# Propuesta de proyecto

Universidad Europea - https://universidadeuropea.com/

## Sistema de Inferencia de Vertidos en Ríos usando Simulación Hidrodinámica Acelerada y DeepONet

Duo Xu Ye, Tadeo Adrián Troncoso Taraborreli, Víctor Pablo Velayos Velayos

Universidad Europea de Madrid, C/Tajo, s/n, Villaviciosa de Odón, Madrid 28670, España

## INFORMACIÓN

*Palabras clave:*

*Problema Inverso (Inverse Problem)*

*DeepONet (Deep Operator Network)*

*Gemelo Digital (Digital Twin)*

*Detección De Vertidos*

*Problemas Mal Condicionados (Ill-Posed Problems)*

*Generación De Datos Sintéticos*

*Simulación Hidrodinámica*

*Transporte De Contaminantes*

*CUDA / GPGPU*

*Computación De Alto Rendimiento (HPC)*

*Calibración De Modelos*

*Aprendizaje De Operadores (Operator Learning)*

*Java / JNI*

*Paralelismo (Embarrassingly Parallel)*

## RESUMEN

La presente propuesta describe un proyecto de investigación computacional orientado a la identificación y localización de eventos de vertido no autorizados en sistemas fluviales. El objetivo central es el desarrollo de una Red de Operador Profundo (DeepONet) diseñada para resolver el problema inverso.

Este problema es intrínsecamente mal condicionado (*ill-posed*), ya que busca inferir un campo de fuentes de contaminantes de alta dimensionalidad (con más de 1.800 puntos de vertido autorizados identificados, más un número indefinido de fuentes no registradas) a partir de datos de sensores espacialmente dispersos y de baja dimensionalidad (32 estaciones de medición). El DeepONet se empleará para aprender el operador inverso, mapeando la función de respuesta temporal de los sensores (Branch Net) a la función espacial que describe la localización e intensidad del vertido (Trunk Net).

Ante la escasez de datos empíricos etiquetados, la metodología se fundamenta en la generación de un conjunto de datos sintético a gran escala. Este se producirá mediante un "gemelo digital" que acopla un simulador hidrodinámico (basado en la ecuación de Manning) con un modelo de transporte de contaminantes (advección-difusión-reacción). Un componente crítico es la calibración del estado base de dicho gemelo, para la cual se realizará un análisis de datos que integra series temporales (calidad del agua y aforo de la CHT) con datos geoespaciales (censo de vertidos). La viabilidad computacional para generar miles de escenarios se garantiza mediante una aceleración masiva en C++/CUDA, posible gracias a la naturaleza "vergonzosamente paralelizable" (*embarrassingly parallel*) de los cálculos hidrodinámicos. Todo el proceso es orquestado desde una aplicación principal en Java a través de JNI.

Este enfoque híbrido, que combina la simulación física de alto rendimiento con el aprendizaje de operadores, permitirá crear un sistema robusto de inferencia capaz de localizar eventos de contaminación en tiempo casi real, superando las limitaciones de los sistemas de monitorización actuales.

*Duo Xu Ye – duo@pemail.es*

Revisada el 22 de octubre 2025

# Índices

## Índice de contenidos

# Objetivo General y Objetivos Específicos

## Objetivo general

Desarrollar e implementar un modelo de aprendizaje profundo, basado en una arquitectura **DeepONet (Red de Operador Profundo)**, capaz de resolver el problema inverso de la detección de vertidos en sistemas fluviales. El sistema inferirá un "mapa de calor" de la localización e intensidad de un evento de vertido en tiempo casi real, utilizando como única entrada los datos de series temporales de la red de sensores de calidad del agua existente (32 estaciones).

