INTERFAZ GRÁFICA PARA UNA PLATAFORMA DE SIMULADORES DE PROCESOS DE USO ACADÉMICO

Trabajo de Fin de Grado

INGENIERÍA INFORMÁTICA

VNiVERSiDAD D SALAMANCA

Septiembre de 2023

ALUMNO:

ÁLVARO GARCÍA LABRADOR

70913088V

_____

TUTORES

MARIO FRANCISCO SUTIL

PASTORA ISABEL VEGA CRUZ

_____

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

Facvltad D Ciencias
VNiVERSiDAD D SALAMANCA

Certificado del/los tutor/es TFG

D./Dña. MARIO FRANCISCO SUTIL / PASTORA VEGA CRUZ profesores del Departamento de Informática y Automática de la Universidad de Salamanca,

HACE/N CONSTAR:

Que el trabajo titulado " Interfaz gráfica para una plataforma de simuladores de uso académico", que se presenta, ha sido realizado por Álvaro García Labrador, con DNI ****3088V y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado en Informática en esta Universidad.

Salamanca, 6 de septiembre de 2023

Fdo.: Mario Francisco Sutil /Pastora Vega Cruz

Mario Francisco Sutil
Firmado digitalmente por Mario Francisco Sutil
Fecha: 2023.09.06 10:58:46 +02'00'

VEGA CRUZ PASTORA ISABEL - 07833143X
Firmado digitalmente por VEGA CRUZ PASTORA ISABEL - 07833143X
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-07833143X, givenName=PASTORA ISABEL, sn=VEGA CRUZ, cn=VEGA CRUZ PASTORA ISABEL - 07833143X
Fecha: 2023.09.06 17:37:34 +02'00'

Índice

Ilustraciones

1. Resumen y palabras clave

1.1 Resumen

Hoy en día la mayoría de las industrias requieren de un sistema de control para regular las variables de interés del proceso que se lleve a cabo. Estos sistemas de control en lazo cerrado se implementan mediante algoritmos software, tomando la señal procedente de sensores y enviando la acción de control calculada a los actuadores (bombas, válvulas, motores, etc.). Dado que son sistemas complejos, hay que sintonizarlos y ajustarlos adecuadamente antes de su implementación en una planta real.

La simulación digital de sistemas es una disciplina que permite realizar pruebas sobre un sistema y evaluar su desempeño con seguridad y rapidez, siendo útil para poder modificar los diferentes parámetros de ajuste y probar el funcionamiento tanto en lazo abierto como en lazo cerrado. En este Trabajo Fin de Grado se ha desarrollado una interfaz gráfica que permita simular diferentes plantas sencillas e industriales de forma sencilla, para ser utilizada a nivel didáctico por estudiantes que no dispongan de conocimientos avanzados del software utilizado ni de técnicas de modelado y simulación, pero que deseen evaluar el funcionamiento de los sistemas propuestos.

La interfaz abarca desde sistemas eléctricos, hidráulicos o mecánicos básicos, hasta una simulación del proceso de fangos activados en una planta depuradora de aguas residuales, considerando los modelos matemáticos BSM1 (Benchmark Simulation Model Nº1). Entre otras cosas, desde la interfaz se puede también realizar comparativas entre diferentes simulaciones, así como guardar los resultados de las sesiones realizadas.

Palabras Clave

- Modelado y Simulación
- Modelo de Planta Depuradora de Aguas Residuales
- MATLAB
- Simulink
- Interfaz

1.2 Summary

Nowadays, most industries require a control system to regulate the variables of interest in the process being carried out. These closed-loop control systems are implemented using software algorithms, taking signals from sensors and sending the calculated control action to actuators (pumps, valves, motors, etc.). As these are complex systems, they need to be tuned and adjusted properly before being implemented in a real plant.

Digital system simulation is a discipline that allows testing a system and evaluating its performance safely and quickly. It is useful for modifying different tuning parameters and testing the operation in both open-loop and closed-loop modes. In this Bachelor's Thesis, a graphical interface has been developed to easily simulate various simple and industrial plants. It is intended for didactic use by students who may not have advanced knowledge of the used software or modeling and simulation techniques but wish to evaluate the performance of proposed systems.

The interface covers a range of systems, from basic electrical, hydraulic, or mechanical systems to a simulation of the activated sludge process in a wastewater treatment plant, considering the mathematical models BSM1 (Benchmark Simulation Model Nº1). Among other features, the interface allows for comparisons between different simulations and saving the results of the sessions conducted.

Key Words

- Modeling and Simulation
- Wastewater Treatment Plant Model
- MATLAB
- Simulink
- Interface

2. Introducción

Hoy en día la automatización de procesos está abarcando la mayoría de los sectores de servicios, de la comunicación, del entretenimiento, de la agricultura, ganadería, etc. Todos estos servicios requieren de sistemas de control que automaticen los procesos y no tengan que depender de la mano humana. No es ninguna sorpresa que la automatización es algo que ha mejorado los sectores mencionados anteriormente, pero hay un sector en concreto que depende mucho de estos sistemas de control y es en el que nos vamos a centrar en este proyecto, sector de la industrialización.

En la actualidad la mayoría de las empresas industriales viven gracias a estos sistemas de control que permiten la automatización de los procesos sin la supervisión de un encargado (mano de obra). La puesta a punto e implementación de estos sistemas es complicada, dado que se manejan muchas variables y los procesos a veces tienen dinámicas complejas.

Por ello, se necesita de la simulación para observar las consecuencias que se tienen en los distintos modelos, y para ello se ha desarrollado una interfaz gráfica para trabajar de forma segura y efectiva.
Así los alumnos pueden entender mejor los diferentes modelos que nos podemos encontrar y modificar sus diferentes variables para observar los cambios y consecuencias que tiene el proceso.
La aplicación está divida en: Modelos Básicos y Modelos Avanzados.

Los modelos básicos están configurados en lazo abierto, es decir, no hay ningún controlador que esté actuando. De esta forma se puede estudiar el comportamiento del sistema, previo a la implementación de un sistema de control en lazo cerrado.
Estos modelos básicos estarían conformados por: Circuito RC, Péndulo y Tanques Acoplados.
Cada modelo permite la modificación de algunas variables, la simulación del proceso en un intervalo de tiempo determinado y por último la visualización del resultado de la simulación.

Aparte de estos modelos básicos tendríamos también un modelo avanzado que sería el de la Depuradora de Agua, concretamente el proceso de fangos activados representado mediante la plataforma de simulación

(benchmark) BSM1, este modelo sería el más complejo y el que más funcionalidades ofrecería al usuario.
El desarrollo de este proyecto gira entorno a la aplicación MATLAB, pues con ella vamos a implementar la interfaz para sus distintas utilidades (simular, visualizar resultados, almacenar resultados…)
Las dos aplicaciones centrales que he usado para la realización del proyecto son:

- App Designer MATLAB: una herramienta que proporciona MATLAB para el diseño de interfaces gráficas.
- Simulink: otra herramienta de MATLAB que permite modelar y simular sistemas mediante bloques individuales, o agrupaciones de bloques (subsistemas), y las conexiones entre ellos.

3. Objetivos del proyecto

El objetivo de este trabajo fin de Grado es el desarrollo de una interfaz gráfica para la simulación digital de sistemas con el fin de facilitar la realización de simulaciones a estudiantes que no tengan conocimientos avanzados sobre modelado y simulación de sistemas. La interfaz gráfica incluirá la posibilidad de utilizar ciertos modelos sencillos predefinidos, así como modelos avanzados correspondientes a un proceso de depuración de aguas residuales.

A continuación, se describirán los objetivos específicos
marcados por los requisitos y los objetivos técnicos, tenidos en cuenta para el desarrollo del proyecto.

3.1 Objetivos marcados por los requisitos

- Crear una interfaz gráfica orientada al alumnado con un conocimiento básico de modelado y simulación, que sea de fácil manejo, accesible y comprensible, facilitando la iniciación en estos temas.
- Permitir al usuario comenzar con modelos sencillos en lazo abierto para ir asimilando ciertos conceptos básicos que luego aparecerán en modelos más complejos
- Incluir modelos de distintos ámbitos: sistemas hidráulicos, mecánicos, eléctricos y químicos (depuración de aguas)
- Permitir la modificación de los parámetros más relevantes en cada planta para poder ver las consecuencias que tiene en la evolución de la simulación.
- Permitir el almacenamiento, carga y comparación de diferentes simulaciones, para el estudio de los resultados y por eficiencia, evitando tener que repetir simulaciones lentas con el consiguiente consumo de tiempo.
- La muestra de gráficas e índices de desempeño tanto de la planta como del controlador involucrado (si el sistema está en lazo cerrado) en el intervalo de tiempo que deseemos.
- Realizar un diseño de aplicación abierto a la ampliación de más modelos, ya sean básicos o más avanzados.

3.2 Objetivos técnicos

Como objetivos técnicos, cabe mencionar que la aplicación es ejecutable desde cualquier ordenador que tenga MATLAB instalada sin necesidad de una conexión a internet.

La aplicación se abre en una pestaña mediana ya que sus funcionalidades no requieren de una pantalla completa.

4. Conceptos teóricos

4.1 Modelado y Simulación de Sistemas

Simular digitalmente un sistema significa recrear su funcionamiento y comportamiento utilizando un modelo computacional. Implica desarrollar un modelo matemático del sistema en cuestión y luego utilizar software o herramientas de simulación para ejecutar ese modelo en una computadora, en mi caso esa herramienta de simulación será Simulink y MATLAB.

La simulación digital permite estudiar el comportamiento de un sistema en un entorno virtual antes de implementarlo en el mundo real. Puede ser utilizado en una amplia variedad de campos, como la ingeniería, la ciencia, la medicina, la economía y muchos otros.

Para simular digitalmente un sistema, se deben seguir varios pasos. Primero, se debe definir el propósito de la simulación y los objetivos que se desean lograr. Luego, se desarrolla un modelo matemático que describa las características y el comportamiento del sistema. Este modelo puede incluir ecuaciones, algoritmos o reglas específicas dependiendo del tipo de sistema que se esté simulando.

Una vez que se ha desarrollado el modelo, se implementa en el software utilizado, Simulink. Este software permite ingresar los parámetros iniciales del sistema y ejecutar la simulación. Durante la simulación, se recopilan y analizan datos para comprender cómo se comporta el sistema en diferentes condiciones.

La simulación digital proporciona una serie de beneficios. Permite probar y optimizar sistemas antes de invertir recursos en su implementación física. También puede ayudar a comprender mejor el comportamiento de sistemas complejos, identificar posibles problemas o fallos, y evaluar el impacto de cambios o modificaciones en el sistema.

En resumen, simular digitalmente un sistema implica recrear su funcionamiento en un entorno virtual utilizando modelos matemáticos y herramientas de simulación. Esto permite estudiar y analizar el sistema antes de implementarlo en el mundo real, proporcionando una valiosa información para la toma de decisiones y la optimización.

4.1.1 Modelos de Ecuaciones Diferenciales (ODE)

Un modelo matemático basado en ecuaciones diferenciales ordinarias (ODE) describe el comportamiento de un sistema utilizando una o varias ODE que relacionan las variables del sistema con sus derivadas con respecto al tiempo u otra variable independiente.

Un modelo matemático basado en ODE generalmente consta de los siguientes componentes:

- Variables del sistema: Estas son las cantidades o propiedades que se desea estudiar y que varían con el tiempo. Las variables pueden ser tanto entradas como salidas del sistema.

- Ecuaciones diferenciales: Estas ecuaciones representan las relaciones entre las variables y sus derivadas con respecto al tiempo. Pueden ser ecuaciones de primer orden o de orden superior, dependiendo del sistema y su complejidad. Cada ecuación describe cómo la variable cambia en función de las variables y sus derivadas.

- Condiciones iniciales: Son los valores de las variables en el tiempo inicial. Estas condiciones son necesarias para determinar la solución única de las ecuaciones diferenciales.

- Parámetros del modelo: Son constantes que aparecen en las ecuaciones diferenciales y pueden influir en el comportamiento del sistema. Los parámetros pueden representar tasas de crecimiento, tasas de interacción, coeficientes de transferencia, entre otros.

Una vez que se establece el modelo matemático basado en ODE, se pueden utilizar técnicas analíticas o numéricas para resolver las ecuaciones y obtener la solución del sistema. Las soluciones pueden proporcionar información valiosa sobre el comportamiento del sistema a lo largo del

tiempo, como tendencias, estabilidad, puntos de equilibrio, oscilaciones, entre otros.

Los modelos matemáticos basados en ODE se aplican en una amplia gama de campos, como la física, la química, la biología, la economía y la ingeniería, para comprender y predecir el comportamiento de sistemas dinámicos. Estos modelos permiten simular y analizar sistemas complejos, realizar predicciones y optimizar el diseño o control de los sistemas en estudio.

4.1.2 Resolución Matemática de Modelos

Método de Euler

El método de Euler es un método numérico simple utilizado para resolver ecuaciones diferenciales ordinarias (ODE). Aunque es un método básico, proporciona una aproximación numérica de la solución de una ODE al descomponerla en pequeños pasos.
El método de Euler aproxima la solución de la ODE al avanzar en pequeños incrementos de tamaño h a lo largo del dominio de tiempo. Cada paso de tiempo utiliza la pendiente calculada en el punto actual para estimar el valor de la función desconocida en el siguiente punto.

Es importante tener en cuenta que el método de Euler puede producir una aproximación menos precisa en comparación con métodos más avanzados, especialmente en ODE con comportamientos no lineales o con cambios rápidos. El error acumulado también puede aumentar a medida que se avanza en el tiempo. Por lo tanto, se recomienda usar tamaños de paso pequeños (h) para obtener resultados más precisos.

A medida que se reduce el tamaño del paso de integración, el método de Euler tiende a converger hacia la solución exacta de la ODE. Sin embargo, en casos donde se requiere una mayor precisión o cuando se trabaja con ODE más complejas, se recomienda utilizar métodos numéricos más avanzados, como el método de Runge-Kutta o métodos de paso variable.

Dada una ecuación diferencial de primer orden dy/dt = f (t, y) con una condición inicial y(t0) = y0, donde t representa la variable independiente e y es la función desconocida, el método de Euler se puede implementar de la siguiente manera:

- Establecer los parámetros iniciales:
  t0: Valor inicial de la variable independiente.
  y0: Valor inicial de la función desconocida.
  h: Tamaño del paso de integración.

- Iterar para cada paso de tiempo:

  - Calcular la pendiente de la función desconocida en el punto actual:
    m = f (t, y)

  - Actualizar los valores de t e y:
    t = t + h
    y = y + h * m

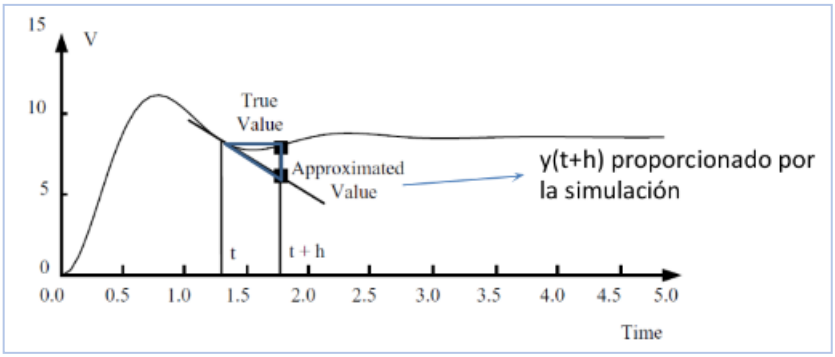


Figura 1 Método de Euler (Cellier et al., 2006)

Método de Runge-Kutta

Este método numérico de orden 4-5 es utilizado para resolver ecuaciones diferenciales ordinarias (ODE). El método combina dos esquemas de Runge-Kutta diferentes, uno de orden 4 y otro de orden 5, para obtener una aproximación más precisa de la solución de la ODE.

