o MLEDGE) por parte del Ministerio de Asuntos Económicos y Transformación Digital del Gobierno de España, con fondos de la Unión Europea dentro del Plan de Recuperación, Transformación y Resiliencia (European Union - NextGenerationEU/PRTR). El proyecto tiene como objetivo habilitar un ecosistema próspero de servicios FL en el borde seguros y eficientes capaces de facilitar el uso de datos personales y B2B confidenciales para entrenar modelos de ML para consumidores mientras se protege la privacidad de los datos y de sus propietarios.

Los **objetivos generales del proyecto** se pueden resumir en los siguientes:

1. Hacer del aprendizaje federado una funcionalidad accesible y de fácil uso en el borde mediante el desarrollo de una capa de software intermedio y componentes que escondan la complejidad del procesamiento y el intercambio de datos que supone.
2. Resolver problemas técnicos asociados al aprendizaje federado en el borde de la nube.
3. Demostrar esta funcionalidad en casos de uso que reflejen problemas reales de la industria que pueden ser resueltos con estas tecnologías.
4. Explotar los resultados del proyecto involucrando a agentes externos y comunicar los hallazgos al público potencial en general.

Uno de los objetivos básicos del proyecto es diseñar, implementar y hacer públicos demostradores que trabajen con datos sensibles de individuos, y alimenten modelos de aprendizaje automático en diferentes campos de la industria. A tal fin, en la primera parte del proyecto se ha realizado una selección de empresas para el desarrollo de la plataforma FLaaS y el monitoreo de costes de computación, así como el diseño e implementación de casos de uso de negocio reales que se beneficien del aprendizaje distribuido en el borde de la nube. La empresa Acuratio Europe SL. con NIF B75217612, en adelante Acuratio, resultó adjudicataria del paquete de trabajo P5 cuyo objetivo es el diseño y la implementación del caso de uso de infraestructuras cloud.

El presente documento se corresponde con el entregable E5.1 de título “Estado del arte y diseño de los componentes del caso de uso de infraestructuras cloud” y tiene como objetivo la documentación detallada, incluyendo la necesidad y el problema que requiere el caso de uso, su impacto y potencial explotación, el diseño de su funcionamiento, y las pruebas de sistema.

Acuratio fundada en 2017 es pionera en Federated Learning (FL), su equipo fue autor en 2018 de la primera implementación *open source* que hubo disponible de Federated Averaging. Acuratio ha sido pionera a nivel mundial en el campo del “Privacy Preserving Machine Learning” y el “Collaborative Machine Learning” que son dos de los grandes retos del Edge Machine Learning. Además, es un proveedor líder en las tecnologías habilitadoras de la privacidad.

Las soluciones federadas de Acuratio son actualmente utilizadas, entre otras, por las multinacionales financieras españolas más grandes. Habitualmente estas corporaciones tienen problemas para compartir datos con sus socios por temas regulatorios y de confidencialidad. Acuratio les proporciona distintas soluciones analíticas, como modelos avanzados de Machine Learning o analítica avanzada que permite clasificar grupos de clientes.

Por lo tanto, Acuratio está únicamente posicionado para diseñar, implementar y hacer públicos demostradores que trabajen con datos sensibles de individuos, y alimenten modelos útiles en áreas de la economía tradicional y de la economía digital como FinTech, identidad, salud, transporte, control de acceso, etc.

1. Definición del problema y objetivos

En este apartado se presenta el problema que trata de resolver el caso de uso de infraestructuras cloud de MLEDGE, comenzando por su contexto. Adicionalmente, se detalla la situación actual y el estado del arte, entendiendo como tal el de las soluciones de infraestructura tecnológica existentes similares a la que soportará los casos de uso de MLEDGE. Para un estado del arte más detallado de la investigación en materia de aprendizaje federado, se refiere al lector a la sección 3.2 del entregable E1.1 del proyecto.

1.1. Contexto

El acceso al dato es el mayor reto al que se enfrentan empresas, administraciones públicas y ciudadanos las próximas décadas. El entrenamiento de modelos de Machine Learning se ve beneficiado por el acceso a grandes cantidades de datos y especialmente de diversas fuentes. Tanto la confidencialidad que exigen las empresas, como la privacidad que demandan los ciudadanos hacen que el "compartir" datos cada vez sea más complejo.

Los gobiernos han escuchado estas preocupaciones y cada vez hay más países en el mundo con regulaciones en materia de protección de datos, algunas de las cuales se resumen en la siguiente figura.

Ilustración 1: Regulaciones y normativas de datos y privacidad alrededor del Mundo (Fuente AWS)

De cara a dar respuesta a esta situación surge el FL que permite entrenar modelos de forma distribuida sin mover los datos. Son los modelos los que se envían a dónde están los datos, lo que reporta numerosos beneficios como veremos a continuación. Estas son las principales características del FL:

1. Minimización del acceso al dato, siempre permanece bajo el control del propietario.
2. En combinación con otras tecnologías habilitadoras de la privacidad, como Differential Privacy, protocolos de agregación segura o técnicas de k-anonimidad cumple las más estrictas garantías de privacidad.

3. Útil y escalable en sistemas productivos. Por ejemplo, Google tiene desplegados modelos predictivos en los billones de dispositivos móviles con sistema operativo Android.
4. Es una herramienta para entrenar modelos en el extremo (Training on the Edge), lo que permitirá, por ejemplo, procesar los terabytes de datos que producen los coches autónomos.
5. Es una herramienta colaborativa que permite, por ejemplo, a hospitales entrenar modelos sobre imágenes médicas, todo ello preservando la privacidad de los pacientes.

Para facilitar la adopción del FL “on the edge” de la red (dispositivos) para un creciente número de aplicaciones que empleen modelos de Machine Learning, MLEDGE busca el desarrollo de técnicas, librerías y componentes que permitan iniciar estos servicios más ágilmente.

Acuratio está en disposición de desarrollar una plataforma de aprendizaje federado y dar soporte a los distintos servicios federados creados por los participantes de los paquetes de trabajo 3 y 4. Para ello se les proporcionará ayuda en el diseño, desarrollo, entrenamiento, demostración y despliegue de modelos federados CloudEdge.

Dentro de la estrategia de datos europea es clave la compartición de datos. Por eso iniciativas como Gaia-X son ampliamente apoyadas por los distintos organismos gubernamentales europeos. En España, este proyecto ha contado con el especial apoyo de la Oficina del Dato dependiente de la Secretaría de Estado de Digitalización e Inteligencia Artificial.

Actualmente, cuando dos entidades desean colaborar para entrenar un modelo sobre los datos en común. O bien una entidad envía sus datos a la otra, lo cual genera riesgos de pérdida de control sobre dichos datos, y en muchas ocasiones no es factible ya que no se han recogido los consentimientos adecuados. O bien se utiliza un intermediario al que se envían los datos anonimizados con la consiguiente pérdida de valor.