## Objetivos específicos

- **OE1: Desarrollo del Gemelo Digital.** Construir un simulador de alto rendimiento que acople un modelo hidrodinámico (basado en la ecuación de Manning) con un modelo de transporte de contaminantes (advección-difusión-reacción). La arquitectura combinará Java (orquestación), C++ (lógica nativa) y CUDA (paralelización de cómputo en GPU) vía JNI.
- **OE2: Adquisición y Análisis de Datos para Calibración.** Obtener y procesar datos reales de la Confederación Hidrográfica del Tajo (CHT), incluyendo series temporales de calidad del agua (SAICA) y aforos (SAIH) mediante *web scraping*, y datos geoespaciales del Censo de Vertidos Autorizados (+1800 puntos). Estos datos se analizarán para calibrar el estado base del gemelo digital.
- **OE3: Generación de Dataset Sintético.** Utilizar el gemelo digital acelerado por GPU para generar un dataset sintético a gran escala (target: >100.000 muestras). Cada muestra consistirá en un par: (a) la función de respuesta temporal de los 32 sensores y (b) la función espacial del vertido que la originó.
- **OE4: Entrenamiento y Validación del DeepONet.** Diseñar, implementar y entrenar el modelo DeepONet sobre el dataset sintético. El modelo aprenderá el mapeo del operador inverso (de lecturas de sensor a mapa de vertido). Se validará su precisión y capacidad de generalización contra un conjunto de prueba reservado.
- **OE5: Desarrollo de un Prototipo Funcional.** Desarrollar la interfaz de usuario principal como una aplicación web (Dashboard y configurador de ríos) usando **React.js**. Para la entrega de este proyecto, se integrará dicha aplicación web dentro de un contenedor de escritorio **JavaFX (OpenJFX 25)**, utilizando su componente **WebView** para enlazar la interfaz web con el *backend* de simulación Java.

# Descripción del Problema y Contexto

## Un Desafío Inverso y Mal Condicionado

La identificación y localización de eventos de vertido no autorizados o accidentales en grandes cuencas fluviales es un problema inverso de alta complejidad técnica. Cuando una estación de medición aguas abajo detecta una anomalía en la calidad del agua (el efecto), el contaminante (la causa) ya ha sido transportado, mezclado y difundido por el flujo del río durante un periodo de tiempo desconocido.

Este desafío se magnifica al ser un problema intrínsecamente mal condicionado (*ill-posed*)**.** En el caso de estudio de la cuenca del Tajo, se dispone de un número muy limitado de puntos de observación (aprox. 32 estaciones de calidad del agua) para inferir un campo de fuentes de alta dimensionalidad (más de 1.800 puntos de vertido autorizado conocidos, más un número indefinido de focos ilegales). Con los métodos actuales, es computacional y logísticamente inviable determinar el origen del vertido con la rapidez necesaria.

## Relevancia y Justificación del Proyecto

La relevancia de este proyecto es triple:

- Relevancia Social y Ambiental: La contaminación fluvial tiene un impacto directo e inmediato en la salud pública (afectando la potabilidad del agua) y en la integridad de los ecosistemas acuáticos. Una detección rápida es fundamental para activar alertas y proteger los puntos de captación de agua.
- Relevancia Operativa (Utilidad): Los organismos de gestión de cuencas carecen de herramientas proactivas. Actualmente, la detección es reactiva, lo que retrasa las medidas correctoras, incrementa el coste de la remediación y dificulta enormemente la imputación de responsabilidades
- Relevancia Técnica y Científica: Abordar un problema físico *ill-posed* mediante el aprendizaje de operadores (DeepONet), en lugar de con métodos de asimilación de datos tradicionales (que serían demasiado lentos), representa un enfoque novedoso y puntero en la aplicación de la IA.

## Solución Esperada y Beneficiarios

La solución esperada es un **sistema de soporte a la decisión (DSS)** basado en el modelo DeepONet entrenado. El sistema se activará automáticamente mediante alertas (por ejemplo, al detectar anomalías o valores fuera de rango en la telemetría de los sensores).