Dada una ODE de primer orden dy/dt = f (t, y) con una condición inicial y(t0) = y0, donde t representa la variable independiente e y es la función desconocida, el método de Runge-Kutta de orden 4-5 se puede implementar de la siguiente manera:

- Establecer los parámetros iniciales:
  t0: Valor inicial de la variable independiente.
  y0: Valor inicial de la función desconocida.
  h: Tamaño del paso de integración.

- Iterar para cada paso de tiempo:

  - Calcular los coeficientes de Runge-Kutta de orden 4:
    k1 = h * f (t, y)
    k2 = h * f (t + h/2, y + k1/2)
    k3 = h * f (t + h/2, y + k2/2)
    k4 = h * f (t + h, y + k3)

  - Calcular la aproximación de orden 4:
    y_4 = y + (k1 + 2k2 + 2k3 + k4) /6

  - Calcular los coeficientes de Runge-Kutta de orden 5:
    k1 = h * f (t, y)
    k2 = h * f (t + h/4, y + k1/4)
    k3 = h * f (t + 3h/8, y + 3k1/32 + 9*k2/32)
    k4 = h * f (t + 12h/13, y + 1932k1/2197 - 7200k2/2197 + 7296k3/2197)
    k5 = h * f (t + h, y + 439k1/216 - 8k2 + 3680k3/513 - 845k4/4104)

- Calcular la aproximación de orden 5:
  y_5 = y + (25k1/216 + 1408k3/2565 + 2197*k4/4104 - k5/5)

- Calcular el error estimado:
  error = |y_5 - y_4|

- Ajustar el tamaño del paso de integración:
  - Si el error es menor que una tolerancia establecida, aumentar el tamaño del paso de integración.
  - Si el error es mayor que la tolerancia, reducir el tamaño del paso de integración y repetir el paso actual.

- Actualizar los valores:
  t = t + h
  y = y_5

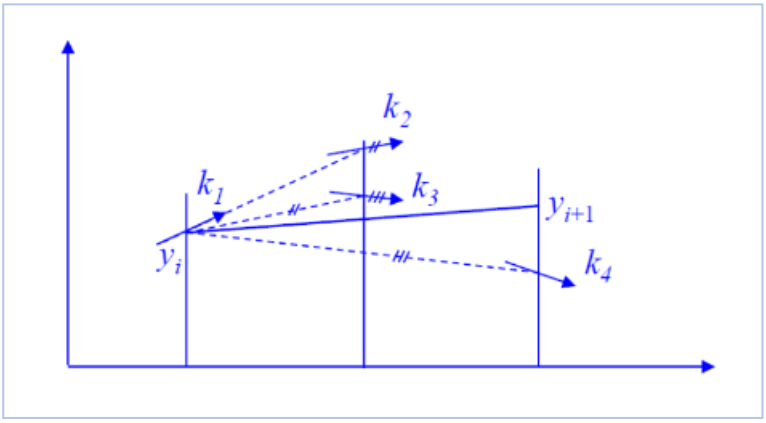


Figura 2 Método de Runge-Kutta

El método de Runge-Kutta de orden 4-5 proporciona una buena aproximación de la solución de la ODE, con un error local del orden de h^5 y un error global del orden de h^4, donde h es el tamaño del paso de integración. Al ajustar dinámicamente el tamaño del paso de integración en función del error estimado, el método es capaz de mantener una alta precisión numérica.

4.2 Sistemas Básicos

4.2.1 Circuito Eléctrico

Este sistema eléctrico RC está conformado por una resistencia (R) y un condensador (C). La corriente fluye a través de la resistencia y el condensador. La carga y descarga del condensador causa cambios en la corriente y la tensión a lo largo del tiempo.

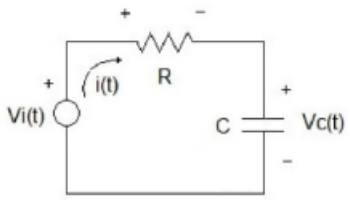


Figura 3 Circuito RC

Al ser un circuito eléctrico, usaremos las leyes de Kirchhoff para plantear el modelo matemático. Tendremos dos ecuaciones: la primera correspondiente a las caídas de tensión en la malla y la segunda a la caída de tensión en el condensador:

$$v_i(t) = Ri(t) + v_c(t)$$

$$C\frac{dv_c(t)}{dt} = i(t)$$

Si sustituimos la segunda ecuación en la primera obtendremos la ODE de primer orden que nos relaciona la entrada $v_i(t)$ con la salida $v_c(t)$:

$$v_i(t) = RC\frac{dv_c(t)}{dt} + v_c(t)$$

$$RC\frac{dv_c(t)}{dt} = v_i(t) - v_c(t)$$

$$\frac{dv_c(t)}{dt} = \frac{1}{RC}\left(v_i(t) - v_c(t)\right)$$

De esta forma tendríamos la ecuación diferencial que resolveremos gracias a MATLAB a través de la función: ode45.

4.2.2 Péndulo

Este es un sistema físico, es un objeto suspendido en un punto fijo, que puede oscilar libremente bajo la influencia de la gravedad en un plano.
El modelo matemático siguiente nos proporcionará su posición θ y velocidad angular w.

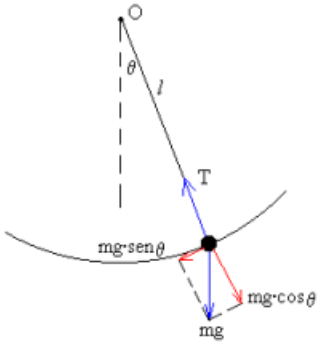


Figura 4 Péndulo

Siendo T la tensión del hilo y l la longitud.
En el caso de un péndulo simple, podemos descomponer el movimiento en dirección radial y tangencial utilizando coordenadas polares. Consideraremos el punto de suspensión como el origen del sistema de coordenadas.
La ecuación de movimiento en la dirección radial estaría compuesta por:

$$ma_n = \sum F_n$$

a_n= aceleración racial y F_n las fueras implicadas en la dirección normal

Desarrollando obtenemos la siguiente ecuación:

$$m\frac{v^2}{l} = T - mg \cdot \cos\theta$$

La ecuación de movimiento en la dirección tangencial sería:

$$ma_t = \sum F_t$$

Siendo a_t=aceleración tangencial y F_t las fuerzas en la dirección tangencial.
La a_t= α (aceleración angular) l (longitud), por lo que la ecuación quedaría

$$m\alpha l = -mg \cdot \sin\theta$$

así:

Dado que:

$$\alpha = \frac{d^2\theta}{dt^2}$$

El modelo matemático en forma de ecuación diferencial de segundo orden no lineal es:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{l}\sin\theta$$

Para poder usar este modelo en el sistema de simulación de MATLAB debemos poner este modelo en forma de ODE, mediante dos ecuaciones de primer grado. De forma que así quedaría el modelo matemático en forma de ODE con dos ecuaciones diferenciales de primer grado:

$$\frac{d\theta}{dt} = \omega$$
$$\frac{d\omega}{dt} = -\frac{g}{l}\sin\theta$$

4.2.3 Tanques Acoplados

El modelo matemático de tanques acoplados se refiere a un sistema de dos tanques interconectados donde el flujo de líquido entre ellos está determinado por las diferencias de nivel de líquido y las características del sistema. Este modelo se utiliza en aplicaciones de ingeniería para describir y controlar sistemas de almacenamiento y flujo de líquidos.

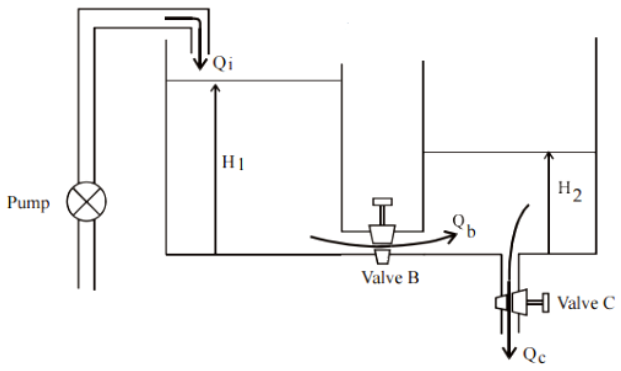


Figura 5 Tanques Acoplados

Cada uno de los tanques tiene una altura de agua que varía a lo largo del tiempo. Para definir el sistema, se requieren de ciertas ecuaciones que describen el flujo de líquidos entre los tanques.
El flujo de los líquidos se produce a través de una tubería de conexión con un área de sección constante.

La ecuación que describe el cambio en la altura del líquido en el Primer Tanque sería:

$$A_1 \frac{dh_1}{dt} = q_1 - F_1 = q_1 - K \cdot \sqrt{h_1 - h_2}$$

A_1: el área del tanque
h_1: la altura del líquido en el primer tanque
h_2: la altura del líquido en el segundo tanque
q_1: la tasa de flujo de líquido que entra en este primer tanque desde una fuente externa
F_1: es la tasa de flujo de líquido que pasa desde este tanque hasta el segundo.
K: sería una constante que representa la eficiencia del flujo entre los tanques y depende de varios factores como el área del orificio.

La ecuación que describe el cambio en la altura del líquido en el Segundo Tanque sería:

$$A_2 \frac{dh_2}{dt} = F_1 - F_2 = K \cdot \sqrt{h_1 - h_2} - K \cdot \sqrt{h_2}$$

A_2: el área del tanque
h_1: la altura del líquido en el primer tanque
h_2: la altura del líquido en el segundo tanque
q_1: la tasa de flujo de líquido que entra en este primer tanque desde una fuente externa
F_1: es la tasa de flujo de líquido que pasa desde el primer tanque hasta el este.
F_2: es la tasa de flujo de líquido que sale de este segundo tanque.
K: sería una constante que representa la eficiencia del flujo entre los tanques y depende de varios factores como el área del orificio.

4.3 Depuradora de Agua

4.3.1 Descripción y Funcionamiento general

Esta planta de depuradora de aguas residuales tiene como objetivo eliminar los contaminantes y reducir el impacto ambiental de estas aguas.
Para la eliminación de todos estos residuos la planta lleva a cabo una serie de etapas de tratamientos: pretratamiento inicial, tratamiento primario, tratamiento secundario y tratamiento terciario.

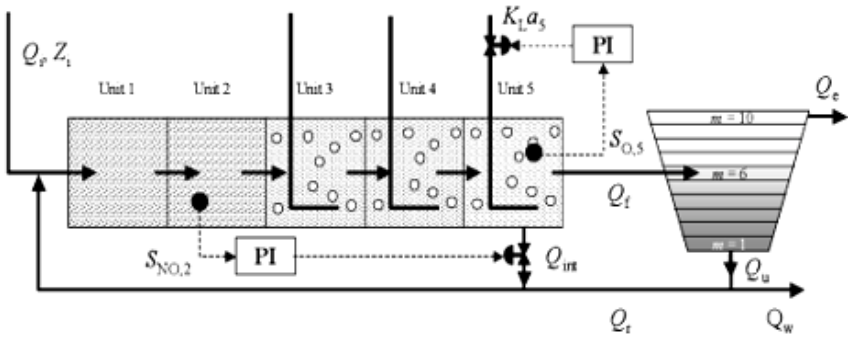


Figura 6 Depuradora de Agua

Vamos a ir explicando cada tratamiento para entender las fases por las que pasa el agua residual una vez entra en la depuradora:

- Pretratamiento: En esta etapa, las aguas residuales pasan por una serie de procesos para eliminar los sólidos más grandes y las grasas que vienen del colector.
  Para retener los sólidos más grandes como arena y piedras, se utilizan rejillas y tamices, retienen estos objetos y evitan daños en los equipos posteriores. Por último, se separan las grasas y aceites aprovechando la flotación de estas sustancias.
- Tratamiento Primario: En esta etapa, las aguas residuales pasan a través de un decantador primario, donde los sólidos suspendidos se asientan por gravedad y se forman lodos primarios. Estos lodos se retiran y se envían para su tratamiento adicional.
- Tratamiento Secundario: Esta es la etapa más importante del proceso de depuración, el objetivo es disminuir la cantidad de materia orgánica degradable biológicamente que contienen estas aguas. Básicamente lo que se hace es acelerar el proceso biológico que ocurriría naturalmente fuera de la depuradora.

Esta reacción tiene lugar en los **reactores biológicos**, son tanques aireados debido a que el proceso de degradación es normalmente aerobio. Después del paso suficiente del agua por estos reactores, quedaría una nueva sedimentación en los **decantadores secundarios** que separa el agua tratada y la biomasa presente. La biomasa se decanta al fondo y el agua tratada continúa con su depuración o se devuelve al río. Una parte de la biomasa decantada vuelve a los reactores biológicos con el objetivo de mantener una concentración suficiente de microorganismos, otro pequeño porcentaje se evacúa por el **canal de purga**.

- Tratamiento Terciario: compuesto por tratamientos más avanzados como puede ser la cloración, adsorción con carbón activo, o la ósmosis inversa, que disminuye más aún los niveles de los contaminantes en el agua.

4.3.2 Modelo Matemático Benchmark

Hay que mencionar también el modelo matemático que se ha utilizado para representar fielmente las características principales de la depuradora de agua, esta plataforma de simulación estándar se denominó BSM1 (Benchmark Simulation Model nº1), que representa el tratamiento secundario de una planta depuradora basada en fangos activados.

El Benchmark además de ser una planta de referencia también es los diferentes protocolos de simulación, datos de diversos influentes e índices de desempeño y comparación de diferentes estrategias de control.