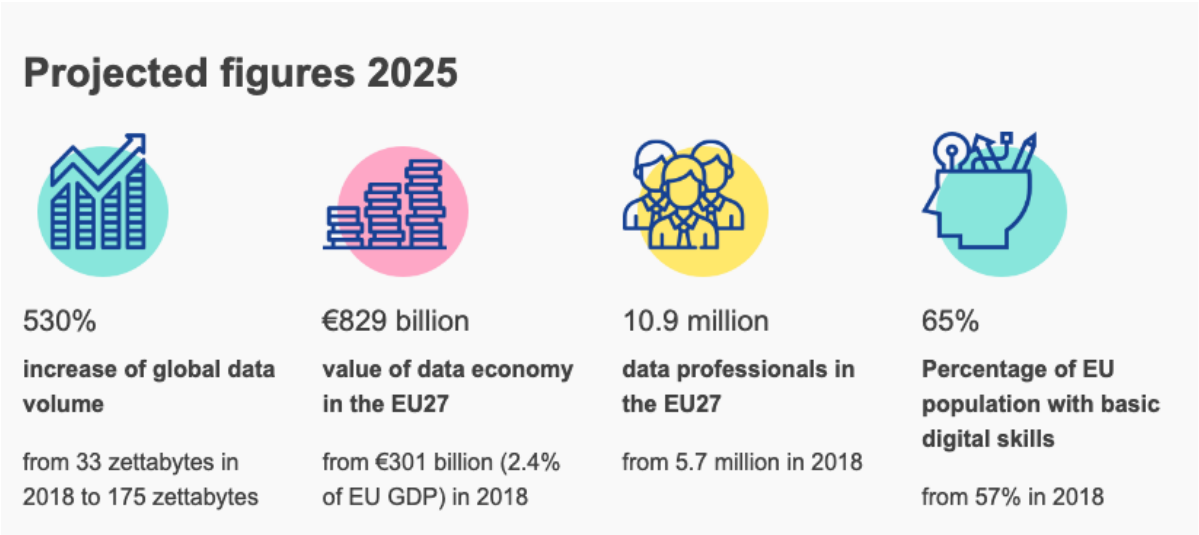


Ilustración 2: Estimaciones de la Unión Europea

Estos procesos son costosos, tediosos, requieren mucho tiempo y aprobaciones, y terminan desincentivado la compartición de datos. Se puede llegar a tardar meses o incluso años y el coste puede llegar a los cientos de miles de euros solo para un intercambio.

Contar con una infraestructura que haga de este tipo de colaboraciones algo sencillo y cotidiano ayudará a impulsar innovaciones en medicina personalizada, mejorar la movilidad y/o tomar decisiones basadas en datos para mejorar los servicios públicos.

Ejemplos de uso de datos industriales y comerciales:

- Los motores a reacción llenos de miles de sensores recopilan y transmiten datos para garantizar un funcionamiento eficiente.
- Los parques eólicos utilizan datos industriales para reducir el impacto sobre el entorno y optimizar la producción de energía eólica.
- La predicción del tráfico en tiempo real puede ahorrar hasta 730 millones de horas. Esto representa hasta 20 mil millones de euros en costes laborales.
- La predicción en tiempo real de los retrasos de los trenes puede ahorrar 27 millones de horas de trabajo. Esto equivale a 740 millones de euros en costes laborales.

1.2. Objetivos

El objetivo global de este paquete de trabajo es el desarrollo y despliegue de una plataforma de FLaaS que facilite el diseño y ejecución de modelos federados incorporando novedosas técnicas de privacidad. Idealmente esta plataforma de FLaaS permitirá tanto la simulación de casos de uso federados como la implementación de modelos reales en entornos de producción para los casos de uso de economía tradicional y digital. Los objetivos específicos se detallan a continuación:

1. Desplegar una capa de software distribuido FLaaS que permita a todos los adjudicatarios de MLEDGE implementar sus casos de uso.
2. Proporcionar la infraestructura para el despliegue de los entornos federados de los demostradores para MLEDGE.
3. Dar soporte técnico para el entrenamiento y explotación de los modelos federados para los casos de uso de MLEDGE.
4. Colaborar con la Fundación en la prueba, integración y demostración de módulos de FL seguro
    - Securing Federated Sensitive Topic Classification against Poisoning Attacks
    - Leveraging Quadratic Voting in Federated Learning.

1.3. Situación actual

El aprendizaje federado es una técnica de aprendizaje automático que entrena un algoritmo a través de una arquitectura descentralizada formada por múltiples dispositivos con datos locales y privados. Este enfoque, creado por Google en 2017, contrasta con las técnicas tradicionales de aprendizaje automático centralizado y de enfoques descentralizados más clásicos, en las que todos los conjuntos de datos se cargan en un servidor de manera idéntica a como están almacenados en local. Gracias a esto se conserva la integridad de la información que está siendo utilizada para el aprendizaje sin poner en peligro la privacidad y seguridad.

Funcionamiento:

El aprendizaje federado se basa en un proceso iterativo en el que se diseña un modelo que se envía los dispositivos en el extremo (edge) donde se cada dispositivo usa sus datos para “entrenar” el modelo sobre una cantidad definida de datos y cuando terminar envían estos modelos “actualizados” al servidor central donde se agregan con el resto de los modelos que han sido entrenados por el resto de los dispositivos distribuidos.

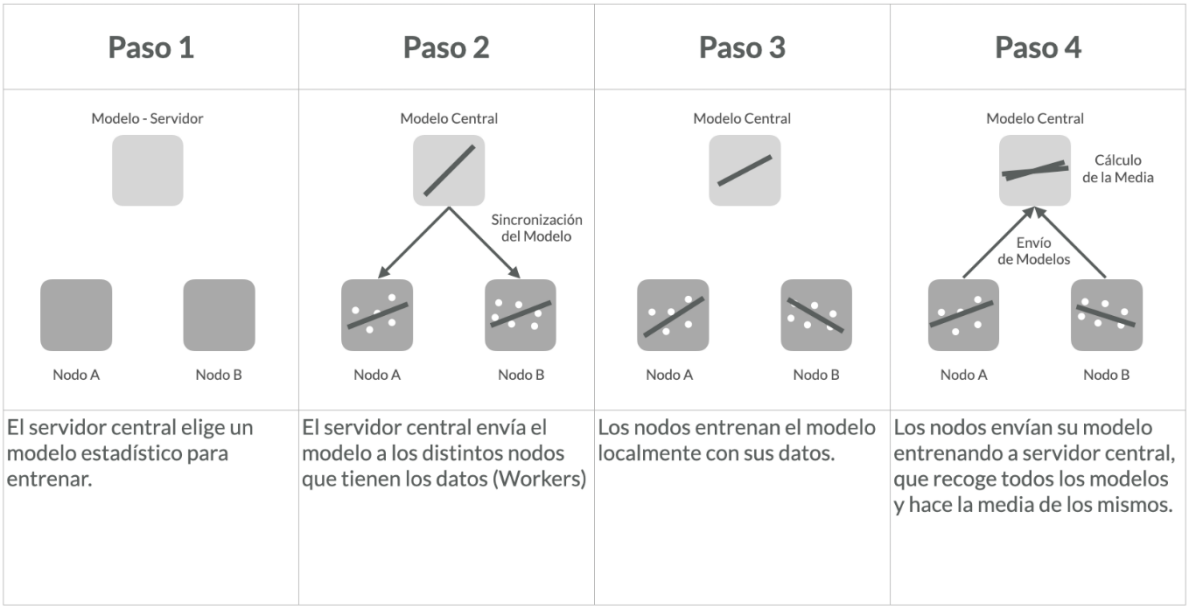


Ilustración 3: Proceso de entrenamiento Federado Horizontal

Plataformas de Aprendizaje Federado

Desde el 2017 en que Google publicara su primera investigación, haciendo posible el método. Han surgido distintas plataformas, proyectos y startups de aprendizaje Federado. En este punto cabe notar que la primera implementación en Open Source la publica el equipo de Acuratio: https://github.com/coMindOrg/federated-averaging-tutorials. Esta implementación fue inmediatamente absorbida por la comunidad, por ejemplo, OpenMined la incluyó en su librería PySyft y los subsiguientes proyectos se han inspirado en ella por su sencillez.