Una vez activado, el sistema ingerirá las series temporales recientes de las 32 estaciones y generará, en tiempo casi real, un "mapa de calor" probabilístico sobre el dominio del río. Este mapa identificará las zonas de origen del vertido más probables y su intensidad estimada, permitiendo una respuesta inmediata y focalizada.

Los **beneficiarios directos** de esta solución son:

1. **Organismos de Gestión de Cuencas:** Como la **Confederación Hidrográfica del Tajo (CHT)**, que podrá optimizar sus recursos de inspección, dirigiéndolos inmediatamente a la zona de alta probabilidad tras recibir una alerta.
2. **Servicios de Protección Ambiental:** Como el **SEPRONA**, al disponer de una herramienta que agiliza la investigación y la recopilación de pruebas.
3. **Empresas de Suministro de Agua:** Que podrán ser alertadas proactivamente de un riesgo aguas arriba de sus puntos de captación.

# Fuentes de datos y justificación de su elección

Para abordar este proyecto, se utiliza un enfoque híbrido que combina datos observados del mundo real (para análisis y calibración) con un dataset sintético a gran escala (para el entrenamiento del modelo).

## Datos Observados (Para Calibración y Contextualización)

Estos datos son la base para entender el problema y asegurar que nuestro simulador ("gemelo digital") opere bajo condiciones realistas.

- **Fuente y Método de Adquisición:**
  - **Datos de Calidad del Agua (SAICA) y Aforo (SAIH):** Los datos de series temporales de las 32 estaciones de medición se obtienen de los portales de datos abiertos de la Confederación Hidrográfica del Tajo (CHT). Para ello, se han desarrollado *scripts* de *web scraping* en Python que, mediante ingeniería inversa de las peticiones XHR, acceden directamente a los *endpoints* JSON de la plataforma, permitiendo una descarga eficiente de la serie de los últimos 10 días

    - **Censo de Vertidos Autorizados:** Se utiliza el censo oficial de la CHT (`Censo vertidos.-xlsx`), que identifica más de 1.800 puntos de vertido autorizados en la cuenca.

- Enlaces Exactos a los Datasets:
    - *Portal SAIH (Aforos): https://saihtajo.chtajo.es/index.php?url=/tr/mapas/ambito:PL/mapa:H1#nav*
    - *Portal SAICA (Calidad del Agua): https://saihtajo.chtajo.es/index.php?url=/tr/mapas/ambito:PL/mapa:H1#nav*
    - *Censo de Vertidos: https://www.chtajo.es/censo-de-vertidos*
- *Descripción del Contenido:*
    - ***Datos de Sensores:*** *Se obtienen múltiples variables clave (pH, Temperatura, Amonio, Nitratos, O2 Disuelto, Conductividad, etc.) en formato JSON. Los ficheros brutos se almacenan en* `data/datasets/raw/`
    - ***Censo de Vertidos:*** Fichero Excel con +1.800 registros, incluyendo (en su mayoría) coordenadas y tipología industrial del vertido.
- Transformación y Limpieza Estimada
    - **Datos de Sensores:** El análisis exploratorio inicial (EDA) y la limpieza implicarán la interpolación de datos faltantes, el tratamiento de valores atípicos y la agregación temporal y por estación.
    - **Censo de Vertidos:** Este dataset requiere un **enriquecimiento de datos** significativo. Se están utilizando flujos de trabajo automatizados (n8n) y herramientas de búsqueda web (Perplexity) para verificar el estado de las empresas y completar la información faltante.