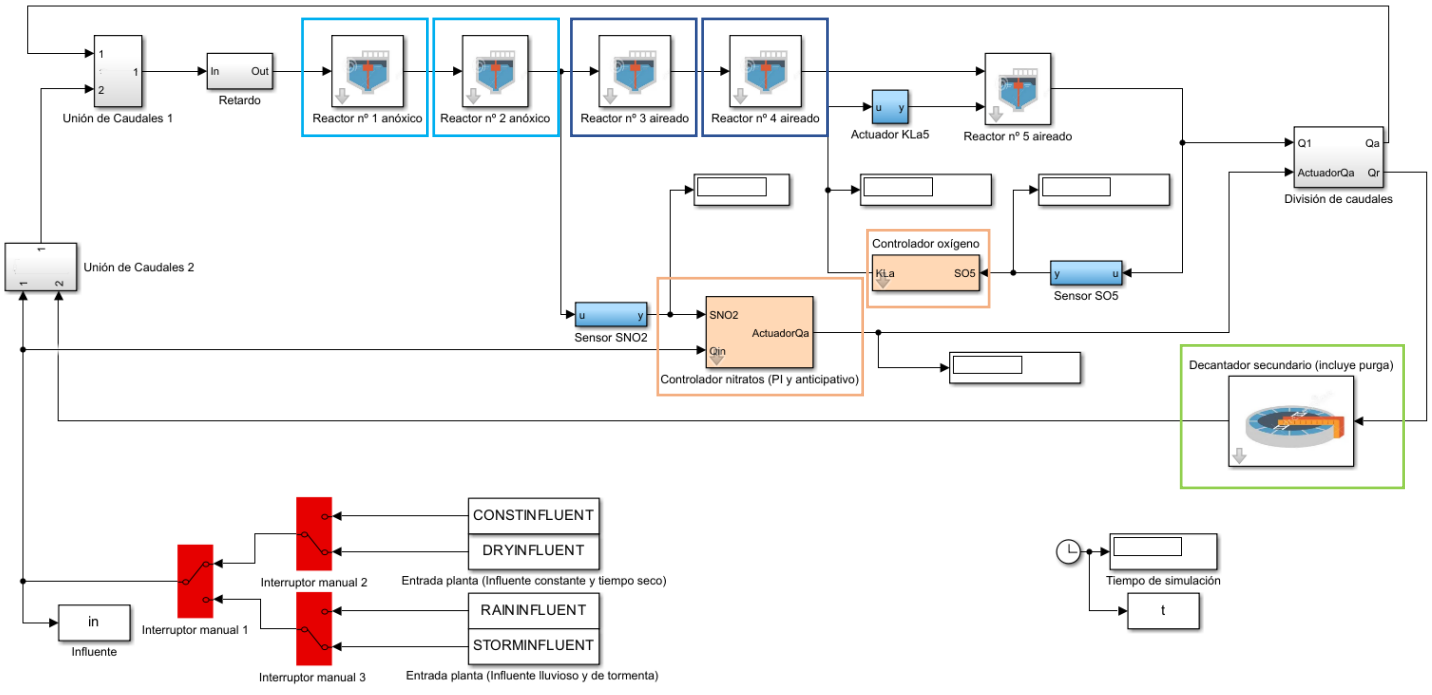


Figura 7 Modelo Matemático BSM1

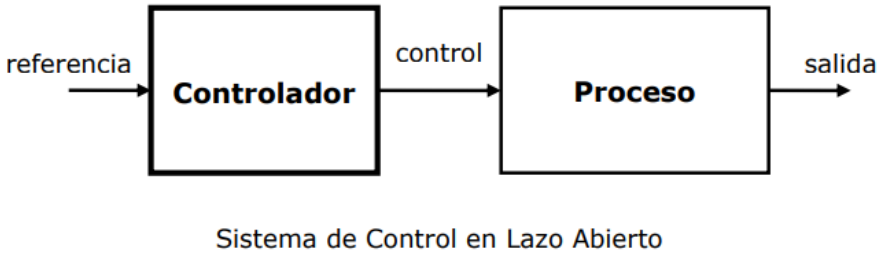
El BSM1 está compuesto por 5 **reactores biológicos**, de los cuales dos son anóxicos y tres **aeróbicos**.
En los dos primeros se lleva a cabo el proceso de desnitrificación, en cambio los tres siguientes se mantienen aireados y se lleva a cabo el proceso de nitrificación. Además de estos reactores biológicos también encontrados los tipos de controladores involucrados en el sistema: el Controlador de Oxígeno y el Controlador de Nitratos.
Estos controladores son PI, ambos tienen la constante proporcional Kp y el tiempo integral Ti. La ecuación de funcionamiento de un PI es:

$$u(t) = Kp *(e(t) + 1/Ti * \int e(t)\, dt)$$

Después de los reactores se encuentra un decantador secundario, encargado de separar los fangos del agua limpia

4.4. Control de Procesos

Los modelos que se plantean en el proyecto son modelos de plantas industriales, en el sector industrial existe una necesidad de controlar ciertos valores para que la planta en cuestión funcione de una determinada manera con una determinada exactitud.
En la antigüedad estas variables que había que controlar sobre un margen se ponían en manos de un trabajador, un control manual, que se encargaba de controlar que, por ejemplo, cierto valor se mantuviera siempre constante y modificar las variables necesarias a lo largo del tiempo para que esto ocurriera.

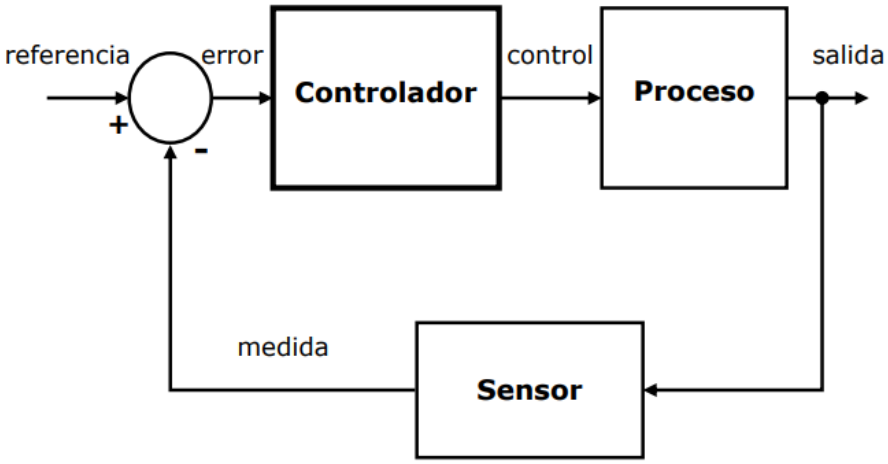


Sistema de Control en Lazo Abierto

Figura 8 Lazo Abierto

Evidentemente este control manual, también llamado de lazo abierto, pasó a segundo plano cuando empezaron a aparecer los sistemas automáticos de control que no requerían de una mano humana para controlar la planta.
Apareció la regulación automática, que permitía controlar las variables física de una planta a lo largo del tiempo incluso con la presencia de perturbaciones.
Las perturbaciones es el mayor de los problemas en los sistemas de control, son alteraciones físicas en la planta impredecibles y no deseadas que modifican los valores marcados por la empresa (set point).
El set point o valor de referencia, es el valor que la empresa industrial decide marcar como objetivo que debe llevar la planta y que debe mantener a lo largo del tiempo, dependiendo del modelo en cuestión puede ser: la altura del agua en un tanque, la temperatura de la leche en un proceso de pasteurización, el % de oxígeno del agua en un determinado momento, etc.

Obviamente dependiendo del tipo de planta que estemos llevando nos interesará más la precisión, error mínimo con estos valores, o la rapidez con la que actúa a ciertas perturbaciones, es decir, la velocidad con la que vuelve a recuperar el valor de referencia después de ciertas perturbaciones.



Sistema de Control en Lazo Cerrado

Figura 9 Lazo Cerrado

El control por lazo cerrado permite controlar las variables de control necesarias para que la planta se mantenga en el valor de referencia marcado a lo largo del tiempo. Básicamente se encarga de medir la salida y compararla con el valor de referencia, esa diferencia o error será lo que indicará al actuador como debe actuar sobre la planta.

Los diferentes componentes de un sistema de control en lazo cerrado son:

- Sensor: el sensor de una planta es un aparato cuya finalidad es captar la variable física que queremos regular y convertirla en una señal eléctrica.
- Transmisor: se conectar directamente con el sensor de la planta y tiene como objetivo adaptar y convertir la señal recibida por el actuador y enviarla al sistema de medida y control.
- Comparador: aquí es donde se genera la señal de error, es donde se compara la señal llegada del sensor, la variable que se está controlando, con la señal de referencia. Es decir, se compara el valor real de la planta con el valor deseado, la diferencia de ambos genera una señal de error que se transmitirá al regulador.
- Regulador: el regulador es uno de los bloques más claves en los sistemas control, pues es el encargado de a partir de una señal de error devolver una señal de control. La señal de control dependerá

tanto de la señal de error como del algoritmo utilizado en el regulador para devolver la señal, estos algoritmos son muy famosos dentro de los sistemas industriales: PID, PI, PD.

- Actuador: el actuador es el encargado de transformar la señal de control recibida en una acción física en la planta.

Todos estos componentes aparecen únicamente en el modelo de la Depuradora de Agua, al ser un sistema complejo implementado en lazo cerrado.

5. Técnicas y herramientas

En este apartado desarrollaré las técnicas y herramientas utilizadas para el desarrollo de la aplicación, mostrando los aspectos más relevantes en cada caso.

5.4. MVC

El patrón de diseño utilizado es el conocido Modelo, Vista y Controlador, usado para el diseño arquitectónico del software.

- Modelo: es el encargado de guardar los datos que se utilizan en la aplicación, gestiona los mismos.
- Vista: es la que el usuario ve, encargado de mostrar al usuario todas las ventanas, es la interacción directa con el usuario.
- Controlador: es el intermediario entre el modelo y la vista, se encarga de comunicar el modelo y la vista, de forma que estos dos no se comuniquen directamente.

5.5. MATLAB

Todo el proyecto gira en torno a la aplicación MATLAB, la cual nos sirve como base para crear la interfaz a través de una app que proporciona llamada App Designer (anteriormente llamada GUIDE).
Para hacer las simulaciones se utiliza el propio MATLAB y el lenguaje de modelado y simulación basado en bloques de Simulink (que se encuentra integrado en MATLAB).

Antes de comentar los aspectos más relevantes de estas dos aplicaciones, comentaremos los papeles fundamentales que tiene MATLAB como aplicación en el desarrollo del trabajo.

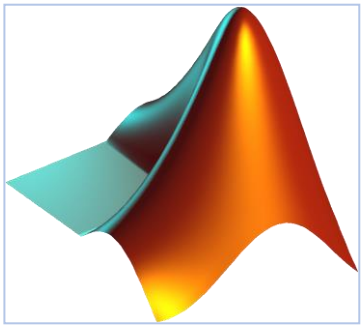


Figura 10 MATLAB

5.5.1 Principales Características de MATLAB

MATLAB es el entorno de trabajo en el que se asienta el desarrollo del proyecto.
Sirve de puente y base para conectar Simulink y App Designer que juntas han servido para el desarrollo de la aplicación. Hay 3 aspectos dentro de MATLAB que nos permiten trabajar con este entorno de trabajo: El Workspace (espacio de trabajo), la Ventana de comandos y por último la carpeta en la que se está trabajando, es decir, donde se encuentran todos los archivos que utilizamos.

- El workspace permite al usuario definir variables y éstas se declararán en este espacio de trabajo. Esto es algo que utilizaremos constantemente en nuestra aplicación pues todas y cada una de las variables que usemos se guardarán en el espacio de trabajo.
- La ventana de comandos es clave para ejecutar cualquier fichero, crear, modificar, borrar cualquier variable, devolver la salida de la ejecución de una función, etc.
- Current Folder, nos permite trabajar en una misma carpeta con todos los archivos que necesitemos en cada momento.

Aparte de estas ventajas, también hay que añadir la capacidad de MATLAB para realizar cualquier tipo de cálculos pues es una aplicación que está diseñada para eso y en nuestro caso nos es muy útil para las simulaciones en lazo abierto y en lazo cerrado.

5.5.2 Principales Características de Simulink

Como se ha señalado antes, Simulink es una aplicación dentro de MATLAB cuya funcionalidad es permitir la creación y simulación de modelos basados en diagramas de bloques, que son componentes que se interconectan entre sí.
La herramienta permite la creación de subsistemas que agrupan diferentes componentes del esquema de proceso. A partir de esos subsistemas se pueden crear Máscaras (interfaces), cuya finalidad es la interacción directa con el usuario para la modificación de variables del propio subsistema.

5.5.3 Principales Características de App Designer

El App Designer es una herramienta de desarrollo gráfico que se encuentra en MATLAB. Es el pilar fundamental del desarrollo de la interfaz gráfica de la aplicación

A continuación, se presentan algunas de las características generales del App Designer de MATLAB:

- Interfaz gráfica de usuario (GUI): App Designer proporciona una interfaz gráfica intuitiva para diseñar la apariencia y el comportamiento de la aplicación. Los elementos de la GUI, como botones, cuadros de texto, gráficos y tablas, se pueden agregar y personalizar fácilmente con simples operaciones de arrastrar y soltar.
- Entorno de desarrollo integrado: App Designer está completamente integrado en MATLAB, lo que permite acceder a todas las funciones y capacidades de MATLAB desde la aplicación creada. Puedes utilizar todas las funciones y operaciones matemáticas disponibles en MATLAB dentro de tu aplicación.
- Editor de código: App Designer también incluye un editor de código, que permite agregar funcionalidades personalizadas a la aplicación mediante la escritura de código en lenguaje MATLAB. Esta función es especialmente útil para realizar cálculos complejos o para agregar lógica y control de eventos.
- Ejecución en tiempo real: Puedes ejecutar y probar la aplicación directamente en el entorno de App Designer. Esto te permite ver los resultados y el comportamiento de la aplicación mientras la construyes, lo que facilita la depuración y el ajuste.
- Soporte para callbacks: Puedes asignar funciones de MATLAB como callbacks a diferentes elementos de la GUI, lo que permite responder a eventos, como clics de botón o cambios en los valores de los cuadros de texto.

5.6. Herramientas Complementarias

A continuación, se comentarán lo que son las herramientas más secundarias del proyecto.

5.6.1 Microsoft Word

Utilizado principalmente para la documentación de toda la memoria del Trabajo Fin de Grado.

Figura 11 Microsoft Word

5.6.2 Microsoft Proyect

Es una herramienta de gestión de proyectos diseñada para planificar, programar, asignar recursos, realizar seguimiento y controlar proyectos de manera eficiente. Esta herramienta se ha utilizado para ir asignando las tareas y los recursos del proyecto y así poder estimar el tiempo que se dedica a cada tarea y, por ende, estimar el tiempo que conllevaría el desarrollo completo del proyecto.

Figura 12 Microsoft Proyect

5.6.3 Visual Paradigm

Esta aplicación se ha utilizado principalmente para la creación y el diseño de todos los diagramas que aparecen en la documentación del proyecto.
Se escogió esta herramienta porque ya la había utilizado en asignaturas de ingeniería del software, pues ya estaba familiarizado con la interfaz.

Figura 13 Visual Paradigm

6. Aspectos relevantes del desarrollo del proyecto

En este apartado desarrollaremos los aspectos más interesantes del desarrollo del sistema software. En una primera instancia explicaremos el marco de trabajo utilizado, el ciclo de vida, se explicará la arquitectura utilizada, especificación de requisitos y las diferentes etapas.

6.1. Ciclo de Vida

En este proyecto se ha usado un marco de trabajo muy conocido para el desarrollo de sistemas software: el Proceso Unificado.

El proceso unificado se basa en los principios de la ingeniería de software orientada a objetos y se enfoca en la iteración y la colaboración entre los miembros del equipo de desarrollo.

El ciclo de vida está divido en hitos que contienen iteraciones y que a su vez estás están formadas por las tareas que se realizan.

Los hitos tienen un principio y un fin, y suelen ser en estos puntos donde se dan las reuniones con los tutores.