Librerías Open Source:

- OpenMined - PySyft: Primera comunidad open source dedicada al Federated Learning, fundada en 2017. https://openmined.org/
- FATE: es un proyecto de código abierto que tiene como objetivo apoyar un ecosistema de IA seguro y federado. FATE está disponible para implementaciones independientes y en clúster. El proyecto de código abierto está respaldado por WeBank, un neobanco de propiedad privada con sede en Shenzhen, China.
- NVIDIA FLARE: Librería que proviene de NVIDIA CLARA que es una suite de plataformas de computación para industrias con datos sensibles. https://nvflare.readthedocs.io/en/main/flare_overview.html
- Flower: Comunidad surgida de Cambridge que ha creado una librería muy versátil. https://flower.ai/
- Substra – Owkins: es un software de aprendizaje federado desarrollado inicialmente por un proyecto de investigación de múltiples socios en torno a Owkin. Owkin contribuyó con Substra a la Fundación Linux, que ahora mantiene Substra. https://github.com/Substra/substra/tree/main/docs
- Intel – Open Federated Learning: es un proyecto de código abierto Python 3 desarrollado por Intel para implementar FL sobre datos confidenciales. OpenFL tiene scripts de implementación en bash y aprovecha certificados para asegurar las comunicaciones, pero requiere que el usuario maneje la mayor parte de esto por sí mismo. https://openfl.readthedocs.io/en/latest/index.html
- TensorFlow Federated (TFF): es un proyecto de código abierto en Python 3 para el aprendizaje federado desarrollado por Google. La principal motivación detrás de TFF

fue la necesidad de Google de implementar predicciones de teclado móvil y búsqueda en el dispositivo. Google utiliza TFF activamente en producción. https://www.tensorflow.org/federated?hl=es
- Baidu – PaddleFL: PaddleFL es un proyecto de aprendizaje federado de código abierto basado en PaddlePaddle: https://paddlefl.readthedocs.io/en/latest/
- FedML: Startup que ha publicado parte de su código en open source: https://github.com/FedML-AI/FedML
- Apple: recientemente han liberado su librería para simular los distintos tipos de Federated Learning: https://apple.github.io/pfl-research/index.html

Startups:

Además de Acuratio que fue la primera startup de Federated Learning, han surgido distintas startups para resolver diversos problemas, las más notables son:

Salud: Owkin, Apheris y TripleBlind

IoT: Scaleout, DynamoFL

Sector Financiero: Acuratio, Inpher, Duallity o Enveil

Investigaciones punteras:

Se han identificado las siguientes investigaciones y técnicas de IMDEA Networks como las más prometedoras para implementar en una plataforma productiva. Su implementación está alineada con los objetivos de MLEDGE y se estudiará la integración de todas o parte de las mismas:

- *Securing Federated Sensitive Topic Classification against Poisoning Attacks*: Técnica que trata de minimizar la influencia de poisoning attacks https://dspace.networks.imdea.org/handle/20.500.12761/1633
- *FedQV: Leveraging Quadratic Voting in Federated Learning*: Tecnica que implementa un sitema de voto/reputación para prevenir poisoning attacks https://arxiv.org/abs/2401.01168
- *FreqyWM: Frequency WaterMarking for the New Data Economy*: Técnica que dearrolla una nueva efoque para introducir marcas de agua en los datos. https://dspace.networks.imdea.org/handle/20.500.12761/1779
- *Try Before You Buy: a Practical Data Purchasing Algorithm for Real-World Data Marketplaces*: Innovador enfoque para que los compradores de datos puedan evaluar la compra sin tener acceso al conjunto de datos completo: https://dspace.networks.imdea.org/handle/20.500.12761/1640

Tras efectuar un análisis pormenorizado de los proyectos de investigación, se ha decidido implementar y demostrar las siguientes técnicas en la plataforma:

- Securing Federated Sensitive Topic Classification against Poisoning Attacks.
- Leveraging Quadratic Voting in Federated Learning.

1.4. La solución

De cara a entrenar modelos en dispositivos en el extremo y poder procesar los datos en el propio dispositivo surge FL que permite entrenar modelos de forma distribuida sin mover los datos. Son los modelos los que se envían a dónde están los datos, lo que reporta numerosos beneficios como los que se señalan en el apartado anterior y que la plataforma FLaaS a desarrollar por Acuratio ayudará a ilustrar en el marco de MLEDGE.

Para facilitar la adopción del FL en un creciente número de aplicaciones que usan modelos de Machine Learning en los dispositivos en el extremo de la red. MLEDGE busca el desarrollo de técnicas, librerías y componentes que permitan iniciar estos servicios más fácil y ágilmente.

Acuratio proveerá una plataforma de aprendizaje federado y dará soporte a los distintos casos de uso de MLEDGE. Para formar la infraestructura básica sobre la que se desarrollarán los casos de uso de Aprendizaje Federado para todos los paquetes de trabajo, necesitaremos cinco componentes principales:

1. *Capa de Ejecución*: En esta capa se ejecutarán los modelos sobre datos locales pertenecientes a cada participante.
2. *Capa de Comunicaciones*: Para poder conectar todos los elementos que componen una plataforma de aprendizaje federado, las comunicaciones son un elemento crítico.
3. *Capa de Coordinación*: Aunque el objetivo es generar una infraestructura lo más distribuida posible, es necesario un elemento común que permita coordinar a todos los participantes.
4. *Capa de Gobernanza*: La capa de gobernanza se implementará como una API accesible a todos los participantes.

Adicionalmente, se implementará un caso de uso para la optimización de costes en el despliegue de proyectos de aprendizaje federado en infraestructuras cloud. Dicha implementación requerirá añadir a la solución una *Capa de Costes*, una abstracción de la integración con las herramientas de facturación de la nube que permita obtener la información necesaria sobre la disponibilidad y el coste de los recursos necesarios para el aprendizaje federado en diferentes plataformas.

Para alcanzar los objetivos seguiremos tres fases bien diferenciadas:

1. **Integración con APIs de facturación:** Integraremos con las APIs de los principales proveedores cloud (GCP, AWS, Azure) para generar un dashboard centralizado en el que evaluar los costes de un proyecto federado.
2. **Identificación de recursos comunes:** Identificaremos el conjunto de recursos de computación más habitual en un proyecto federado, y su coste en cada una de las plataformas.
3. **Generación de plantillas:** Generaremos plantillas que permitan lanzar proyectos federados en distintas nubes, de manera que la herramienta pueda servir para seleccionar la distribución optima de cargas para acceder a los recursos cloud con el proveedor más asequible.