# Datos Sintéticos (Para Entrenamiento del DeepONet)

*Fuente y Justificación*

- **Fuente:** Este dataset será generado íntegramente por nuestro **gemelo digital** (el simulador hidrodinámico y de transporte Java/JNI/CUDA).
- **Justificación:** El objetivo del proyecto es detectar vertidos *ilegales*, de los cuales, por definición, **no existen datos etiquetados**. Es imposible saber cuándo, dónde y cuánta cantidad se vertió en un evento real pasado.
- Además, para que el DeepONet aprenda a resolver el **problema *ill-posed*** (inferir +1.800 fuentes desde 32 sensores), se requiere un conjunto de entrenamiento masivo (>100.000 muestras) que cubra una vasta combinación de escenarios (diferentes ubicaciones, intensidades, duraciones, caudales del río, etc.). Esto solo es factible mediante la simulación de alto rendimiento.

*Descripción del Contenido*

- **Formato:** Pares de `(Input_Function, Output_Function)`.
- **Tamaño Estimado:** > 100.000 muestras.
- **Variables de Entrada (Branch Net):** Series temporales simuladas de las 32 estaciones de sensores (ej. vectores de `[32 sensores x 48 horas]`).

- **Variables de Salida (Trunk Net):** La función o "mapa de calor" que describe la ubicación e intensidad del vertido que generó esas lecturas.

# Metodología prevista

La metodología del proyecto sigue un proceso estructurado en fases, diseñado para abordar el desafío central: la escasez de datos de entrenamiento y la naturaleza *ill-posed* del problema.

## *Fases del Proceso*

El proyecto se divide en las siguientes etapas clave, alineadas con los objetivos específicos:

1. **Adquisición y Análisis (OE2):** Obtención de datos reales (sensores y censo) mediante los *scrapers* de Python y análisis exploratorio (utilizando *notebooks* como `Análisis del Censo de autorizados a verter.ipynb`) para comprender el dominio y obtener los parámetros de calibración.

2. **Desarrollo del Gemelo Digital (OE1):** Creación del simulador físico. Esta es la etapa más crítica y se detalla a continuación.

3. **Generación de Datos Sintéticos (OE3):** Uso del simulador para crear el dataset de entrenamiento.

4. **Modelado y Evaluación (OE4):** Entrenamiento y validación del modelo DeepONet.

5. **Prototipado (OE5):** Integración del modelo en un sistema funcional.

## *Metodología del Gemelo Digital (OE1)*

El "gemelo digital" es un sistema híbrido que simula la hidrodinámica (cómo fluye el agua) y el transporte de contaminantes (cómo se mueven los vertidos).

- **Arquitectura:** La orquestación de la simulación se maneja en **Java**, que se comunica vía **JNI (Java Native Interface)** con un *solver* (solucionador) numérico de alto rendimiento escrito en **C++/CUDA**.

- **Simulación Física:**

    1. **Hidrodinámica:** Se resuelve la **ecuación de Manning** para calcular el calado (H) y la velocidad (V) del agua en cada sección del río.

    2. **Transporte:** Sobre este flujo, se simula el transporte de contaminantes mediante la ecuación de **advección-difusión-reacción**.

- **Metodología de Optimización (Viabilidad):** La generación de >100.000 escenarios es un desafío computacional masivo (Riesgo: Tiempo de Cómputo). La metodología para mitigar esto se basa en la optimización extrema del *solver* de CUDA (`manning_kernel.cu`):

    1. Se explota la naturaleza **"vergonzosamente paralelizable"** del problema.

    2. **Eliminación de Divergencia de Ramas:** Se ha reescrito el código del kernel para operar **sin sentencias `if`** en el bucle principal. Las comprobaciones (ej. división por cero) se manejan mediante aritmética segura (añadiendo `epsilon`) y saneamiento de datos en la entrada.

    3. **Precisión de Datos:** Se reduce la precisión de los cálculos de `FP64` (doble, estándar en CPU) a `FP32` (simple), que es mucho más rápida en GPU (64:1 en arquitecturas Blackwell RTX5090), sin pérdida significativa de precisión para este problema físico (se estima en 10E-4)

4. **Optimización de Transferencia de Datos:** Para minimizar la latencia CPU-GPU, se implementarán **copias de memoria asíncronas** y se usará *pinned memory*. Esto permite que el motor de **Acceso Directo a Memoria (DMA)** solape cómputo y comunicación.