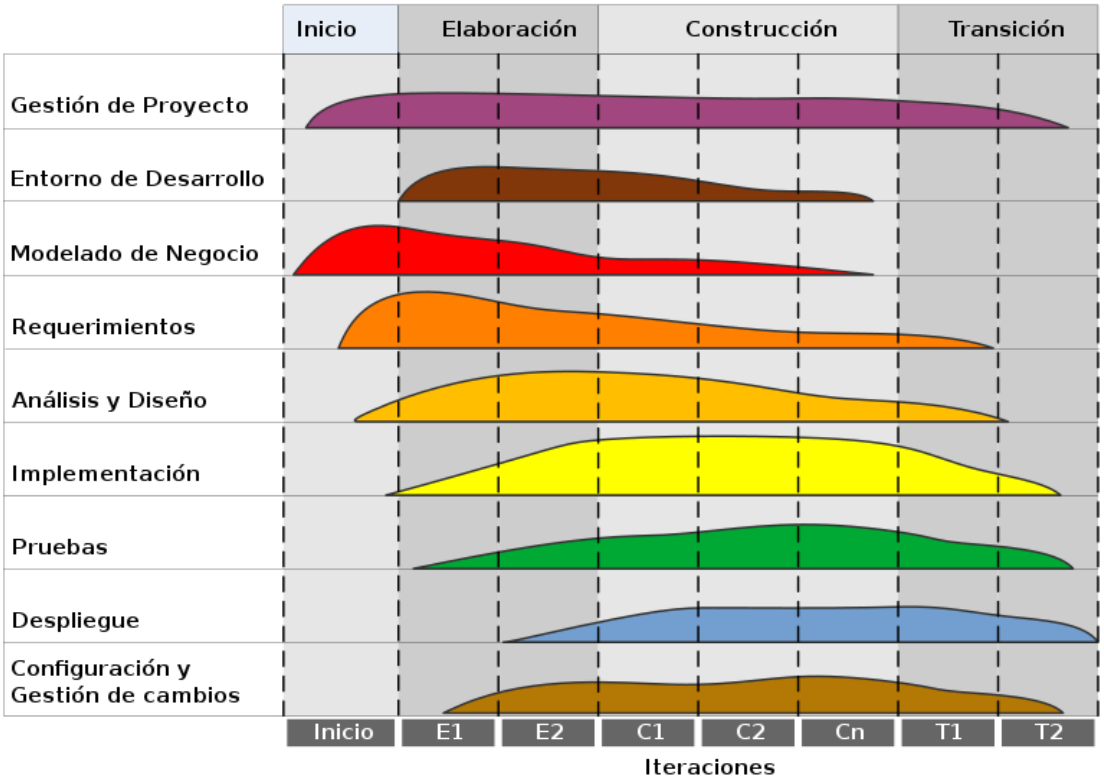


Figura 14 Proceso Unificado

Es un proceso cíclico que se va repitiendo y refinando, cada ciclo está compuesto por cuatro fases:

- Inicio: es la primera fase del proceso unificado y normalmente la más corta. Se mantienen las primeras reuniones con los tutores del TFG y se establecen los requisitos para la aplicación. Se busca el enfoque que se le quiere dar al proyecto y se deciden las herramientas que se van a utilizar, en este caso se decidió giran todo entorno a MATLAB. Comenzamos con los casos de uso y empezamos a familiarizarnos con el lenguaje de programación que utilizaremos.
- Elaboración: en esta fase de elaboración siguen las reuniones con los tutores seguir refinando las funcionalidades de la aplicación, añadiendo nuevas y descartando otras. Se definen los objetivos del proyecto basándonos en los requisitos anteriores y refinamos los casos de uso.
- Construcción: esta fase de construcción es la más larga de todo el proceso unificado, es básicamente, en la que se desarrolla todo el software en cuestión, se construye el sistema que vamos a presentar.

  En esta fase tendremos reuniones con los tutores una vez a la semana normalmente, donde seguiremos refinando el diseño del proyecto, pero las reuniones serán más centradas en los posibles fallos, cambios o mejoras técnicas del desarrollo del software.

  Seguimos refinando los casos de uso y aquí ya se empiezan a hacer pruebas individuales del sistema para ver cómo responde ante el uso de un usuario externo.
- Transición: por último, tendríamos la fase final que junto con la de inicio es de las más cortas. En esta fase realizamos la mayor cantidad de pruebas al sistema para asegurarnos que responde perfectamente y no existen fallos.

  Tenemos las últimas tutorías, ya menos frecuentes, y mostrando a los tutores el sistema completo, mejorando ciertos detalles menos importantes en estas reuniones.

6.2. Planificación Temporal

En esta parte mostraremos el cronograma creado con Microsoft Proyect, donde establecemos el orden y la secuencia de las diferentes tareas realizadas, así como el inicio y fin de ellas.
En una primera instancia crearemos nuestro calendario con los días laborales, horas de trabajo por día, por semana y días de trabajo al mes. Especificar los días no laborales, en mi caso sería la semana santa.

Después se debe definir el alcance del proyecto, es decir, estimar la duración total y desde así tener una base para planificar los hitos, iteraciones y tareas.
Como hemos comentado antes, cada fase del proyecto se centra en unas determinadas disciplinas: Modelado de negocio, Requisitos, Análisis, Diseño, Implementación y Pruebas.
Dependiendo de la fase en la que estemos le dedicaremos más tiempo a una disciplina u otra, adaptándonos a lo que el proyecto necesite en ese momento.

A continuación, se muestra una tabla con todas las tareas realizadas que se encuentran en la planificación temporal:

Número	Tarea	Descripción
1	Primera reunión con tutores	La primera reunión entender el enfoque inicial del proyecto
2	Identificar Actores	Se identifican los actores a los que irá dirigida la aplicación
3	Identificar Casos de Uso	Se identifican los casos de uso
4	Identificar requisitos de información	Se identifican los requisitos de información
5	Identificar objetivos	Se identifican los objetivos principales de la aplicación

6	Identificar Requisitos no funcionales	Se definen cuales son los requisitos no funcionales aplicables al sistema
7	Elegir las herramientas para el desarrollo	Se elige MATLAB como aplicación sobre la que se desarrollará el proyecto
8	Analizar los Objetivos	Se analizan los objetivos identificados
9	Prototipo de Diseño de Arquitectura	Se diseña un primer prototipo de cómo será el diseño de la aplicación
10	Reunión con tutores	Sucesivas reuniones dónde se muestran los avances a los tutores recibiendo un feedback
11	Mejorar del prototipo de diseño	Se mejora el prototipo del diseño después del feedback de la reunión
12	Primeros casos de uso	Se empiezan a hacer los casos de uso
13	Requisitos no funcionales	Se empiezan a hacer los requisitos no funcionales
14	Diagramas de actores	Se hacen los diagramas de actores
15	Diagrama de clases	Se hacen los diagramas de clases
16	Diagramas de diseño	Creamos los primeros diagramas de diseño
17	Primeros Diagramas de secuencia	Creamos los primeros diagramas de secuencia

18	Comienzo del proyecto	Se empieza a desarrollar el proyecto en la App Designer de MATLAB
19	Refinamiento de los casos de uso	Se refinan los casos de uso creados anteriormente
20	Refinar Requisitos de información	Se refinan los requisitos de información después de reunión con los tutores
21	Refinamiento de requisitos	Se refinan los requisitos definidos
22	Refinamiento del diseño	Refinamos el diseño
23	Añadimos nuevos casos de uso	Después de reunión añadimos nuevos casos de uso por nuevas funcionalidades
24	Propuesta inicial de arquitectura	Presentamos la primera propuesta de arquitectura MVC
25	Primeros subsistemas de diseño	Creamos los primeros subsistemas del diseño
26	Implementación del Objetivo-01 “Gestionar Circuito”	Implementamos el primer objetivo definido en la especificación de requisitos
27	Implementación del Objetivo-02 “Gestionar Péndulo”	Implementamos el segundo objetivo, el segundo modelo básico
28	Implementación del Objetivo-03 “Gestionar Tanques Acoplados”	Se implementa el siguiente modelo básico, los tanques acoplados, este nos llevará más tiempo en

		terminar
29	Implementación del Objetivo-04 “Gestionar Depuradora de Agua”	Se implementa el modelo más complejo del sistema, al que más tiempo dedicaremos.
30	Pruebas del Objetivo-01	Hacemos pruebas al Objetivo-01
31	Pruebas del Objetivo-02	Hacemos pruebas al Objetivo-02
32	Pruebas del Objetivo-03	Hacemos pruebas de usuario al Objetivo-03
33	Pruebas del Objetivo-04	Hacemos pruebas de usuario al Objetivo-04
34	Pruebas del Sistemas Completo	Hacemos pruebas del sistema completo
35	Últimas Pruebas de la Aplicación	Hacemos las últimas pruebas del sistema para asegurarnos que todo está en orden y no hay errores.
36	Refinamiento del Objetivo-01 “Gestionar Circuito”	Después del feedback recibido de los tutores, refinamos este primer modelo
37	Refinamiento del Objetivo-02 “Gestionar Péndulo”	Igual que la tarea anterior, pero con el segundo modelo
38	Refinamiento del Objetivo-03 "Gestionar Tanques Acoplados"	Después de reunión, refinamos el Objetivo-03 mejorando la interfaz y funcionalidad
39	Refinamiento del Objetivo-04 "Gestionar Depuradora de Agua"	Sucesivas reuniones para refinar el modelo más complejo
40	Finalización de Requisitos	Dejamos ya terminados todos los requisitos
41	Analizar los Resultados	Analizamos los resultados que hemos obtenido

42	Últimas Reuniones con los Tutores	Concretamos las últimas reuniones con los tutores para refinar los últimos detalles
43	Aplicar cambios en los objetivos	Aplicamos los cambios en los objetivos
44	Inicio de Documentación	Comenzamos con la documentación
45	Refinamiento de la Documentación	Después del feedback recibido por los tutores refinamos la documentación
46	Finalización de la Documentación	Terminamos definitivamente la documentación del trabajo fin de grado

Como podemos observar se trata de 46 tareas realizadas, con una duración total del proyecto de 115 días, desde el jueves 16/02/2023 hasta el miércoles 02/08/2023.

Aquí se muestra el diagrama de Gantt que se ha creado en Microsoft Proyect planificando todos los hitos, iteraciones y tareas a lo largo del tiempo:

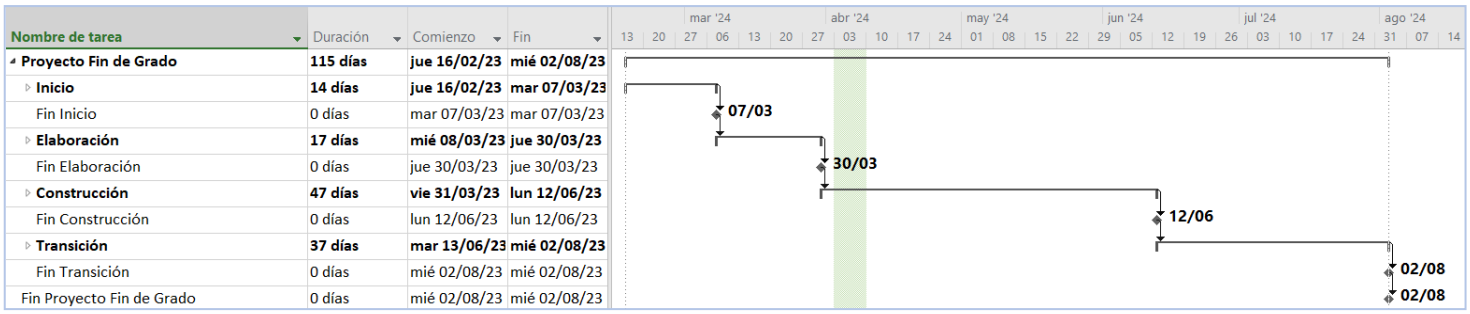


Figura 15 Diagrama de Gantt (Parte 1)

Nombre de tarea	Duración	Comienzo	Fin
Proyecto Fin de Grado	**115 días**	**jue 16/02/23**	**mié 02/08/23**
Inicio	**14 días**	**jue 16/02/23**	**mar 07/03/23**
Iteración 1	**3 días**	**jue 16/02/23**	**lun 20/02/23**
Modelado del Negocio	**1 día**	**jue 16/02/23**	**jue 16/02/23**
Primera Reunión con Tutores	1 día	jue 16/02/23	jue 16/02/23
Requisitos	**2 días**	**vie 17/02/23**	**lun 20/02/23**
Identificar actores	1 día	vie 17/02/23	vie 17/02/23
Identificar casos de uso	1 día	lun 20/02/23	lun 20/02/23
Fin Iteración	0 días	lun 20/02/23	lun 20/02/23
Iteración 2	**6 días**	**mar 21/02/23**	**mar 28/02/23**
Modelado del Negocio	**1 día**	**mar 21/02/23**	**mar 21/02/23**
Elegir las herramientas para el desarrollo	1 día	mar 21/02/23	mar 21/02/23
Requisistos	**2 días**	**mié 22/02/23**	**jue 23/02/23**
Identificar requisitos de información	1 día	mié 22/02/23	mié 22/02/23
Identificar objetivos	1 día	jue 23/02/23	jue 23/02/23
Análisis	**1 día**	**vie 24/02/23**	**vie 24/02/23**
Analizar los objetivos	1 día	vie 24/02/23	vie 24/02/23
Diseño	**2 días**	**lun 27/02/23**	**mar 28/02/23**
Prototipo del diseño de arquitectura	2 días	lun 27/02/23	mar 28/02/23
Fin Iteración 2	0 días	mar 28/02/23	mar 28/02/23

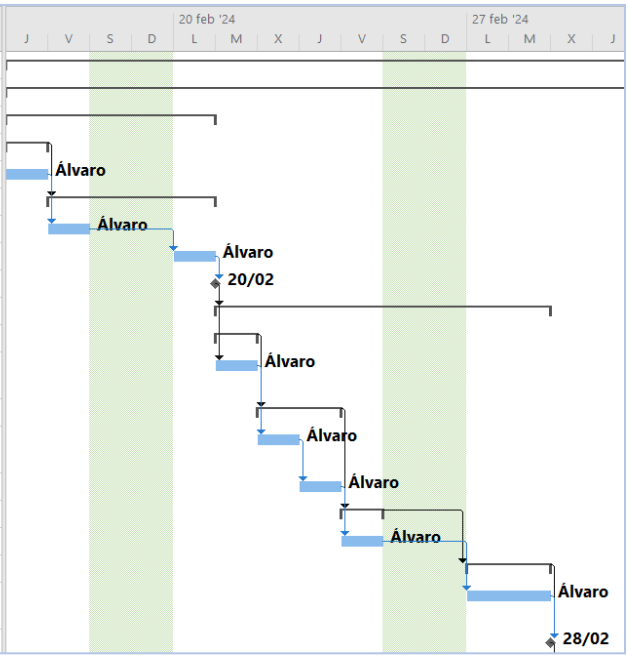


Figura 16 Diagrama de Gantt (Parte 2)

Nombre de tarea	Duración	Comienzo	Fin
Iteración 3	**5 días**	**mié 01/03/23**	**mar 07/03/23**
Modelado del Negocio	**1 día**	**mié 01/03/23**	**mié 01/03/23**
Reunión con tutores	1 día	mié 01/03/23	mié 01/03/23
Requisitos	**2 días**	**jue 02/03/23**	**vie 03/03/23**
Refinar requisitos de información	1 día	jue 02/03/23	jue 02/03/23
Identificar requisitos no funcionales	1 día	vie 03/03/23	vie 03/03/23
Análisis	**1 día**	**lun 06/03/23**	**lun 06/03/23**
Analizar los objetivos	1 día	lun 06/03/23	lun 06/03/23
Diseño	**1 día**	**mar 07/03/23**	**mar 07/03/23**
Mejora del prototipo de diseño	1 día	mar 07/03/23	mar 07/03/23
Fin Iteración 3	0 días	mar 07/03/23	mar 07/03/23
Fin Inicio	0 días	mar 07/03/23	mar 07/03/23

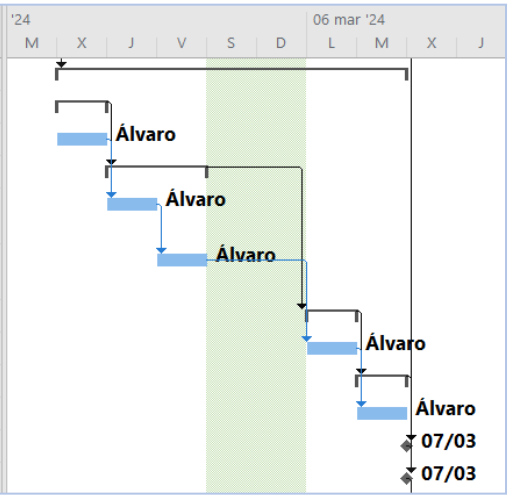


Figura 17 Diagrama de Gantt (Parte 3)