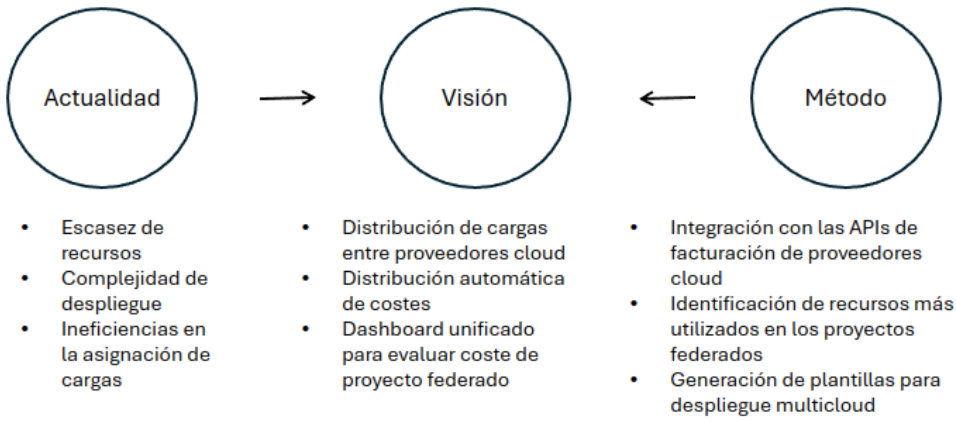


Ilustración 4: Metodología de la solución

1.5. Casos de uso

Caso de uso	Usuario	Objetivo	Beneficio, resultado, razón del caso de uso
OPTIMIZACIÓN DE COSTES CLOUD	Desarrollador de modelos federados	Facilitar el proceso de despliegue de modelos federados en múltiples nubes	Reducir los costes al permitir comparar los precios de los recursos en distintas nubes
MEJORA PROTOCOLOS FEDERATED LEARNING	Usuarios de la plataforma FLaaS	Incorporar la investigación de la fundación IMDEA en los protocolos federados soportados por la plataforma	Mejorar la seguridad de los sistemas de aprendizaje federado facilitando el uso de nuevos mecanismos de protección desarrollados por IMDEA

Los casos de uso planteados permitirán a los usuarios de la plataforma FLaaS que desarrollen modelos federados desplegar estos modelos en cualquiera de los principales proveedores de servicios cloud a través de una única interfaz. Además, podrán incorporar mecanismos de seguridad avanzados utilizando librerías y métodos integrados en la propia solución, mejorando de este modo la seguridad de sus desarrollos.

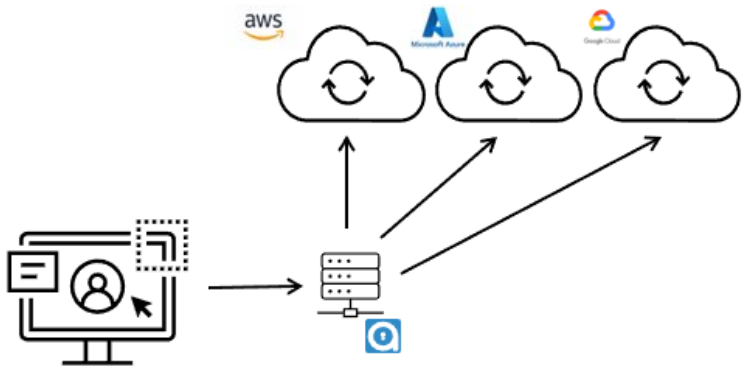


Ilustración 5: Optimización de la selección de proveedor cloud

En la ilustración 5, se presenta el caso de uso de optimización de infraestructuras cloud. El objetivo principal es automatizar el proceso de selección de proveedor para que, con la definición de requisitos necesarios para cada modelo, el componente desarrollado pueda seleccionar la nube más asequible para cumplir con los criterios establecidos.

```
input_layer = layers.InputLayer(input_shape=(23,))
net = layers.Dense(12, activation=nn.relu)(input_layer.output)
net = layers.Dropout(0.5)(net)
net = layers.Dense(1, activation=nn.sigmoid)(net)

model = federatedmodel.FederatedModel(
    inputs=input_layer.output,
    outputs=net,
    secure_aggregation=True,
)
```

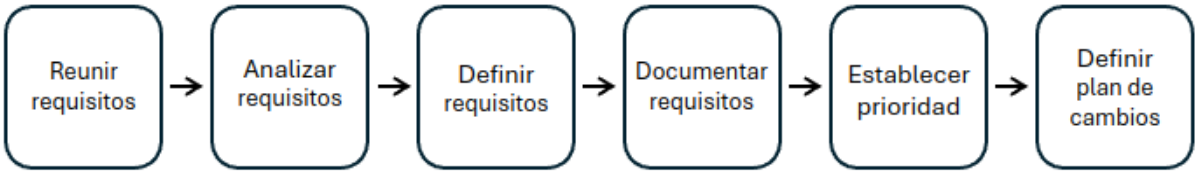
Ilustración 6: Definición de un modelo federado con redes neuronales.

En la ilustración 6, se presenta la interfaz prevista para el caso de uso de mejora de protocolos de federated learning. La plataforma FLaaS ofrecerá APIs de python que permitan entrenar modelos con protocolos federados como Federated Averaging en el caso de la figura. Aquí por ejemplo se muestra cómo se puede activar un protocolo de activación segura mediante la configuración de un único parámetro en la API de entrenamiento. Se seguirá el mismo procedimiento para los dos módulos de FL adicionales que implementan medidas contra ataques de envenenamiento y para el uso de *quadratic voting* en la agregación de datos.

2. Especificación de requisitos

2.1. Metodología

Para la elaboración de los requisitos se ha seguido una metodología en seis fases.



Primero, se acuerdan los requisitos de los componentes para el caso de uso propuesto, para que todos los objetivos puedan cumplirse de manera satisfactoria. A continuación, se alinean estos requisitos con los objetivos del caso de uso, y se clasifican en funcionales y no funcionales.

Una vez hecho esto se definen los requisitos de manera que sean claros, precisos y fácilmente comprobables. Este proceso de definición se acompaña de la documentación apropiada para poder alinear cada requisito con su objetivo dentro del caso de uso.

Por último, se establece la prioridad de cada requisito entre opcionales y obligatorios y se establece el procedimiento para incorporar o eliminar requisitos a lo largo del proyecto. A continuación, se especifican los requerimientos funcionales y no funcionales de la plataforma, incluyendo las nuevas funcionalidades de optimización de los costes cloud y la mejora de los protocolos de FL usando las técnicas fruto de la investigación de IMDEA Networks.

2.2. Requerimientos funcionales

Requisitos de la capa de ejecución de la plataforma FLaaS			
#	**Requisito**	**Descripción**	**Tipo**
RF1	Carga de datos	El sistema debe permitir la carga de datos desde múltiples fuentes diferentes, tales como sistemas de almacenamiento en la nube o a través de una interfaz con cifrado en el navegador	Obligatorio
RF2	Procesamiento de datos	El sistema debe permitir el tratamiento de datos con herramientas tradicionales como: pandas, numpy, sklearn...	Opcional
RF3	Visualización de datos	El sistema debe permitir la visualización de datos mediante una interfaz basada en *jupyter notebook* o similar.	Obligatorio
RF4	Desarrollo de modelos	El sistema debe permitir entrenar modelos para desarrollar los distintos casos de uso.	Obligatorio
RF5	Procesamiento seguro	El sistema debe permitir implementar técnicas de procesamiento seguro con sistemas de cifrado y diversas Privacy Enhancing Technologies.	Obligatorio

Requisitos de la capa de ejecución de la plataforma FLaaS			
#	**Requisito**	**Descripción**	**Tipo**
RF6	Portabilidad	La infraestructura sobre la que se ejecutan las cargas de trabajo debe ser intercambiable, para ello el despliegue de la capa de ejecución se servirá de tecnologías open source de despliegue con contenedores (Docker).	Obligatorio
RF7	Versatilidad	El sistema debe permitir diferentes tipos de aprendizaje federado para servir a los casos de uso, incluyendo aprendizaje federado vertical y horizontal	Obligatorio
RF8	Multicloud	El sistema debe ser una plataforma multicloud, que trabaje al menos con las principales plataformas (AWS, Azure, Google Cloud) y permitir su despliegue también en infraestructura *on-premise* del cliente o en el borde de la nube (cloud Edge).	Opcional
RF9	Administración y gestión	La plataforma debe proporcionar herramientas administrativas y permisos especiales para la gestión de los modelos a ser entrenados.	Obligatorio