5. **Profiling y Escalabilidad de Cómputo:** Se utiliza **Nvidia Nsight** para perfilar el kernel y determinar la configuración óptima de hilos y *batch size* para la GPU local (RTX 5090). Si los tiempos de generación lo requieren, el plan contempla la **escalabilidad en la nube**, alquilando tiempo de cómputo en GPUs de última generación (ej. **Nvidia B200**) para asegurar la finalización del dataset.

*Metodología de Modelado (OE4)*

- **Modelo de Aprendizaje Automático:** Se utilizará una **Red de Operador Profundo (DeepONet)**.
- **Justificación de la Elección:**
    - **Problema *Ill-Posed*:** Un modelo tradicional fracasaría al intentar mapear un vector de 32 sensores a uno de +1.800 posibles fuentes.
    - **Aprendizaje de Operadores:** El DeepONet está diseñado para aprender **operadores** (un mapeo entre espacios funcionales). En nuestro caso:
        1. La **Branch Net** recibirá la *función de respuesta del sensor* (los datos de las 32 series temporales).
        2. La **Trunk Net** recibirá la *función de consulta espacial* (coordenadas del río).
        3. La salida del modelo será la *función de mapa de calor* del vertido.

- **Procesamiento y Entrenamiento Distribuido:** Dado el volumen masivo del dataset sintético (cientos de miles de muestras de alta dimensionalidad), la metodología contempla el uso de **Apache Spark** para el procesamiento distribuido (ETL) de los datos. Asimismo, si el entrenamiento del DeepONet se convierte en un cuello de botella, se utilizarán *frameworks* de entrenamiento distribuido (ej. Horovod sobre Spark) para paralelizar la carga de trabajo en un clúster.

- **Evaluación y *Baselines*:** La robustez del modelo DeepONet se evaluará rigurosamente sobre un conjunto de prueba sintético, usando el Error Cuadrático Medio (MSE) entre el mapa de calor predicho y el real. Para contextualizar su rendimiento, el modelo se comparará contra dos modelos de referencia (*baselines*):
    - **Heurística Simple:** Un algoritmo básico que simule una estrategia de "remontar el río", como una triangulación basada en los tiempos de llegada de la anomalía a los diferentes sensores.
    - **Modelo SOTA Alternativo (PINN):** Se entrenará una **Red Neuronal Informada por la Física (PINN)**. A diferencia del DeepONet (que aprende el operador a partir de los datos), el PINN intentará resolver el problema inverso incorporando las Ecuaciones Diferenciales Parciales (EDPs) del transporte de contaminantes directamente en su función de pérdida. Esta comparación determinará la eficiencia y precisión de nuestro enfoque de aprendizaje de operador (DeepONet) frente a un enfoque basado en la física de la pérdida (PINN) para este problema específico.

# Herramientas y Tecnologías

*Herramientas y Tecnologías*

La naturaleza híbrida del proyecto, que combina simulación física de alto rendimiento, gestión de bases de datos, desarrollo web *full-stack* y *deep learning*, requiere un conjunto de herramientas especializado y moderno.