Nombre de tarea	Duración	Comienzo	Fin
Elaboración	**17 días**	**mié 08/03/23**	**jue 30/03/23**
Iteración 1	**10 días**	**mié 08/03/23**	**mar 21/03/23**
Modelado del Negocio	**1 día**	**mié 08/03/23**	**mié 08/03/23**
Reunión con los tutores	1 día	mié 08/03/23	mié 08/03/23
Requisitos	**3 días**	**jue 09/03/23**	**lun 13/03/23**
Primeros casos de uso	2 días	jue 09/03/23	vie 10/03/23
Requisitos no funcionales	1 día	lun 13/03/23	lun 13/03/23
Análisis	**3 días**	**mar 14/03/23**	**jue 16/03/23**
Diagramas de actores	1 día	mar 14/03/23	mar 14/03/23
Diagramas de clases	2 días	mié 15/03/23	jue 16/03/23
Diseño	**2 días**	**vie 17/03/23**	**lun 20/03/23**
Diagramas de diseño	2 días	vie 17/03/23	lun 20/03/23
Implementación	**1 día**	**mar 21/03/23**	**mar 21/03/23**
Comienzo del proyecto	1 día	mar 21/03/23	mar 21/03/23
Fin Iteración 1	0 días	mar 21/03/23	mar 21/03/23

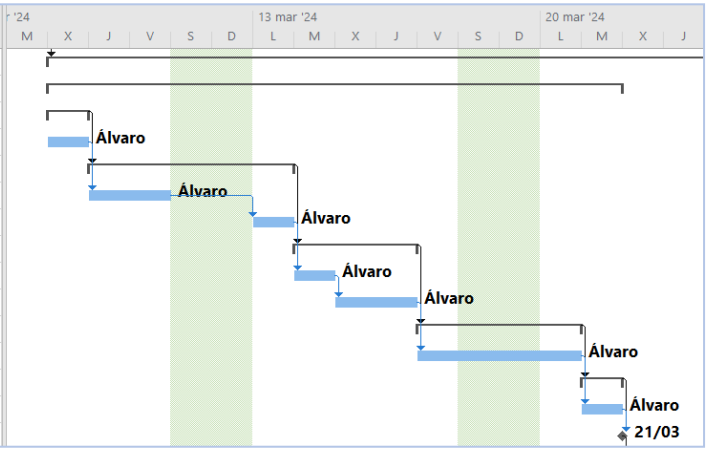


Figura 18 Diagrama de Gantt (Parte 4)

Nombre de tarea	Duración	Comienzo	Fin
Iteración 2	**7 días**	**mié 22/03/23**	**jue 30/03/23**
Modelación del Negocio	**1 día**	**mié 22/03/23**	**mié 22/03/23**
Reunión con los tutores	1 día	mié 22/03/23	mié 22/03/23
Requisitos	**4 días**	**jue 23/03/23**	**mar 28/03/23**
Refinamiento de casos de uso	1 día	jue 23/03/23	jue 23/03/23
Añadimos nuevos casos de uso	2 días	vie 24/03/23	lun 27/03/23
Primeros diagramas de secuencia	1 día	mar 28/03/23	mar 28/03/23
Análisis	**1 día**	**mié 29/03/23**	**mié 29/03/23**
Propuesta inicial de Arquitectura	1 día	mié 29/03/23	mié 29/03/23
Diseño	**1 día**	**jue 30/03/23**	**jue 30/03/23**
Primeros subsistemas de diseño	1 día	jue 30/03/23	jue 30/03/23
Fin Iteración 2	0 días	jue 30/03/23	jue 30/03/23
Fin Elaboración	0 días	jue 30/03/23	jue 30/03/23

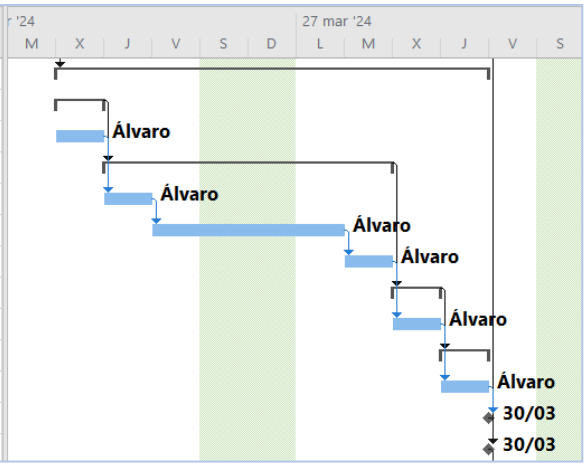


Figura 19 Diagrama de Gantt (Parte 5)

Nombre de tarea	Duración	Comienzo	Fin
Construcción	**47 días**	**vie 31/03/23**	**lun 12/06/23**
Iteracción 1	**9 días**	**vie 31/03/23**	**mié 19/04/23**
Requisitos	**1 día**	**vie 31/03/23**	**vie 31/03/23**
Refinamiento de Requisitos	1 día	vie 31/03/23	vie 31/03/23
Análisis	**1 día**	**lun 10/04/23**	**lun 10/04/23**
Refinamiento de Casos de Uso	1 día	lun 10/04/23	lun 10/04/23
Diseño	**2 días**	**mar 11/04/23**	**mié 12/04/23**
Refinamiento de Diseño	2 días	mar 11/04/23	mié 12/04/23
Implementación	**3 días**	**jue 13/04/23**	**lun 17/04/23**
Implementación del Objetivo-01 "Gestionar Circuito"	2 días	jue 13/04/23	vie 14/04/23
Implementación del Objetivo-02 "Gestionar Péndulo"	1 día	lun 17/04/23	lun 17/04/23
Pruebas	**2 días**	**mar 18/04/23**	**mié 19/04/23**
Pruebas del Objetivo-01	1 día	mar 18/04/23	mar 18/04/23
Pruebas del Objetivo-02	1 día	mié 19/04/23	mié 19/04/23
Fin Iteracción 1	0 días	mié 19/04/23	mié 19/04/23

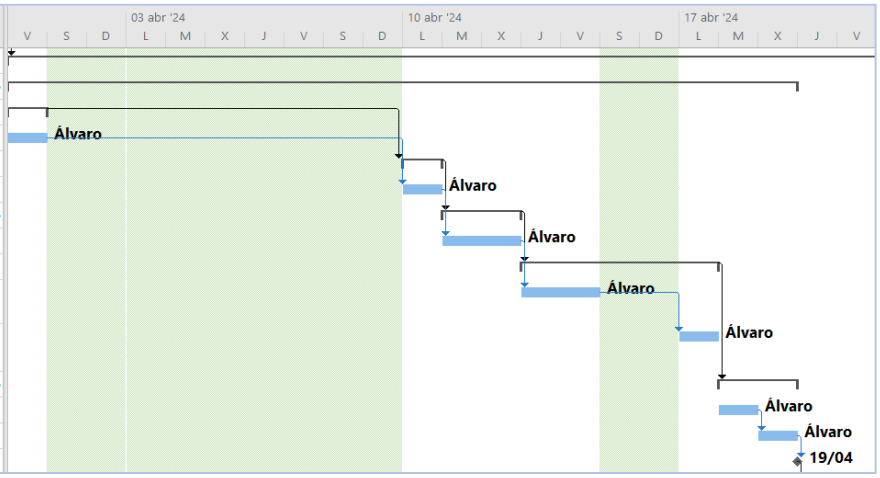


Figura 20 Diagrama de Gantt (Parte 6)

Nombre de tarea	Duración	Comienzo	Fin
Iteracción 2	**16 días**	**jue 20/04/23**	**jue 11/05/23**
Modelado del Negocio	**1 día**	**jue 20/04/23**	**jue 20/04/23**
Reunión con los tutores	1 día	jue 20/04/23	jue 20/04/23
Requisitos	**1 día**	**vie 21/04/23**	**vie 21/04/23**
Refinamiento de los Requisitos	1 día	vie 21/04/23	vie 21/04/23
Análisis	**2 días**	**lun 24/04/23**	**mar 25/04/23**
Refinamiento del Objetivo-01 “Gestionar Circuito”	1 día	lun 24/04/23	lun 24/04/23
Refinamiento del Objetivo-02 “Gestionar Péndulo”	1 día	mar 25/04/23	mar 25/04/23
Diseño	**1 día**	**mié 26/04/23**	**mié 26/04/23**
Refinamiento del Diseño	1 día	mié 26/04/23	mié 26/04/23
Implementación	**5 días**	**jue 27/04/23**	**mié 03/05/23**
Implementación del Objetivo-03 “Gestionar Tanques Acoplados”	2 días	jue 27/04/23	vie 28/04/23
Implementación del Objetivo-04 “Gestionar Depuradora de Agua”	3 días	lun 01/05/23	mié 03/05/23
Pruebas	**6 días**	**jue 04/05/23**	**jue 11/05/23**
Pruebas del Objetivo-01	1 día	jue 04/05/23	jue 04/05/23
Pruebas del Objetivo-02	1 día	vie 05/05/23	vie 05/05/23
Pruebas del Objetivo-03	2 días	lun 08/05/23	mar 09/05/23
Pruebas del Objetivo-04	2 días	mié 10/05/23	jue 11/05/23
Fin Iteracción 2	0 días	jue 11/05/23	jue 11/05/23

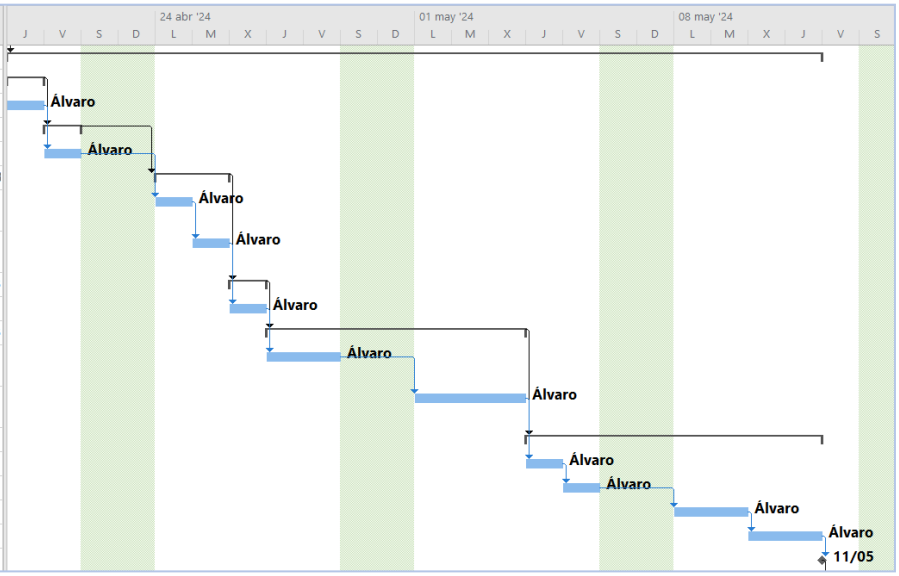


Figura 21 Diagrama de Gantt (Parte 7)

Nombre de tarea	Duración	Comienzo	Fin
Iteracción 3	**12 días**	**vie 12/05/23**	**lun 29/05/23**
Modelado del Negocio	**1 día**	**vie 12/05/23**	**vie 12/05/23**
Reunión con los tutores	1 día	vie 12/05/23	vie 12/05/23
Requisitos	**1 día**	**lun 15/05/23**	**lun 15/05/23**
Refinamiento de los Requisitos	1 día	lun 15/05/23	lun 15/05/23
Análisis	**2 días**	**mar 16/05/23**	**mié 17/05/23**
Refinamiento del Objetivo-03 "Gestionar Tanques Acoplados"	1 día	mar 16/05/23	mar 16/05/23
Refinamiento del Objetivo-04 "Gestionar Depuradora de Agua"	1 día	mié 17/05/23	mié 17/05/23
Diseño	**1 día**	**jue 18/05/23**	**jue 18/05/23**
Refinamiento del Diseño	1 día	jue 18/05/23	jue 18/05/23
Implementación	**3 días**	**vie 19/05/23**	**mar 23/05/23**
Implementación del Objetivo-03 “Gestionar Tanques Acoplados”	1 día	vie 19/05/23	vie 19/05/23
Implementación del Objetivo-04 “Gestionar Depuradora de Agua”	2 días	lun 22/05/23	mar 23/05/23
Pruebas	**4 días**	**mié 24/05/23**	**lun 29/05/23**
Pruebas del Objetivo-03	1 día	mié 24/05/23	mié 24/05/23
Pruebas del Objetivo-04	2 días	jue 25/05/23	vie 26/05/23
Pruebas del Sistema completo	1 día	lun 29/05/23	lun 29/05/23
Fin Iteracción 3	0 días	lun 29/05/23	lun 29/05/23

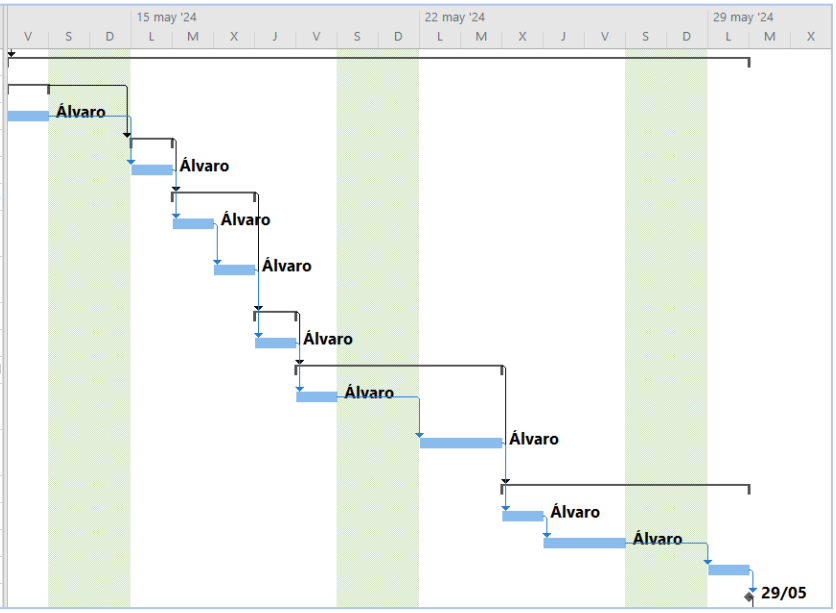


Figura 22 Diagrama de Gantt (Parte 8)

Nombre de tarea	Duración	Comienzo	Fin
Iteracción 4	**10 días**	**mar 30/05/23**	**lun 12/06/23**
Modelado de Negocio	**1 día**	**mar 30/05/23**	**mar 30/05/23**
Reunión con los Tutores	1 día	mar 30/05/23	mar 30/05/23
Requisitos	**1 día**	**mié 31/05/23**	**mié 31/05/23**
Refinamiento de los Requisitos	1 día	mié 31/05/23	mié 31/05/23
Análisis	**2 días**	**jue 01/06/23**	**vie 02/06/23**
Refinamiento del Objetivo-04 "Gestionar Depuradora de Agua"	2 días	jue 01/06/23	vie 02/06/23
Diseño	**1 día**	**lun 05/06/23**	**lun 05/06/23**
Refinamiento del Diseño	1 día	lun 05/06/23	lun 05/06/23
Implementación	**2 días**	**mar 06/06/23**	**mié 07/06/23**
Implementación del Objetivo-04 “Gestionar Depuradora de Agua”	2 días	mar 06/06/23	mié 07/06/23
Pruebas	**3 días**	**jue 08/06/23**	**lun 12/06/23**
Pruebas del Objetivo-04	1 día	jue 08/06/23	jue 08/06/23
Pruebas del Sistema completo	2 días	vie 09/06/23	lun 12/06/23
Fin Iteracción 4	0 días	lun 12/06/23	lun 12/06/23
Fin Construcción	0 días	lun 12/06/23	lun 12/06/23