	Requisitos de la capa de comunicaciones de la plataforma FLaaS		
#	**Requisito**	**Descripción**	**Tipo**
RF10	Transmisión segura	El sistema debe permitir la transmisión segura de datos con protocolos de cifrado como TLS.	Obligatorio
RF11	Compatibilidad	La capa de comunicaciones debe permitir comunicarse con diferentes nubes.	Obligatorio

Requisitos de la capa de coordinación de la plataforma FLaaS			
#	**Requisito**	**Descripción**	**Tipo**
RF12	Federación de modelos	El sistema debe permitir implementar protocolos federados para el entrenamiento de modelos preservando la privacidad del dato.	Obligatorio
RF13	Analítica federada	El sistema debe permitir coordinar protocolos de analítica federada basada en Secure Multi-Party Computation y técnicas de cifrado homomórfico entre otros.	Obligatorio
RF14	Programación de tareas	El sistema debe permitir pre-autorizar y coordinar colas de experimentos para facilitar la experimentación y el desarrollo de nuevos sistemas federados.	Opcional

#	Requisito	Descripción	Tipo
RF15	Confidencialidad	La API de coordinación debe garantizar la seguridad y confidencialidad de la información sobre los participantes que almacene en su base de datos	Obligatorio
RF16	Aislamiento de recursos	La API de coordinación debe proporcionar la posibilidad de aislar los recursos de cada organización, de forma que una misma API sirva para múltiples proyectos federados.	Obligatorio

Requisitos de la capa de coordinación de la plataforma FLaaS			
#	**Requisito**	**Descripción**	**Tipo**
RF17	Acceso al dato	El sistema debe implementar un control de accesos que permita controlar quién accede a la interfaz de visualización de datos.	Obligatorio
RF18	Abstracción de datos	El sistema debe utilizar roles para agrupar los permisos relacionados.	Opcional
RF19	Minimización de privilegios	A los usuarios se les deben asignar solo los roles que les dan el menor nivel de acceso necesario para desarrollar las funciones designadas.	Opcional
RF20	Separación de funciones	Ningún usuario debe tener control total sobre los elementos críticos de la capa de gobernanza.	Opcional

Requisitos de la capa de costes de la plataforma FLaaS			
#	**Requisito**	**Descripción**	**Tipo**
RF21	Control de costes	El sistema debe permitir visualizar el coste total de los recursos consumidos por cada proyecto federado cuando estos recursos estén desplegados en la nube.	Obligatorio
RF22	Arbitraje	El sistema debe permitir la comparativa de costes entre proveedores para recomendar y poder seleccionar la opción más asequible para cada modelo.	Opcional
RF23	Personalización	La plataforma debe permitir la integración con las credenciales de cada cliente para la interacción con las herramientas de gestión de costes de los proveedores cloud, de forma que la recomendación pueda ser personalizada.	Opcional

2.3. Requerimientos no funcionales

#	Requisito	Descripción	Tipo
RNF1	Facilidad de uso	El sistema debe ser sencillo de usar y facilitar la integración a las empresas que lo usen.	Opcional
RNF2	Seguridad de comunicaciones	La herramienta debe incorporar mecanismos de agregación segura de información entre los nodos basados en el estado del arte	Obligatorio
RNF3	Fiabilidad	El sistema debe permitir la comunicación robusta en entornos complejos como aquellas redes protegidas con firewall de grado empresarial o NAT.	Obligatorio

3. Arquitectura de la plataforma FLaaS

En esta sección, se presenta la arquitectura de la plataforma FLaaS del proyecto MLEDGE junto con una descripción de los diferentes componentes de esta. Este conjunto de elementos conformará la infraestructura básica sobre la que se desarrollarán los casos de uso de Aprendizaje Federado para todos los paquetes de trabajo.

3.1. Descripción de la arquitectura

La plataforma FLaaS constará de cinco componentes principales: capa de ejecución, capa de comunicaciones, capa de coordinación, capa de gobernanza y capa de costes.

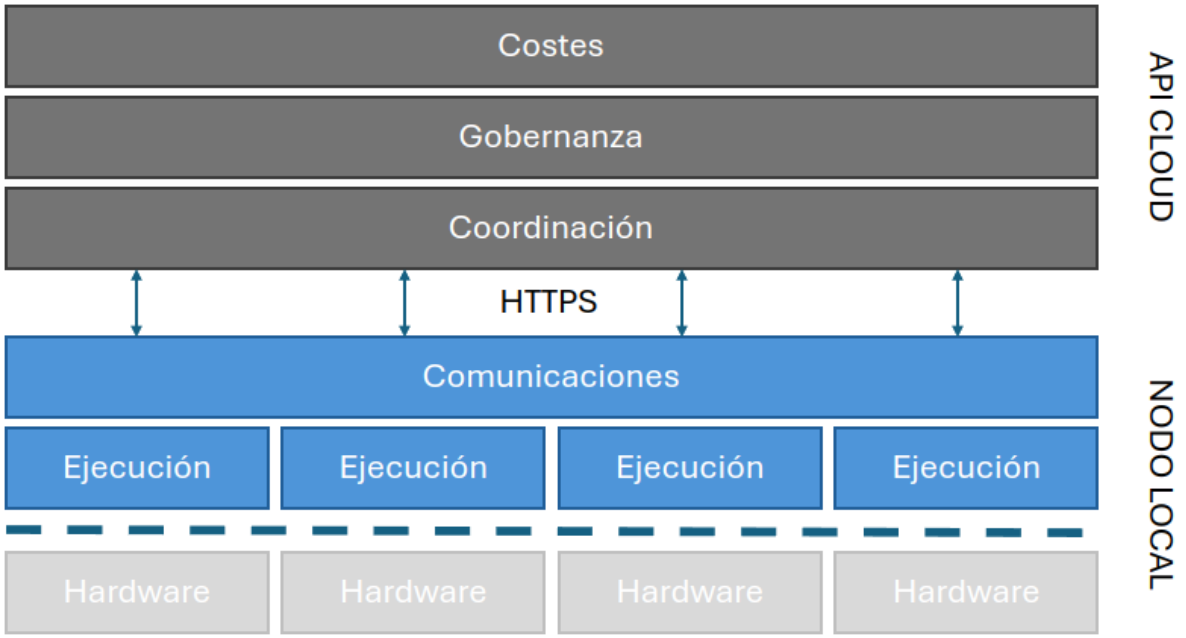


Ilustración 7: Arquitectura FLaaS

La capa de ejecución y la capa de comunicaciones se implementarán localmente en los nodos de computación. Los nodos de computación serán a su vez los servidores Edge con acceso al dato, y por lo tanto los encargados de entrenar los modelos de Federated Learning. La filosofía de este diseño es minimizar el movimiento de datos llevando el procesamiento al extremo para, de este modo, lograr una mayor privacidad y seguridad en el entrenamiento de nuestros modelos de Machine Learning.

Las capas de costes, gobernanza y coordinación se implementarán en servidores en la nube accesibles para todos los nodos a través de HTTPS. De este modo podremos controlar los nodos de forma remota a través de código o mediante sencillas interfaces de usuario. Adicionalmente la capa de costes nos permitirá implementar el caso de uso de optimización de infraestructura cloud con un panel de control en el que podamos visualizar los costes de despliegue en la nube asociados a cada proyecto de Federated Learning.