- **Arquitectura y Simulación (Gemelo Digital):**
    - **Java (JDK 21):** Lenguaje principal para la orquestación del simulador, gestión de configuraciones y la lógica de negocio del *backend*.
    - **C++ (C++17):** Utilizado para escribir el *solver* numérico nativo.
    - **NVIDIA CUDA 12+:** La tecnología clave para la aceleración del *solver* hidrodinámico en GPU.
    - **JNI (Java Native Interface):** El "pegamento" que permite la comunicación de alto rendimiento entre la JVM (Java) y el código nativo (C++/CUDA).
    - **Nvidia Nsight Systems:** Herramienta de *profiling* para analizar el rendimiento del kernel de CUDA.
    - **Gradle:** Sistema de automatización de la compilación (`build.gradle`) que maneja el complejo proceso de compilación de Java 21, C++ y el enlazado de las librerías nativas.
- **Stack de Backend y Datos:**
    - **Python (FastAPI & Uvicorn):** Utilizado como microservicio de inferencia de alto rendimiento para servir los modelos DeepONet/PINN entrenados.
    - **PostgreSQL:** Sistema gestor de base de datos relacional (datos del censo, resultados, etc.).
    - **Redis:** Base de datos en memoria de alta velocidad, usada para el cacheo de resultados de simulación, gestión de sesiones y como *message broker* si es necesario.
    - **Kong API Gateway:** Actúa como punto de entrada único (*Single Point of Entry*), gestionando, securizando y enrutando las peticiones de la UI a los microservicios correspondientes (el *backend* de Java y el *backend* de inferencia de FastAPI).
    - **JDBI3 & HikariCP:** Stack de persistencia de datos de alto rendimiento para Java.
    - **SLF4J/Logback & Jackson:** Utilizados para un logging robusto y la serialización/deserialización de datos y configuraciones JSON.
- **Interfaz de Usuario (UI) y Prototipado (OE5):**
    - **React.js (HTML5/CSS3/JavaScript):** La interfaz de usuario principal (dashboard, configurador de ríos, visualizador de mapas de calor) se desarrolla como una aplicación web moderna.
    - **Nginx:** Servidor web y *reverse proxy* de alto rendimiento utilizado para servir la aplicación React.js.
    - **VPS Hosting:** La aplicación web estará alojada en un Servidor Privado Virtual (VPS).
    - **JavaFX (OpenJFX 25):** Utilizado como un **contenedor de escritorio**. El prototipo para este proyecto empleará el componente **WebView** de JavaFX para renderizar la aplicación web de

React.js alojada, permitiendo una integración limpia y un puente de comunicación nativo (Java-a-JavaScript) entre el *frontend* (web) y el *backend* (Java).

- **Stack de Ciencia de Datos (Python):**
    - **Python 3.10+:** Lenguaje central para *scraping*, análisis exploratorio y modelado de ML.
    - **Pandas / Geopandas:** Librería fundamental para la manipulación y análisis de datos tabulares (ej. el Censo de Vertidos y los resultados de los *scrapers*).
    - **Matplotlib / Seaborn:** Librerías de visualización para el Análisis Exploratorio de Datos (EDA).
    - **Jupyter Notebooks:** Entorno interactivo para el análisis de datos.
    - **Requests :** Herramientas de elección para *parsear* sitios estáticos y para realizar ingeniería inversa de *endpoints* de API/XHR (el método más eficiente, ya validado en el proyecto).
        - **Playwright / Selenium:** *Frameworks* de simulación de navegador (headless browser)
    - **n8n (N-node-N):** Plataforma *low-code* para el flujo automatizado de enriquecimiento de datos.
    - **Apache Spark:** Contemplado para el procesamiento (ETL) distribuido del dataset sintético.
- **Machine Learning (Modelado):**
    - **Python (TensorFlow / PyTorch):** *Frameworks* de elección para el diseño y entrenamiento de los modelos DeepONet y PINN.
    - **Plataformas de Cómputo en la Nube (AWS/GCP):** Contempladas para el alquiler de GPUs de alto rendimiento (ej. **Nvidia B200**).
- **Testing y Calidad:**
    - **JUnit 5, Mockito & AssertJ:** Stack estándar para *testing* unitario y de integración en el ecosistema Java.
    - **Testeo de GPU (`@Tag("GPU")`):** El `build.gradle` define una tarea personalizada (`gpuTest`) para aislar los tests que requieren hardware de GPU nativo.