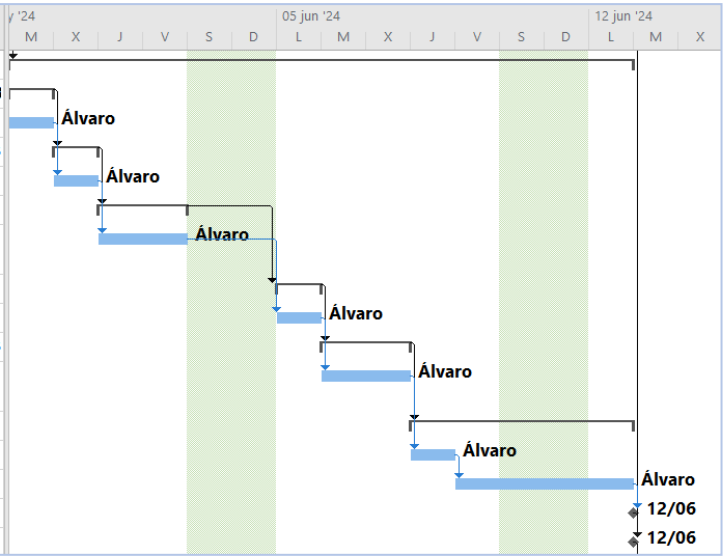


Figura 23 Diagrama de Gantt (Parte 9)

Nombre de tarea	Duración	Comienzo	Fin
Transición	**37 días**	**mar 13/06/23**	**mié 02/08/23**
Iteracción 1	**10 días**	**mar 13/06/23**	**lun 26/06/23**
Modelado de Negocio	**1 día**	**mar 13/06/23**	**mar 13/06/23**
Reunión con los Tutores	1 día	mar 13/06/23	mar 13/06/23
Requisitos	**1 día**	**mié 14/06/23**	**mié 14/06/23**
Finalización de Requisitos	1 día	mié 14/06/23	mié 14/06/23
Análisis	**1 día**	**jue 15/06/23**	**jue 15/06/23**
Analizar los Resultados	1 día	jue 15/06/23	jue 15/06/23
Diseño	**5 días**	**vie 16/06/23**	**jue 22/06/23**
Inicio de la Documentación	5 días	vie 16/06/23	jue 22/06/23
Implementación	**1 día**	**vie 23/06/23**	**vie 23/06/23**
Aplicar cambios en los objetivos	1 día	vie 23/06/23	vie 23/06/23
Pruebas	**1 día**	**lun 26/06/23**	**lun 26/06/23**
Prueba del Sistema Completo	1 día	lun 26/06/23	lun 26/06/23
Fin Iteracción 1	0 días	lun 26/06/23	lun 26/06/23

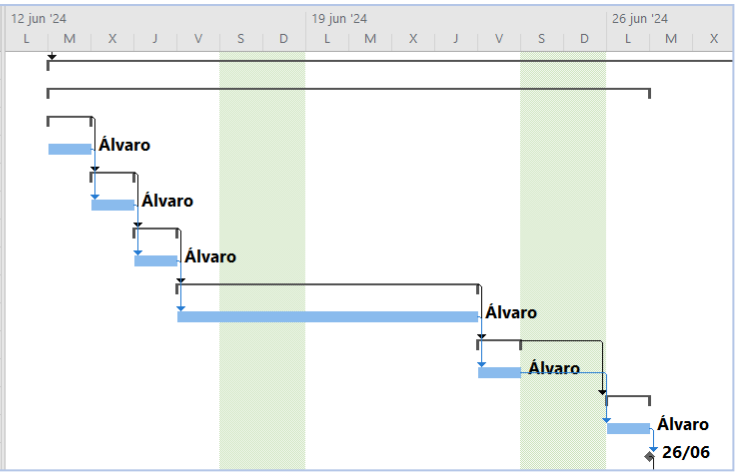


Figura 24 Diagrama de Gantt (Parte 10)

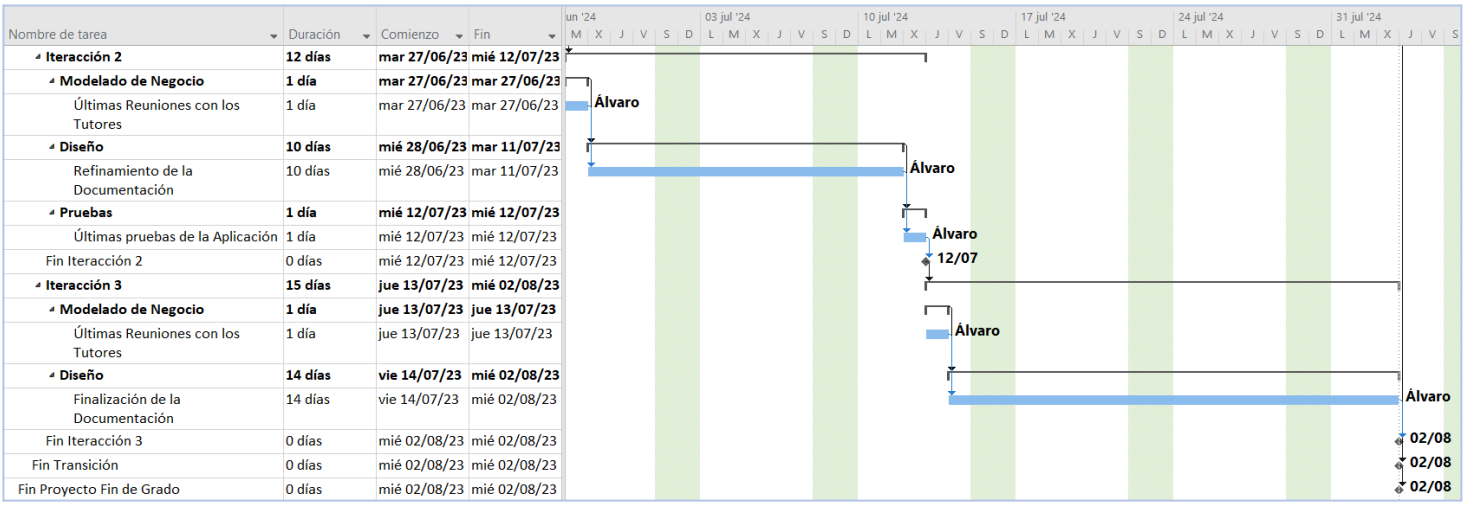


Figura 25 Diagrama de Gantt (Parte 11)

De todas las tareas mostradas en el diagrama de Gantt, cabe destacar las principales o más importantes el proceso:

- Reunión con los Tutores: esta es una de las tareas más importante y repetida del proyecto, pues de no ser por la ayuda y el feedback de los tutores, no habría desarrollado el proyecto de una manera tan clara y eficaz.
- Prototipo de Diseño de Arquitectura: en la primera fase de inicio, se diseña un prototipo de diseño de la arquitectura que será muy clave para sentar las bases en el diseño de la arquitectura y mostrar el enfoque de la aplicación. Poco a poco se irá refinando con exactitud el prototipo hasta que finalmente se complete.
- Implementación de Objetivos: la columna vertebral del proyecto, por decirlo así, esta tarea engloba todos los objetivos principales desarrollados e implementados en el entorno de programación, como se ha comentado anteriormente forma parte de la fase del proceso con mayor implicación de tiempo pues se trata de la parte más compleja.

- Refinamiento de Objetivos: tras las sucesivas reuniones, se han ido modificando y mejorando los objetivos gracias al feedback recibido por los tutores del proyecto.
- Pruebas del Objetivo-04: El Objetivo-04, Gestionar el Modelo Depuradora de Agua, es el más amplio, complejo y que más dificultades me ha generado. No es ninguna sorpresa que la tarea de realizar las pruebas, para que no exista ningún error, sea una de las más importantes.
- Pruebas del Sistema Completo: las pruebas del sistema completo también son muy importante pues hay que asegurar que el sistema en conjunto funcione correctamente y estén todos los modelos interconectados de forma clara y eficaz para el usuario.
- Inicio de la Documentación: una vez está todo el sistema diseñado y construido, sólo queda documentar todo el proceso realizado siguiendo los pasos secuenciales que nos han llevado a la correcta finalización del proyecto.

6.3. Especificación de Requisitos

En este capítulo se detallarán todos los requisitos definidos en la primera parte del proyecto después de varias reuniones con los tutores.
En estas reuniones se decidía cuáles serían las funcionalidades principales de la aplicación, los objetivos principales:

OBJ-01	Gestionar Circuito
Versión	1.0 (13/04/2023)
Autores	Álvaro García Labrador
Fuentes	
Descripción	El sistema permitirá al usuario gestionar todas las funcionalidades que permita el modelo eléctrico (Circuito RC)
Subobjetivos	Ninguno
Importancia	Media-Alta
Urgencia	
Estado	En construcción
Estabilidad	Alta
Comentarios	Gestiona las diferentes funcionalidades que ofrece al usuario para interactuar con el modelo eléctrico

OBJ-02	Gestionar Péndulo
Versión	1.0 (17/04/2023)
Autores	Álvaro García Labrador
Fuentes	
Descripción	El sistema permitirá al usuario gestionar todas las funcionalidades que permita el modelo físico (Péndulo)
Subobjetivos	Ninguno
Importancia	Media
Urgencia	

Estado	En construcción
Estabilidad	Alta
Comentarios	Gestiona la funcionalidad que ofrece el modelo físico

OBJ-03	Gestionar Tanques Acoplados
Versión	1.0 (27/04/2023)
Autores	Álvaro García Labrador
Fuentes	
Descripción	El sistema permitirá al usuario gestionar todas las funcionalidades que permita el modelo hidráulico (Tanques Acoplados). Además, este modelo, al ser un modelo algo más complejo que los anteriores, tiene una conexión con Simulink, lo que le permitirá más funcionalidades de modificación de variables y mayores facilidades de interacción con la interfaz de cara al usuario. La visualización en la propia ventana de las gráficas de los tanques acoplados, es decir, la altura del agua en cada tanque en el intervalo de tiempo especificado en la simulación.
Subobjetivos	Ninguno
Importancia	Alta
Urgencia	
Estado	En construcción
Estabilidad	Alta
Comentarios	Gestiona las diferentes funcionalidades que ofrece al usuario para interactuar con el modelo hidráulico (Tanques acoplados)

OBJ-04	Gestionar Depuradora de Agua
Versión	1.0 (01/05/2023)
Autores	Álvaro García Labrador
Fuentes	
Descripción	Este es el objetivo más completo ya que involucra al modelo avanzado del proyecto. Permitirá al usuario muchas más funcionalidades que los modelos anteriores y al igual que el modelo de Tanques Acoplados, tiene una conexión con Simulink. Permite la modificación de las variables de control, su simulación, el guardado en un fichero *.mat de los resultados, la carga de un fichero de simulación, visualización de todas las gráficas generadas después de la simulación, comparación de las gráficas de diferentes simulaciones y la generación de tanto un informe de planta como del controlador involucrado en la depuradora de agua en el intervalo de tiempo especificado.
Subobjetivos	Ninguno
Importancia	Muy Alta
Urgencia	
Estado	En construcción
Estabilidad	Alta
Comentarios	Gestiona las diferentes funcionalidades que ofrece al usuario para interactuar con el modelo hidráulico (Depuradora de Agua)

6.4. Etapa de Análisis

Una vez que se han recogido todos los tipos de requisitos del sistema, entramos en la fase de análisis donde se mostrará el diagrama de clases, que como su nombre indica, está compuesto por todas las clases necesarias para la realización del sistema y todas las relaciones entre las mismas:

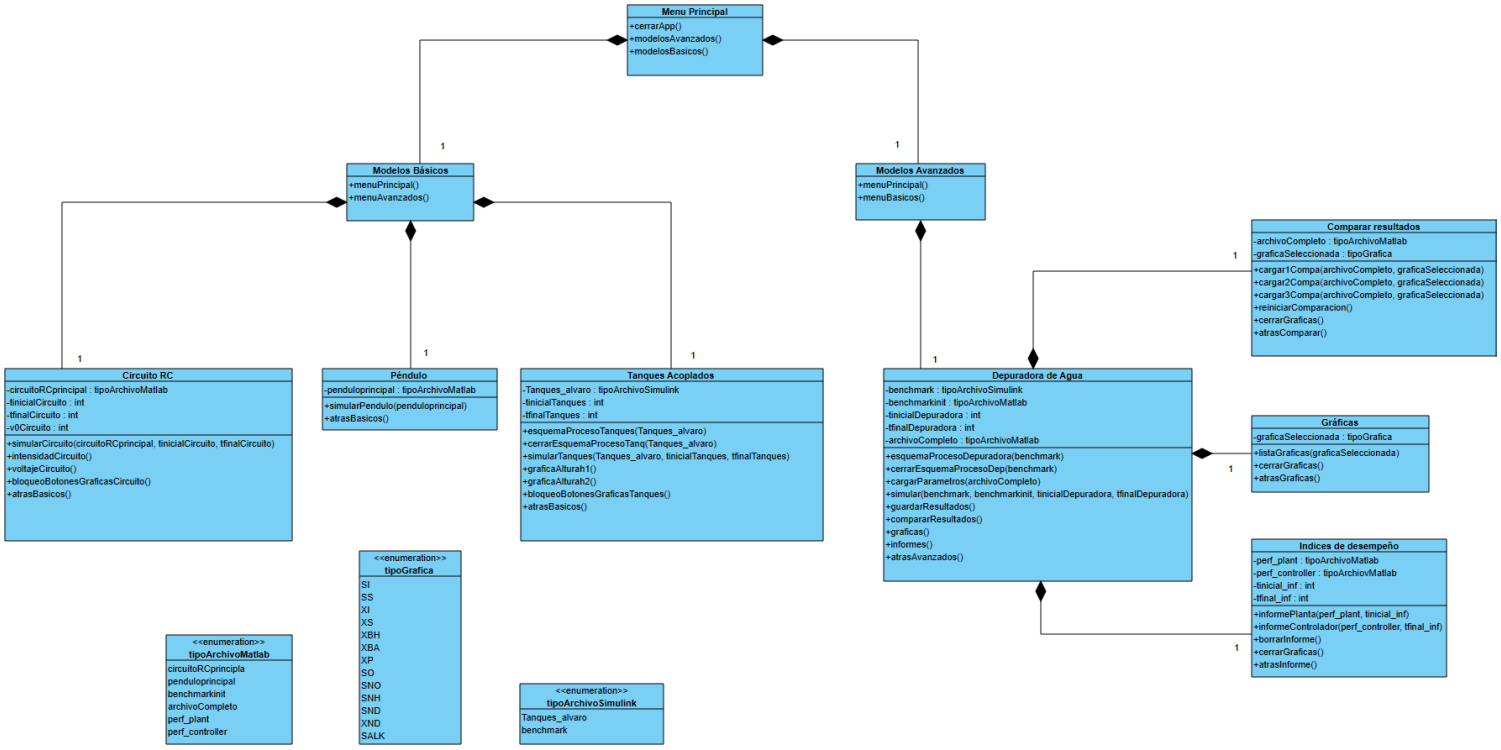


Figura 26 Diagrama de clases

A continuación, explicaré detalladamente cada atributo y método incluido en cada clase:

- **Menú Principal**: esta clase sería la clase principal que nos sirve como marco de trabajo y funcionalidades principal. Es la ventana principal y estaría compuesta por dos tipos de clases: Modelos Básicos y Modelos Avanzados.
  - cerrarApp(): este método haría un correcto cierre de toda la aplicación y sería accesible desde esta misma clase.
  - modelosAvanzados(): nos llevaría directamente a la ventana donde se encuentran los modelos avanzados (en este caso sería sólo uno, la depuradora de agua).

- modelosBasicos(): este método nos dirigiría a la ventana de modelos básicos dónde podremos encontrar los siguientes modelos: circuito eléctrico, péndulo y tanques acoplados.