Para el desarrollo se priorizan soluciones de código abierto, integrando librerías muy utilizadas en la industria como:

- **Keras:** Librería para desarrollo de ML multi-plataforma https://keras.io/keras_3/
- **TensorFlow:** Plataforma de extremo a extremo enfocada en el aprendizaje automático https://www.tensorflow.org/

- **Pandas:** Librería para el analísis de datos en python https://pandas.pydata.org/
- **NumPy:** Librería para crear vectores y matrices multidimensionales en python https://numpy.org/
- **PyArrow:** Plataforma para el desarrollo de analítica eficiente en memoria https://arrow.apache.org/
- **Shap:** Librería para explicabilidad de modelos en python https://shap.readthedocs.io/en/latest/

Para el despliegue se utilizará una arquitectura basada en contenedores Docker que permita lanzar cargas de trabajo federadas en cualquier sistema operativo y hardware. Así lograremos un entorno estable en el que poder desarrollar los modelos federados y que proporcione flexibilidad a la plataforma.

4. Diseño detallado de la solución

4.1. Capa de Ejecución

En esta capa se ejecutarán los modelos sobre datos locales pertenecientes a cada participante. Es el componente de más bajo nivel, se instala y ejecuta lo más cerca posible del punto en el que se almacenan los datos. Para los demostradores de este proyecto esta capa se desplegará en servidores de la nube, sin embargo, tiene que ser portable y fácil de desplegar ya que en casos de uso de producción podría ser necesario su despliegue en dispositivos *edge*.

Se seguirá una filosofía de diseño modular basada en contenedores ya que ofrece la mayor flexibilidad a la hora de desplegar. Las herramientas se desarrollarán empleando principalmente dos lenguajes de programación: *python* y *Rust*. Las interfaces que se diseñen para ser empleadas directamente por el usuario se programarán en *python* por su flexibilidad y facilidad de uso, además esto nos permitirá construir encima de herramientas como *TensorFlow y PyTorch* muy extendidas en el mundo del aprendizaje automático. En cambio, los componentes más automatizados serán construidos con Rust por su eficiencia y seguridad.

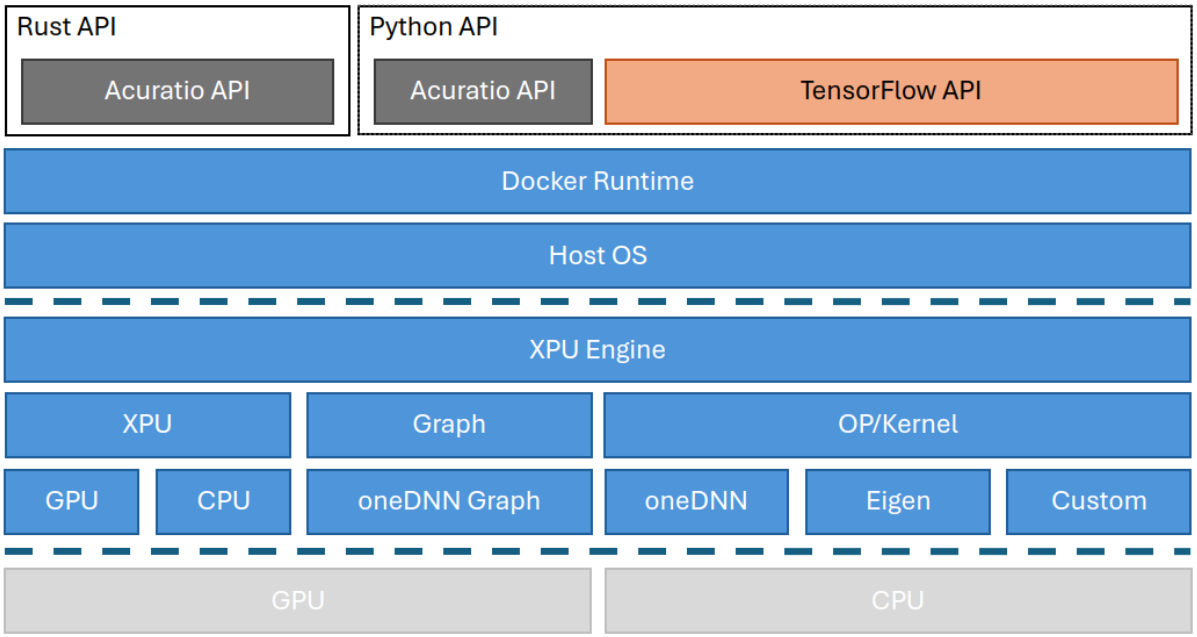


Ilustración 8: Arquitectura de la capa de ejecución

Docker utiliza contenedores para crear entornos virtuales que aíslan nuestro *software* del resto del sistema. La principal ventaja es que los programas que se ejecutan dentro de este entorno virtual pueden compartir recursos con su máquina *host*. En la ilustración 8 se muestra un diagrama de la arquitectura para la capa de ejecución, podemos observar que *docker* nos permite aislar nuestro *software* del *hardware* subyacente, facilitando el despliegue en entornos diversos (nube o *edge*).

Dentro del contenedor instalaremos las APIs de *python* que permitirán el desarrollo de modelos, así como la API de *rust* para el resto de los procesamientos. El desarrollo de la librería en *python* nos permitirá soportar múltiples modelos diferentes y el despliegue basado en contenedores nos proporcionará versatilidad y la capacidad de lanzar el software en múltiples nubes.

Para la facilidad de uso, lanzaremos una interfaz basada en Jupyter Lab. De este modo proporcionaremos acceso programático a la capa de ejecución de modo que un usuario pueda desplegar y entrenar modelos directamente sobre el servidor en el que se almacenan los datos. En el *kernel* de *jupyter* instalaremos las librerías de *python* de Acuratio, que expondrán las APIs necesarias para poder entrenar modelos federados de forma sencilla y sin necesidad de ser un experto.

Finalmente, para garantizar la seguridad en las comunicaciones todos los *endpoints* estarán protegidos con HTTPS.

4.2. Capa de Comunicaciones

Para poder conectar todos los elementos que componen una plataforma de aprendizaje federado, las comunicaciones son un elemento crítico. Estas comunicaciones deben ser seguras y fiables, pero a la vez tiene que ser posible conectar nodos que puedan desplegarse en redes muy diversas. Algunos de estos nodos pueden estar protegidos por firewall complejos, conectados a redes privadas o tener una IP cambiante entre otros desafíos. Por lo tanto, contar con una capa de comunicaciones capaz de garantizar una conexión fiable y robusta entre todos los dispositivos que forman parte de la capa de ejecución es crucial.

La capa de comunicaciones se basará en protocolos de código abierto, que permitan crear un conjunto de túneles cifrados extremadamente livianos entre dos puntos en una red. Estos túneles tienen que permitir la conexión directa entre todos los nodos en una red de malla, sin un concentrador que actúe de intermediario. Para generar esta red de forma segura y eficiente seguiremos el siguiente protocolo:

1. Cada nodo genera un par de claves pública/privada aleatorias para sí mismo y asocia la clave pública con su identidad.
2. El nodo contacta al servidor de coordinación y deja su clave pública y una nota sobre dónde se puede encontrar actualmente ese nodo y en qué dominio se encuentra.
3. El nodo descarga una lista de claves públicas y direcciones en su dominio, que otros nodos han dejado en el servidor de coordinación.

La clave privada nunca abandona el nodo. Esto es importante porque la clave privada es lo único que podría usarse para hacerse pasar por ese nodo. Como resultado, sólo ese nodo puede cifrar o descifrar paquetes dirigidos a él mismo. Dicho de otra manera, las conexiones están cifradas de extremo a extremo.