### *Justificación de la Elección (No-Python)*

Aunque la asignatura se centra en Python, la arquitectura del proyecto utiliza un enfoque políglota de "la herramienta adecuada para el trabajo adecuado":

1. **Python (Ciencia de Datos y MLOps):** Se usa donde es el líder indiscutible: EDA (*Pandas*), *scraping*, *notebooks* (*Jupyter*), entrenamiento de ML (*TensorFlow/PyTorch*) y servicio de modelos de inferencia (*FastAPI*).

2. **Java/C++/CUDA (HPC y Backend):** Esta combinación se elige para el núcleo del proyecto (generación de datos y orquestación) por razones que Python no puede cubrir:
    - **Rendimiento Extremo (HPC):** La generación de >100.000 escenarios físicos es inviable en Python. Se requiere C++/CUDA para optimizaciones de bajo nivel (código *branchless*, `FP32`, DMA).
    - **Backend Robusto:** Java 21 y su ecosistema (JDBI, HikariCP) proporcionan una base robusta y concurrente para la lógica de negocio principal.

3. **Prototipo Integrado:** El uso de **JavaFX WebView** permite que el prototipo de escritorio (OE5) cargue la interfaz web principal (React.js), proporcionando un puente de comunicación nativo y entregando una experiencia de usuario moderna.

# Plan de Trabajo y Roles del Equipo

*Contexto Multi-Asignatura del Proyecto*

Es importante destacar que este es un proyecto de gran escala y ambición, concebido para ser desarrollado y presentado en el contexto de **múltiples asignaturas** del Grado. Su arquitectura completa abarca áreas como Computación de Alto Rendimiento, Desarrollo de Backend/Frontend e Inteligencia Artificial.

Para la presente asignatura, **"Proyecto de Computación I"**, el alcance de la evaluación se centrará exclusivamente en el **componente de Ciencia y Análisis de Datos**. Las fases de desarrollo de software (el *solver* CUDA, la arquitectura de microservicios, la UI en React, etc.) se presentan como contexto para justificar la generación y el uso de los datos, pero no forman parte del trabajo a evaluar aquí.

## Roles del Equipo (Para "Proyecto de Computación I")

El equipo para esta asignatura está enfocado en el rol de Científicos de Datos

- **Duo Xu Ye:**
    - **Rol Principal (Fuera de esta asignatura):** Arquitecto del Sistema y desarrollador principal del *stack* tecnológico completo (HPC, backend, frontend).
    - **Rol en esta asignatura:** Líder del equipo de datos. Responsable de la integración de los datos con el simulador y del diseño de los experimentos de modelado.
- **Tadeo Adrián Troncoso Taraborreli:**
    - **Rol en esta asignatura:** Analista de Datos y Especialista en ETL. Responsable de la adquisición, limpieza y enriquecimiento de los datos de la CHT y de los nuevos conjuntos de datos (embalses, cartografía).
- **Víctor Pablo Velayos Velayos:**
    - **Rol en esta asignatura:** Analista de Datos y Modelado. Responsable del Análisis Exploratorio de Datos (EDA), la visualización y el desarrollo de los modelos de inferencia *baseline*.

*Ampliación del Alcance de Análisis de Datos*

Para enriquecer el componente de ciencia de datos de este proyecto, se incorporarán las siguientes tareas y fuentes de datos adicionales:

1. **Integración de Nuevos Datasets:** Además de los datos de la CHT, se analizarán:
    - **Datos de Embalses:** Se obtendrán y analizarán los datos de nivel y volumen de los embalses principales de la cuenca. El objetivo es modelar su impacto en el caudal y en parámetros de calidad del agua aguas abajo (ej. temperatura, dilución).
    - **Datos Cartográficos y Geológicos (IGN):** Se utilizarán capas de información geográfica del Instituto Geográfico Nacional para analizar la correlación entre la geología del terreno, las zonas de inundación y los puntos de vertido, enriqueciendo el análisis de riesgos.