- Modelos Básicos: esta clase está compuesta por otras tres: Circuito RC, Péndulo y Tanques Acoplados.
  - menuPrincipal(): un método que redirige al usuario de nuevo a la página principal de la aplicación.
  - menuAvanzados(): permite al usuario moverse a la ventana de modelos avanzados directamente sin pasar por la ventana principal.
- Modelos Avanzados: al igual que la clase anterior, pero esta está dedicada a los modelos más avanzados que estaría compuesta por una sola clase: Depuradora de Agua.
  - menuPrincipal()
  - menuBasicos(): permite al usuario moverse a la ventana de modelos básicos desde la actual.
- Circuito RC: esta clase representa la ventana que permite al usuario usar todas las funcionalidades que ofrece.
    - circuitoRCprincipal: tipoArchivoMATLAB
    - tinicialCircuito: int
    - tfinalCircuito: int
    - v0Circuito: int
  - simularCircuito(circuitoRCprincipal, tinicialCircuito, tfinalCircuito): este es el método que realiza la simulación del circuitoRC tomando el archivo de MATLAB y ejecutándolo en el propio MATLAB con los tiempos iniciales y finales introducidos en la propia ventana.
  - intensidadCircuito(): este método lo que hace es tomar las variables del tiempo e intensidad que se han generado después de la simulación y genera una gráfica con el resultado en la propia ventana.
  - voltajeCircuito(): igual que el método de intensidadCircuito() pero con las variables de voltaje y tiempo.
  - bloqueoBotonesGraficasCircuito(): este método, reinicia la simulación justo cuando cambias ya sea el tiempo inicial o final.
  - atrasBasicos(): redirige al usuario a la ventana principal.
- Péndulo: esta clase representa la ventana de otro modelo básico.
    - Penduloprincipal: tipoArchivoMATLAB

- simularPendulo(penduloprincipal): ejecuta el archivo de MATLAB señalado y dibuja en la propia gráfica de la ventana los resultados de la simulación.
- atrasBasicos()

➢ Tanques Acoplados: la última clase que compone los modelos básicos, esta clase representa la ventana de la interfaz de tanques acoplados y como se puede observar en la cantidad de métodos, es algo más compleja de las clases anteriores.

  - Tanques_alvaro: tipoArchivoSimulink
  - tinicialTanques: int
  - tfinalTanques: int

- esquemaProcesoTanques(Tanques_alvaro): este método llama al modelo "Tanques_alvaro" de Simulink y lo abre en segunda pantalla.
- cerrarEsquemaProcesoTanq(Tanques_alvaro): cierra el modelo "Tanques_alvaro".
- simularTanques(Tanques_alvaro, tinicialTanques, tfinalTanques): este método ejecuta una simulación del modelo de tanques acoplados especificado en el primer campo con un tiempo inicial y final introducido por el usuario a través de dos campos editables.
- graficaAlturah1(): lo que este método hace es dibujar la gráfica de la altura del agua del primer tanque en la propia ventana de la interfaz.
- graficaAlturah2(): igual que graficaAlturah1() pero dibujando la altura del agua del segundo tanque
- bloqueoBotonesGraficasTanques(): este método, reinicia la simulación justo cuando cambias ya sea el tiempo inicial o final
- atrasBasicos()

➢ Depuradora de Agua: esta clase representa todos los atributos y métodos que proporciona el modelo de la depuradora de agua. Esta clase representa el modelo más complejo y con más funcionalidades, está compuesta por: Comparar resultado, Gráficas y Índices de desempeño.

  - Benchmark: tipoArchivoSimulink
  - Benchmarkinit: tipoArchivoMATLAB
  - tinicialDepuradora: int
  - tfinalDepuradora: int
  - archivoCompleto:tipoArchivoMATLAB

- esquemaProcesoDepuradora(benchmark): abre el esquema del proceso “benchmark” del propio Simulink para que el usuario pueda modificar las variables que le interesen.
- cerrarEsquemaProcesoDep(benchmark): cierra el esquema “benchmark”.
- cargarParametros(archivoCompleto): otro método alternativo antes de realizar la simulación es cargar directamente al archivo con la simulación ya realizada: archivoCompleto. Este método carga el archivo en la base de MATLAB.
- simular(benchmark, benchmarkinit, tinicialDepuradora, tfinalDepuradora): ejecuta el archivo “benchmarkinit” el cual ejecuta los archivos necesarios para realizar la simulación, son los archivos iniciales antes de realizar la simulación. Después realiza la simulación del modelo “benchmark” con el intervalo de simulación introducido por el usuario. Una vez hecha la simulación permite el uso de otras dos funcionalidades: mostrar las gráficas y ver los informes de la planta y el controlador.
- guardarResultados(): permite salvar el archivoCompleto después de haber realizado una simulación para su posterior uso.
- compararResultados(): dirige al usuario a la ventana donde podrá comparar diferentes resultados de simulaciones.
- graficas(): dirige al usuario a la ventana donde podrá visualizar las diferentes gráficas generadas después de simular.
- informes(): dirige al usuario a la ventana de índices de desempeño, aquí podrá visualizar los informes de la planta y el controlador en el intervalo deseado después de la simulación.
- atrasAvanzados(): redirige al usuario a la ventana principal.

➢ Comparar resultados: representa las diferentes funcionalidades de la ventana homónima.
    - archivoCompleto: tipoArchivoMATLAB
    - graficaSeleccionada: tipoGrafica
- cargar1Compa(archivoCompleto, graficaSeleccionada): permitiría al usuario seleccionar un archivo de tipo MATLAB de alguna simulación salvada anteriormente, cargarla en la base de MATLAB y sacar por pantalla la gráfica seleccionada con los datos de esa simulación.
- cargar2Compa(archivoCompleto, graficaSeleccionada): tiene la misma funcionalidad que el método anterior con el matiz de

que si alguno de estos tres métodos de carga se ha ejecutado, entonces se dibujarán los datos de este archivo en la gráfica generada anteriormente.

- cargar3Compa(archivoCompleto, graficaSeleccionada): igual que los otros dos métodos de carga para la comparación.
- reiniciarComparacion(): permite al usuario volver a seleccionar el tipo de gráfica que quiere comparar y volver a cargar simulaciones.
- cerrarGraficas(): cierra todas las ventanas de tipo figure que se hayan abierto anteriormente.
- atrasComparar():redirige al usuario a la ventana de modelos avanzados.

➢ Gráficas:

  - graficaSeleccionada: tipoGrafica

- listaGraficas(graficaSeleccionada): abre una pestaña de tipo figure de la gráfica que se haya seleccionado para visualizar.
- cerrarGraficas()
- atrasGraficas():redirige al usuario a la ventana de modelos avanzados.

➢ Índices de desempeño:

  - perf_plant: tipoArchivoMATLAB
  - perf_controller: tipoArchivoMATLAB
  - tinicial_int: int
  - tfinal_inf: int

- informePlanta(perf_plant, tinicial_inf): muestra en la propia ventana un informe detallado de la planta (perf_plant) sobre la última simulación ejecutada en un intervalo de tiempo introducido por el usuario y abre pestañas de tipo figure con gráficas sobre la misma.
- informeControlador(perf_controller, tfinal_inf): muestra en la propia ventana un informe detallado del controlador de la planta (perf_controller) sobre la última simulación ejecutada en un intervalo de tiempo introducido por el usuario y abre pestañas de tipo figure con gráficas sobre el mismo.
- borrarInforme(): este método, borra los informes mostrados en la ventana cuando se introduce un nuevo tiempo de simulación.
- cerrarGraficas()
- atrasInforme(): redirige al usuario a la ventana de modelos avanzados.

También mencionar las clases de enumeración que nos permiten definir otros tipos de atributos:

- <<enumeration>> tipoArchivoMATLAB
- <<enumeration>> tipoGrafica
- <<enumeration>>tipoArchivoSimulink

6.5. Etapa de Diseño

Una vez realizada la fase de análisis de los requisitos recogidos, se aborda esta etapa de diseño, en la que se define una arquitectura clara para la aplicación, en mi caso usamos el patrón de diseño MVC (Modelo, Vista, Controlador), pues era el que mejor se adaptaba a las funcionalidades y diferentes requisitos analizados.

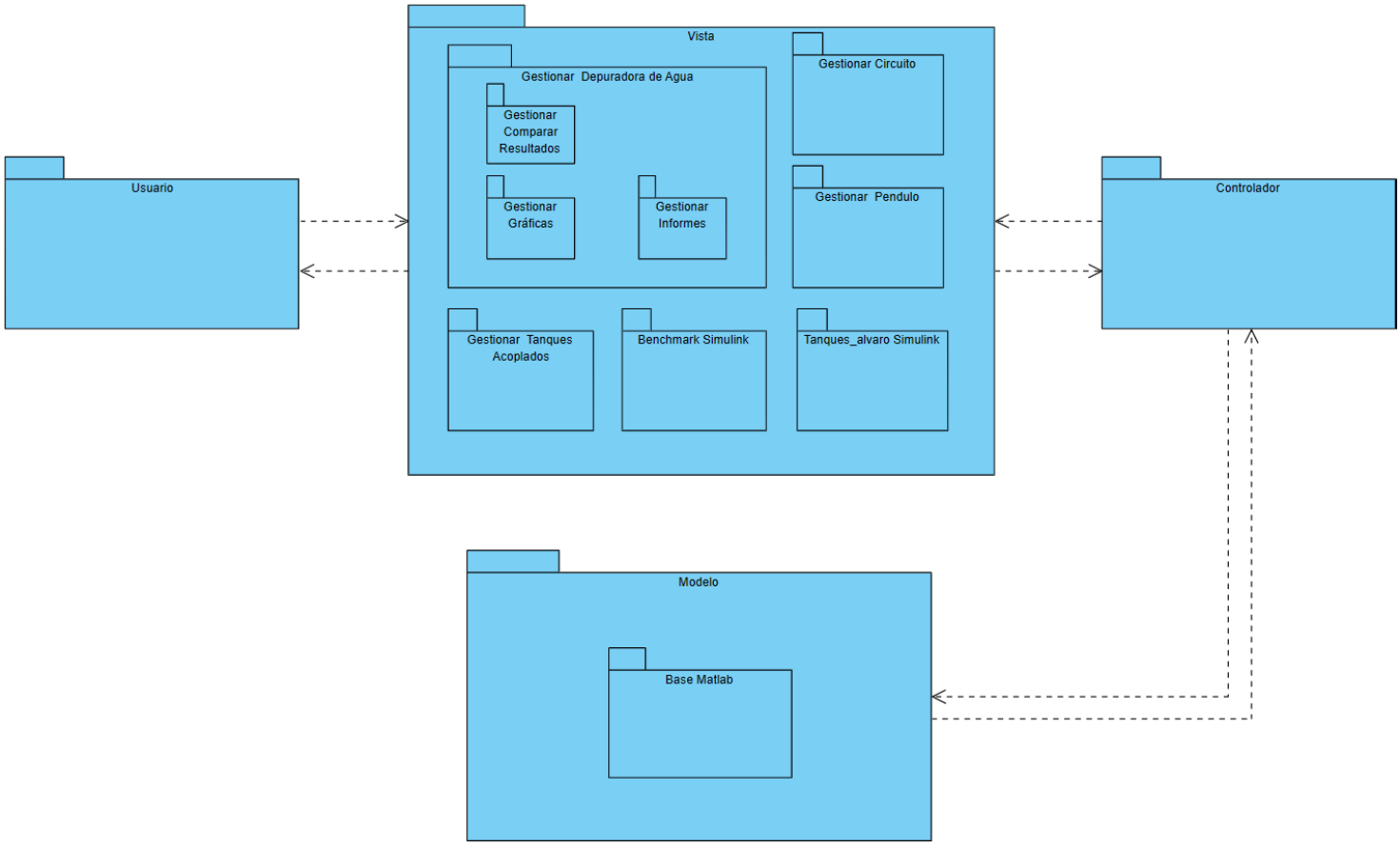


Figura 27 Diseño de arquitectura

Se puede apreciar en la arquitectura como el usuario interacciona directamente con la vista, que contiene las diferentes ventanas con las que puede comunicar.

Como se observa la vista y el modelo no tienen conexión directa, sino que el controlador hace de intermediario entre ambas, una de las características principales de este patrón de diseño.
El modelo está formado por la base de MATLAB, que es donde se definen todos los datos que utilizan los modelos a través del controlador.

A continuación, se mostrarán los diagramas de secuencia que representen las funcionalidades más importantes del sistema con su respectiva explicación de lo que significa cada secuencia de iteraciones:

- Diagrama de Secuencia caso de uso 'abrirEsquemaDepuradora': este diagrama representa la secuencia de acciones necesarias para para abrir el esquema de la depuradora de agua. Importante recalcar que justo antes de mostrar el modelo 'benchmark' también muestra el botón de cerrarEsquemaProceso para cerrarlo cuando el usuario desee.

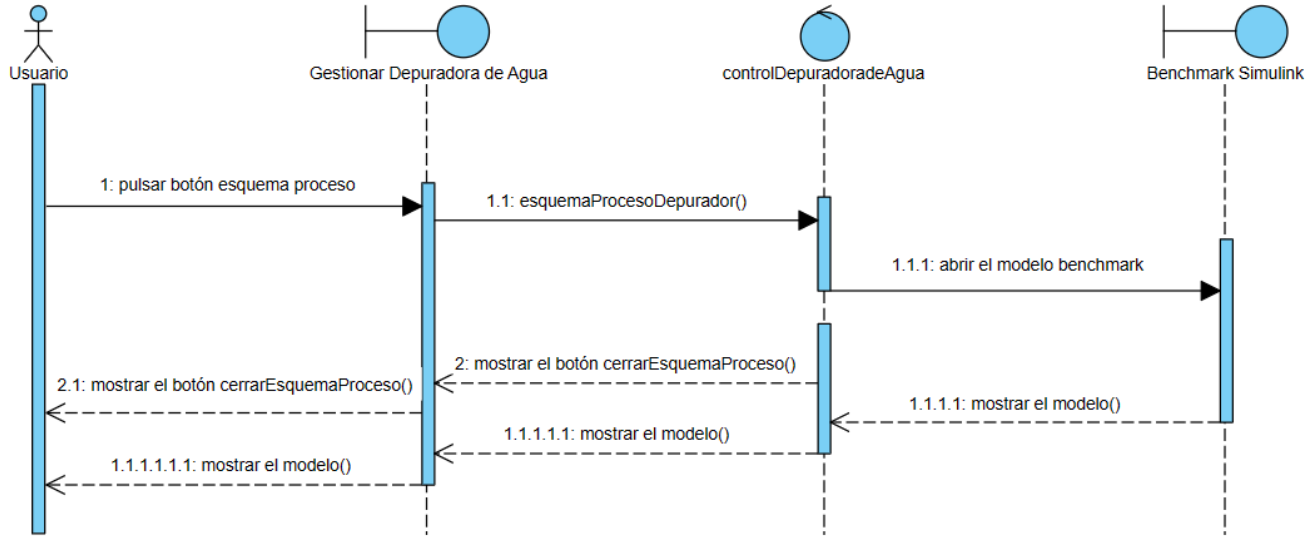


Figura 28 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso ‘cargarParametros’: representa la carga de archivos *.mat compuesto por una simulación realizada anteriormente (con todos los parámetros calculados). Ejecuta el archivo en MATLAB y se guardan todas las variables en base, además se muestra el botón de gráficas y el de índices de desempeño, pues se entiende que al tener las variables necesarias ya son accesibles esas funcionalidades.