4.3. Capa de Coordinación

Aunque el objetivo es generar una infraestructura lo más distribuida posible, es necesario un elemento común que permita coordinar a todos los participantes. Este elemento común es la capa de coordinación, será una API desplegada en un dominio seguro y accesible para todos los elementos de la capa de ejecución. El papel de esta API será indicar a cada participante cuando debe iniciar una computación, así como informarle de la identidad del resto. En ningún caso esta capa de coordinación trasegará datos en crudo ni gradientes de los modelos, su único papel es coordinar a los distintos elementos de la capa de ejecución y estos se conectarán directamente en una red de malla para intercambiar la información necesaria para entrenar los modelos.

La arquitectura escogida para nuestro api de coordinación es *REST* (*REpresentational State Transfer*), por su sencillez y flexibilidad. Esto nos permitirá construir un API escalable, sencillo

y fácil de extender o modificar. Para que una API se considere RESTful, debe cumplir con estos criterios:

- **Una arquitectura cliente-servidor** formada por clientes, servidores y recursos, con solicitudes gestionadas a través de HTTP.
- **Comunicación cliente-servidor sin estado**, lo que significa que no se almacena información del cliente entre solicitudes de obtención y cada solicitud es independiente y desconectada.
- **Datos almacenables en caché** que agilizan las interacciones cliente-servidor.
- **Una interfaz uniforme** entre componentes para que la información se transfiera de forma estándar. Esto requiere que:
    - Los recursos solicitados son identificables y separados de las representaciones enviadas al cliente.
    - Los recursos pueden ser manipulados por el cliente a través de la representación que reciben porque la representación contiene suficiente información para hacerlo.
    - Los mensajes autodescriptivos devueltos al cliente tienen suficiente información para describir cómo el cliente debe procesarlos.
    - El hipertexto/hipermedia está disponible, es decir, tras acceder a un recurso, el cliente debería usar hipervínculos para encontrar todas las demás acciones disponibles que puede realizar.
- **Un sistema por capas** que organiza cada tipo de servidor (los responsables de la seguridad, el equilibrio de carga, etc.) implicaba la recuperación de la información solicitada en jerarquías, invisibles para el cliente.
- **Código bajo demanda** (opcional): la capacidad de enviar código ejecutable desde el servidor al cliente cuando lo solicite, ampliando la funcionalidad del cliente.

4.4. Capa de Gobernanza

La capa de gobernanza se implementará como una API accesible a todos los participantes. Esta API utilizará un sistema de cifrado asimétrico y funcionará como un proveedor de identidades. De este modo podrá controlar quién tiene acceso a cada recurso. Desde esta capa se controlan las interacciones entre los distintos participantes, sin embargo, el acceso a la capa de ejecución de cada entidad (donde residen los datos sensibles) está protegido también en un nivel inferior, en la capa de comunicaciones. De este modo si por cualquier motivo la capa de gobernanza se ve comprometida y alguien obtiene acceso a la capa de ejecución de otro participante, no podrá acceder a sus datos ya que los controles en la capa de comunicaciones lo impedirán.

RBAC

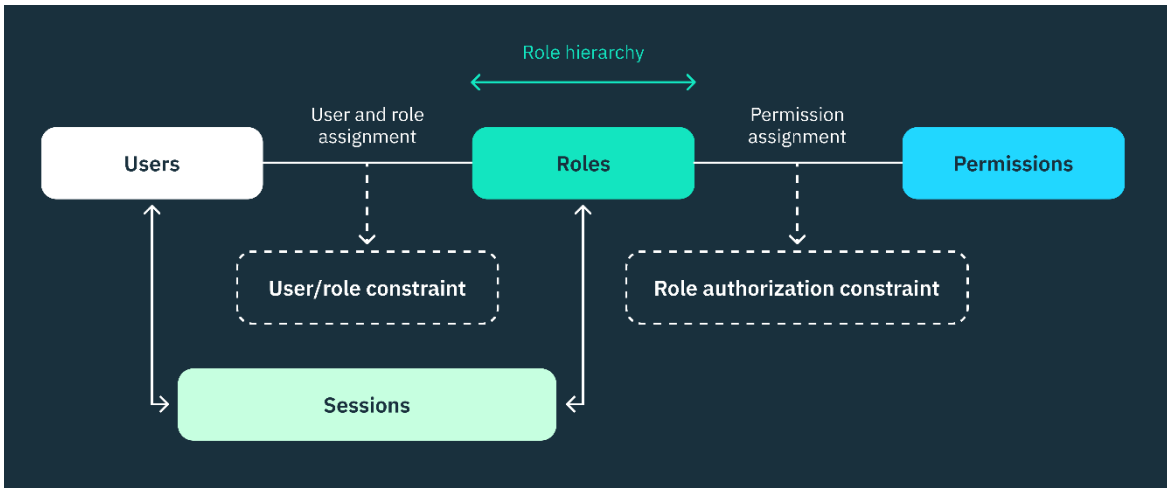
Para implementar está capa de gobernanza se seguirá un modelo *RBAC* (*Role Based Access-Control*), debido a que se trata de una arquitectura de seguridad flexible que proporciona un marco para crear un sistema de control de acceso adaptado a las necesidades de nuestro caso de uso.

RBAC se basa en tres principios de seguridad básicos:

1. **Abstracción de datos:** No se asignan permisos directamente a los usuarios, sino que se utilizan roles para agrupar los permisos relacionados. Estos roles son abstracciones que ocultan las complejidades subyacentes de los permisos y derechos de acceso de usuarios individuales. De este modo podemos crear una estructura más

organizada que facilita la gestión independientemente del número de usuarios con el que cuente nuestro sistema.

2. **Minimización de privilegios:** A los usuarios se les asignan solo los roles que les dan el menor nivel de acceso necesario para desarrollar las funciones designadas. Este enfoque redice la superficie de ataque en caso un incidente de seguridad al limitar los permisos del atacante que consiga hacerse pasar por un usuario.
3. **Separación de funciones:** Ningún usuario tiene control total sobre los elementos críticos de la capa de gobernanza. Las acciones críticas siempre requieren la aprobación de varios usuarios, lo que proporciona una capa de seguridad y atribución de responsabilidades muy robusta.



Roles y permisos

Los roles en RBAC son una colección de permisos que definen las operaciones que se pueden realizar sobre los recursos de un sistema. Estos permisos incluyen normalmente operaciones de lectura, escritura o eliminación y se pueden asignar a usuarios o grupos de usuarios que requieren un nivel de acceso similar.

Por ejemplo, los roles pueden variar desde asignaciones de alto nivel, como administrador de un espacio de trabajo, hasta roles más específicos que pueden heredar varios usuarios, como el rol de desarrollador. Esta arquitectura permite a los administradores gestionar el acceso a los recursos de un espacio de trabajo sin configurar manualmente los permisos para cada usuario.

Cuando a un usuario se le asignan múltiples roles con permisos similares o superpuestos en un sistema RBAC, efectivamente hereda una combinación de todos los permisos asociados con cada rol.

El modelo del Instituto Nacional de Estándares y Tecnología (NIST) para el control de acceso basado en roles (RBAC) describe cuatro niveles de implementación incrementales para RBAC. Estos niveles guían la incorporación de roles, permisos, sesiones y otros componentes dentro de las arquitecturas RBAC. Cada nivel se basa en los requisitos del nivel anterior y ofrece mayor funcionalidad y complejidad:

1. **RBAC básico:** Se trata de la implementación mínima de los principios RBAC. En este formato, a los roles se les asignan permisos y posteriormente a los usuarios se les asignan los roles. No existe una estructura jerárquica para los roles. En esta modalidad se permite asignar múltiples roles a un solo usuario, además también se pueden asignar varios usuarios a un mismo rol.