2. **Desarrollo de Modelos de Inferencia *Baseline*:** Antes de abordar los modelos complejos (DeepONet/PINN), el equipo desarrollará y evaluará modelos más simples (ej. **Gradient Boosting, Redes Neuronales Densas**) para resolver sub-problemas concretos, como:

    - Predecir la temperatura del agua en un punto a partir del nivel del embalse y datos meteorológicos.
    - Estimar la concentración de un contaminante en un sensor basándose en el caudal y las lecturas de sensores aguas arriba. Estos modelos servirán como una valiosa referencia para contextualizar el rendimiento de los modelos más avanzados.

### *Plan de Trabajo (Cronograma de 8 Semanas para Análisis de Datos)*

Este cronograma se centra exclusivamente en las tareas de Ciencia de Datos (el alcance de esta asignatura) a realizar por el equipo, asumiendo que el *solver* de HPC (OE1) y el *scraper* de contaminantes ya están operativos.

- **Semanas 1-2: Finalización de la Adquisición de Datos**
    - **Objetivo:** Completar la capa de adquisición de datos para tener todos los *datasets* necesarios.
    - **Tareas:**
        - Desarrollar el *scraper* para datos de Aforos (SAIH). (Responsable: Tadeo)
        - Desarrollar el *scraper* para datos de Embalses. (Responsable: Tadeo)
        - Desarrollar el *scraper* para datos Cartográficos y Geológicos (IGN). (Responsable: Víctor)
        - Desarrollar el *scraper* para la conexión a datos en tiempo real. (Responsable: Duo)
    - **Hito:** Todos los datos brutos (Contaminantes, Aforos, Embalses, Censo, IGN) están consolidados en la base de datos PostgreSQL.
- **Semanas 3-4: Análisis Exploratorio de Datos (EDA) Intensivo**
    - **Objetivo:** Comprender profundamente los datos, encontrar correlaciones y preparar el *feature engineering*. **Esta es la tarea central que está pendiente.**
    - **Tareas:**
        - EDA y visualización de series temporales (contaminantes, aforos, embalses). Búsqueda de correlaciones, estacionalidad y latencias. (Responsable: Víctor)
        - EDA geoespacial: Cruce de datos del Censo de Vertidos con las capas cartográficas del IGN (zonas inundables, geología). (Responsable: Tadeo)
        - Creación de *notebooks* maestros de análisis y documentación del proceso de limpieza de datos. (Responsable: Duo)
    - **Hito:** Informe de EDA completado, con las principales hipótesis y variables identificadas.
- **Semanas 5-6: Desarrollo de Modelos *Baseline***
    - **Objetivo:** Evaluar la predictibilidad de los sub-problemas identificados.
    - **Tareas:**
        - *Feature Engineering* basado en los hallazgos del EDA. (Responsable: Tadeo)

- Entrenar y evaluar los modelos de inferencia *baseline* (ej. Gradient Boosting, Redes Densas) para predecir variables clave (ej. Tª del agua, nivel de embalse). (Responsable: Víctor)

- **Hito:** Modelos *baseline* entrenados y evaluados.

- **Semana 7: Preparación de Datasets Finales**
  - **Objetivo:** Preparar los conjuntos de datos limpios para el gemelo digital y los modelos avanzados.
  - **Tareas:**
    - Crear los *datasets* de calibración para el gemelo digital. (Responsable: Duo)
    - Documentar el *data lineage* y las transformaciones finales. (Responsable: Tadeo, Víctor)
  - **Hito:** *Datasets* de calibración y modelado generados y documentados.
- **Semana 8: Documentación y Entrega Final**
  - **Objetivo:** Consolidar el trabajo de la asignatura.
  - **Tareas:**
    - Redacción del informe final del proyecto (enfocado en el análisis de datos y los modelos *baseline*).
    - Limpieza y entrega de los *notebooks* de análisis y modelado.
  - **Hito:** Entrega final del proyecto de la asignatura