2. **RBAC jerárquico:** En este nivel se agrega una estructura jerárquica a los roles de RBAC. De este modo los roles de nivel superior heredan los permisos y privilegios de los roles de nivel inferior. Esta jerarquía puede ser estricta, todos los roles tienen un rol principal y potencialmente múltiples roles secundarios, o flexible de manera que las relaciones entre roles no exijan un predecesor o sucesor directo.
3. **RBAC restringido:** Esta modalidad incorpora restricciones para cumplir el principio de separación de funciones para mitigar los riesgos asociados con usuarios con privilegios excesivos. De esta manera garantizamos que un usuario no puede desempeñar simultáneamente roles con principios superpuestos.
4. **RBAC simétrico:** Este nivel amplía los anteriores al incorporar la capacidad de revisar los permisos asociados con los roles, de esta manera los administradores pueden identificar y mitigar roles con más privilegios de los necesarios.

4.5. Capa de Costes

La capa de costes es una abstracción de la integración con las herramientas de facturación de la nube. Los principales proveedores de servicios en la nube proporcionan clientes en varios lenguajes de programación para realizar consultas a sus servicios de facturación y monitorizar todos los consumos. Mediante esta integración podremos obtener datos que nos permitan optimizar el uso de la infraestructura *CloudEdge*, ya sea mediante herramientas de gestión tradicionales o mediante el uso de novedosos algoritmos federados que permitan compartir datos de consumo en la nube entre distintas entidades.

Los cuatro componentes anteriormente descritos componen la arquitectura básica de la plataforma de FLaaS. Sobre esta plataforma se desarrollarán los casos de uso para los paquetes de trabajo 3 y 4, y además se ofrecerán los servicios de soporte técnico necesarios para garantizar el éxito de estos casos de uso.

Este último componente, además de ser una pieza clave de la plataforma de FLaaS, permitirá desarrollar el caso de uso de optimización de infraestructuras cloud. Los principales proveedores cloud proporcionan APIs Rest para consultar sus servicios de facturación, estas APIs cumplen los mismos estándares que el componente 4.3 de este documento por lo que se integrarán perfectamente en el diseño de la solución.

A alto nivel, esta capa se compondrá de tres elementos:

1. **Backend** para la recolección de información desde las APIs de los proveedores cloud. Se desarrollará en python y permitirá interactuar con: GCP, Azure y AWS.
2. **Frontend** para la visualización de los costes asociados a cada proyecto federado. Permitirá controlar los costes de los proyectos federados cuyos recursos estén desplegados en la nube.
3. **Wizard** para lanzar proyectos en la nube y proporcionar una estimación del coste en cada uno de los proveedores.

5. Diseño detallado de los demostradores

En esta sección se detalla el diseño de los demostradores para los casos de uso de optimización de infraestructuras cloud e implementación de nuevos mecanismos de seguridad para protocolos de Federated Learning. Esta sección se centra en los casos de uso específicos del paquete de trabajo P5, si bien la plataforma de FLaaS se empleará también en los casos de uso de los paquetes de trabajo P3 y P4 para gestionar modelos de aprendizaje federado. Para más información, se refiere al lector a los entregables E3.1 y E4.1 del proyecto.

5.1. Caso de uso de optimización de costes cloud

Para poder monitorizar los costes asociados con un proyecto de aprendizaje federado, se desarrollará una interfaz en la que se puedan consultar de forma unificada los costes asociados a los recursos lanzados en los principales proveedores de servicios en la nube (GCP, AWS y Azure). Esto, en combinación con la plataforma FLaaS que permite lanzar y gestionar recursos en dichas nubes y coordinar un entrenamiento federado permitirá optimizar los costes asociados a estos proyectos.

La capa de costes de la plataforma FLaaS es el elemento esencial de este demostrador. Consistirá en una integración vía API con las herramientas de costes de las principales nubes y una interfaz en la que puedan visualizarse estos costes de una forma sencilla e intuitiva.

Adicionalmente, este demostrador permitirá usar el modelo de optimización desarrollado en el marco del proyecto MLEDGE para seleccionar de forma automática en que nube es más rentable lanzar un proyecto de aprendizaje federado. Para solucionar esta problemática se planteará una interfaz en la que se pueda configurar un conjunto de recursos en la nube predeterminados, con varias opciones para elegir en función de los requisitos de cada caso de uso. Una vez seleccionado el conjunto de recursos a lanzar, el demostrador recomendará la nube con mejor coste y permitirá la opción de escoger automáticamente la plataforma en la que lanzará estos recursos gestionándolos también de forma automática a través de la plataforma FLaaS.

Guion del demostrador para el caso de uso de optimización de costes cloud	
Paso	**Descripción**
Acceso a la plataforma	Se utilizan las credenciales de la plataforma de acuratio para acceder a la interfaz de gestión de recursos federados
Selección de plantilla	Se selecciona una plantilla para el despliegue de recursos en la nube, en la plantilla se detallarán los recursos necesarios (vCPUs, VRAM, GPUs...)
Comprobación de precios	La capa de costes comprobará el coste de la plantilla seleccionada en los distintos proveedores cloud y mostrará las opciones disponibles.
Selección de proveedor	Se acepta la opción recomendada o se selecciona otra de las opciones disponibles.
Comprobación del servicio	Se comprueba el despliegue en el proveedor seleccionado y se comprueba el progreso del entrenamiento del modelo

5.2. Caso de uso de mejora de protocolos de FL

Para demostrar la implementación de las investigaciones de la fundación IMDEA, se desarrollará un modelo que implemente dichas investigaciones sobre un protocolo de Federated Averaging. El código de este demostrador se liberará en open source.

Los dos métodos a implementar serán los asociados a las siguientes investigaciones:

- Securing Federated Sensitive Topic Classification against Poisoning Attacks
- Leveraging Quadratic Voting in Federated Learning.

En ambos casos, para demostrar su implementación se mostrará un código dentro de la plataforma FLaaS capaz de entrenar un modelo con y sin estos mecanismos de seguridad implementados. Para demostrar su efectividad, se mostrarán comparativas de rendimiento alineadas con los resultados de dichas investigaciones.

Es esencial que el demostrador se ejecute dentro de la plataforma FLaaS, ya que el principal objetivo de este caso de uso es acercar métodos innovadores al desarrollo práctico de modelos federados en la industria, por lo que integrarlos en los flujos de entrenamiento de la propia plataforma hará que estos módulos sean utilizables para los casos de uso de los otros paquetes de trabajo. Por tanto, dichas demostraciones se realizarán según alguno de los casos de uso del proyecto, pero en su defecto se podría considerar en los casos de uso planteados por IMDEA Networks en sus artículos de investigación.

Guion del demostrador para el caso de mejora de protocolos de FL	
Paso	**Descripción**
Acceso a la plataforma	Se utilizan las credenciales de la plataforma de acuratio para acceder a la interfaz de gestión de recursos federados
Selección de plantilla	Se selecciona una plantilla para el despliegue de recursos en la nube, en la plantilla se seleccionará el método que se quiere demostrar (Sensitive Topic Classification o Quadratic Voting)
Despliegue automático del proyecto	La plataforma lanzará automáticamente los recursos necesarios para un entrenamiento incluyendo el módulo de seguridad solicitado y comenzará el entrenamiento según la plantilla prediseñada
Comprobación de resultados	El usuario podrá acceder al servidor recién lanzado y comprobar el estado de la tarea de entrenamiento. De este modo podrá comprobar la implementación de la técnica de seguridad solicitada.