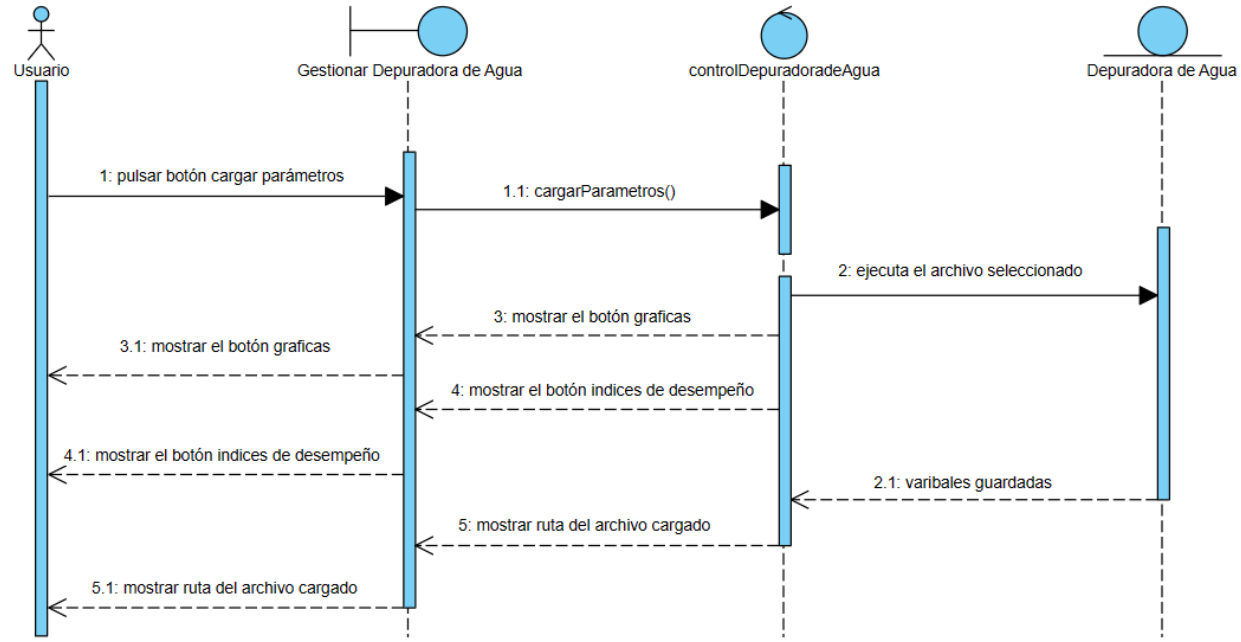


Figura 29 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso ‘simularDepuradora’: la funcionalidad de este caso de uso es paralela a la de ‘cargarParametros’, con la diferencia que en aquí realiza la simulación al instante y muestra un nuevo botón: ‘guardarResultados’ para salvar la simulación y poderla usar posteriormente si se desea.

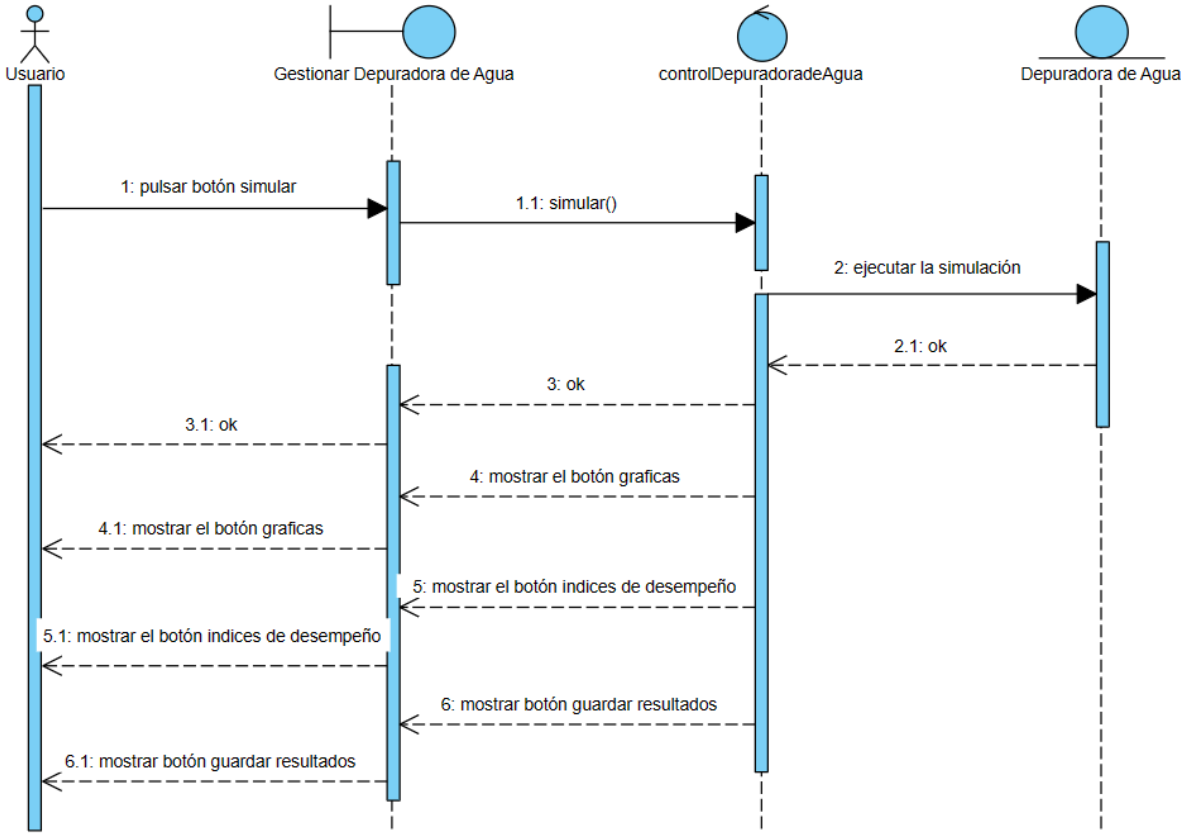


Figura 30 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso 'comparar3Resultados': este diagrama de secuencia muestra las acciones necesarias para cargar 3 archivos *.mat con simulaciones anteriores y visualizar los datos de las 3 simulaciones en la gráfica seleccionada simultáneamente para comparar los diferentes datos.

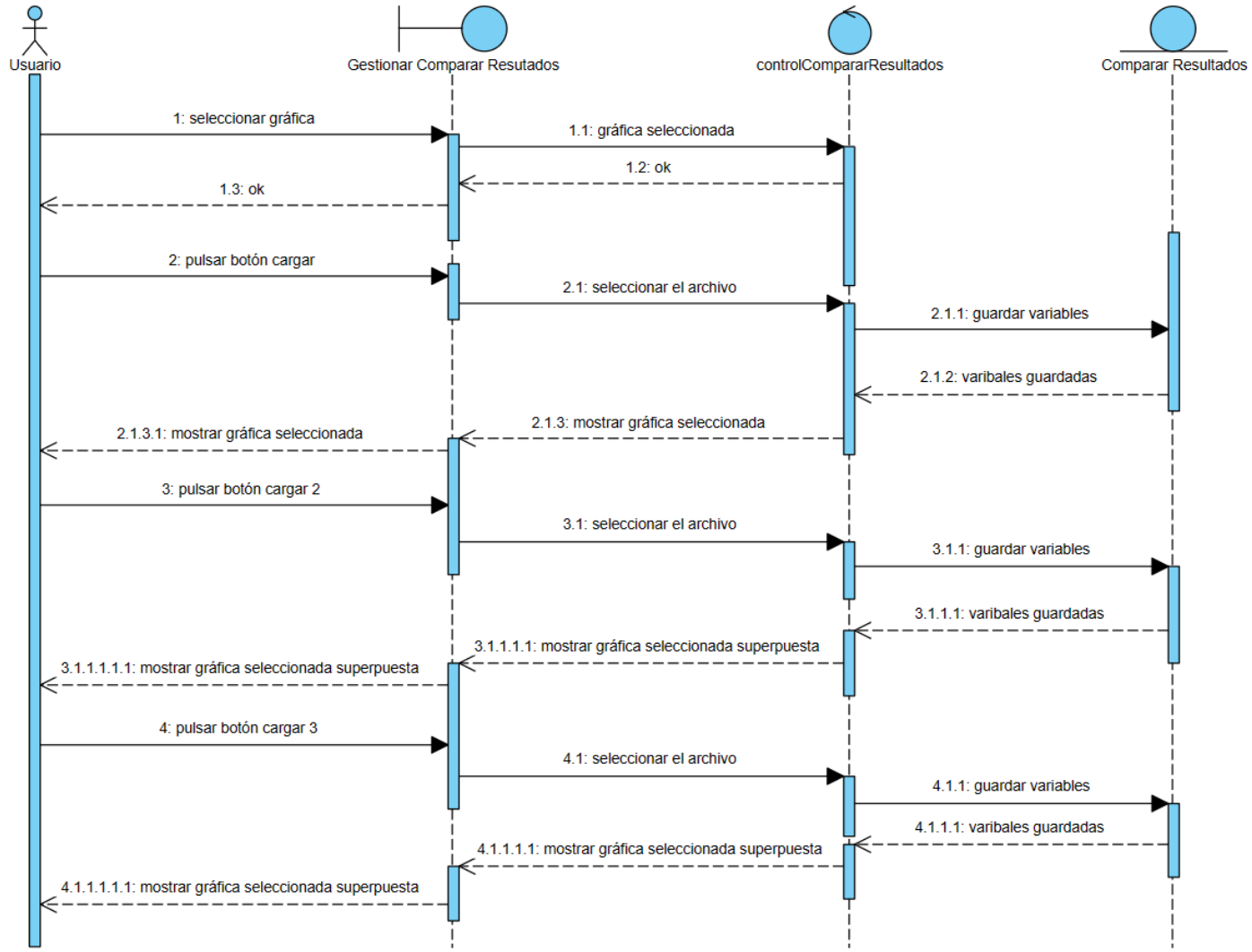


Figura 31 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso 'mostrarGraficaDepuradora': una vez el usuario ha seleccionada la gráfica que quiere visualizar desde la interfaz, el controlador ejecuta el archivo que contiene tal gráfica y la muestra en una ventana alternativa.

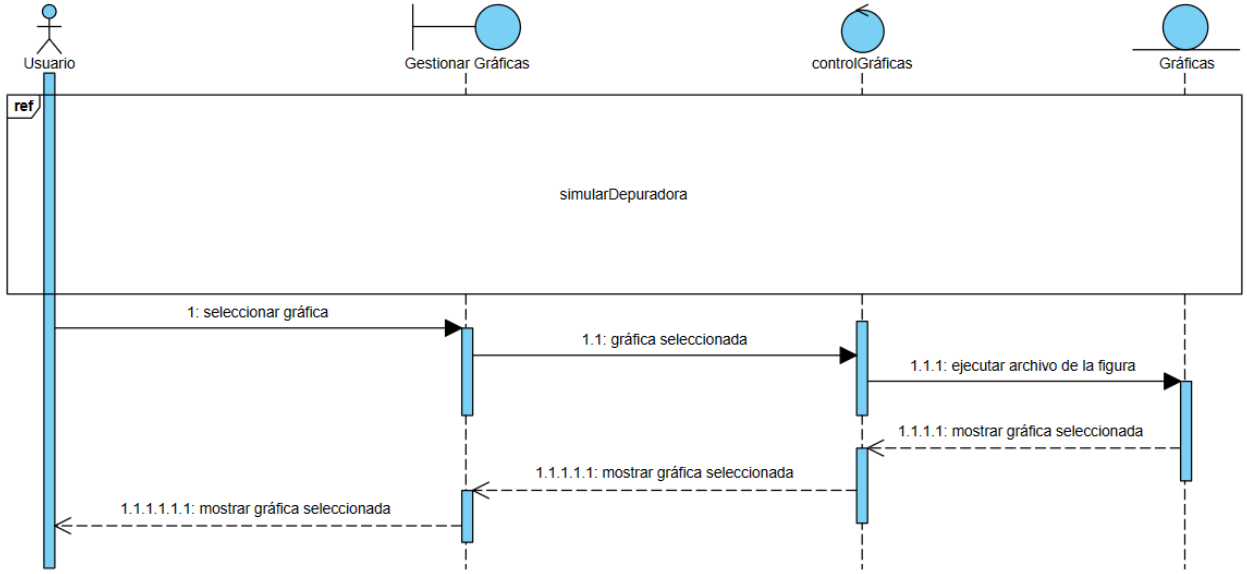


Figura 32 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso 'mostrarInformePlanta': cuando el usuario pulsa el botón de informe de planta, el controlador ejecuta el archivo 'perf_plant' y el modelo devuelve el informe de tipo String y las gráficas de tipo figure. Todo ello se muestra al usuario y además se muestra el botón de informe de controlador, pues ya se puede mostrar el informe del controlador.
  Es una condición necesaria pues no se puede generar el informe del controlador sin antes haber ejecutado el perf_plant.

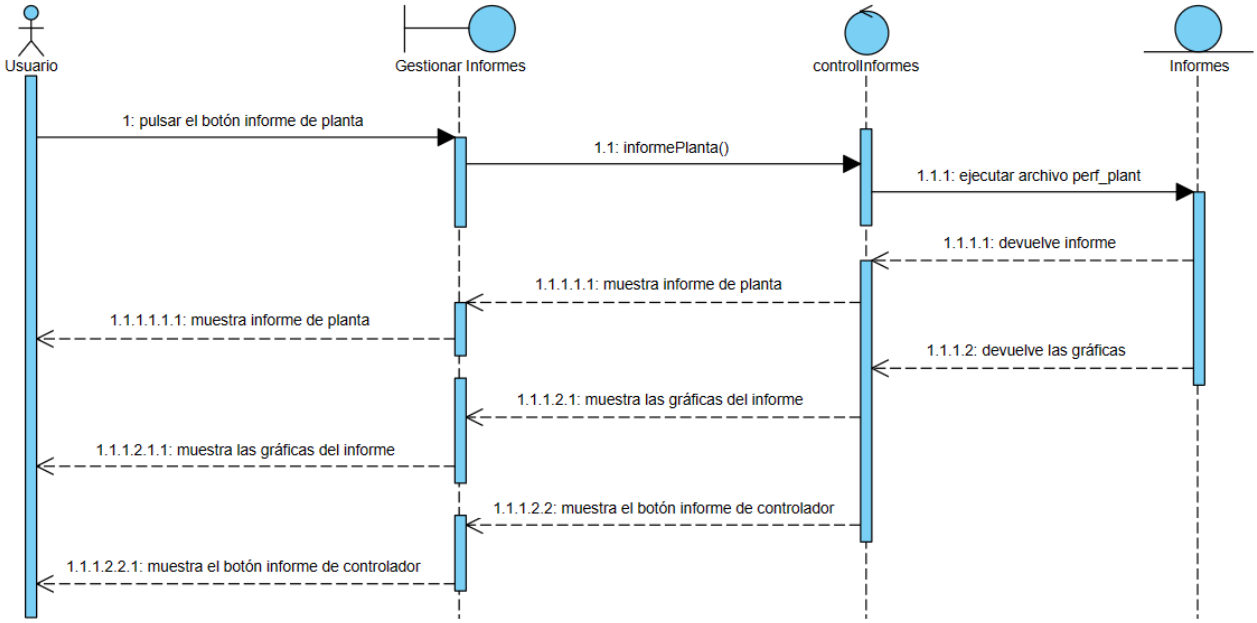


Figura 33 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso ‘mostrarInformeControlador’: como se pone de condición que se haya ejecutado anteriormente el ‘perf_plant’, aquí aparece representado en el diagrama de secuencia, en el caso de que sí se cumpla la condición, ejecuta el ‘perf_controller’, muestra el informe y las gráficas. Si no se cumple no se hace nada.

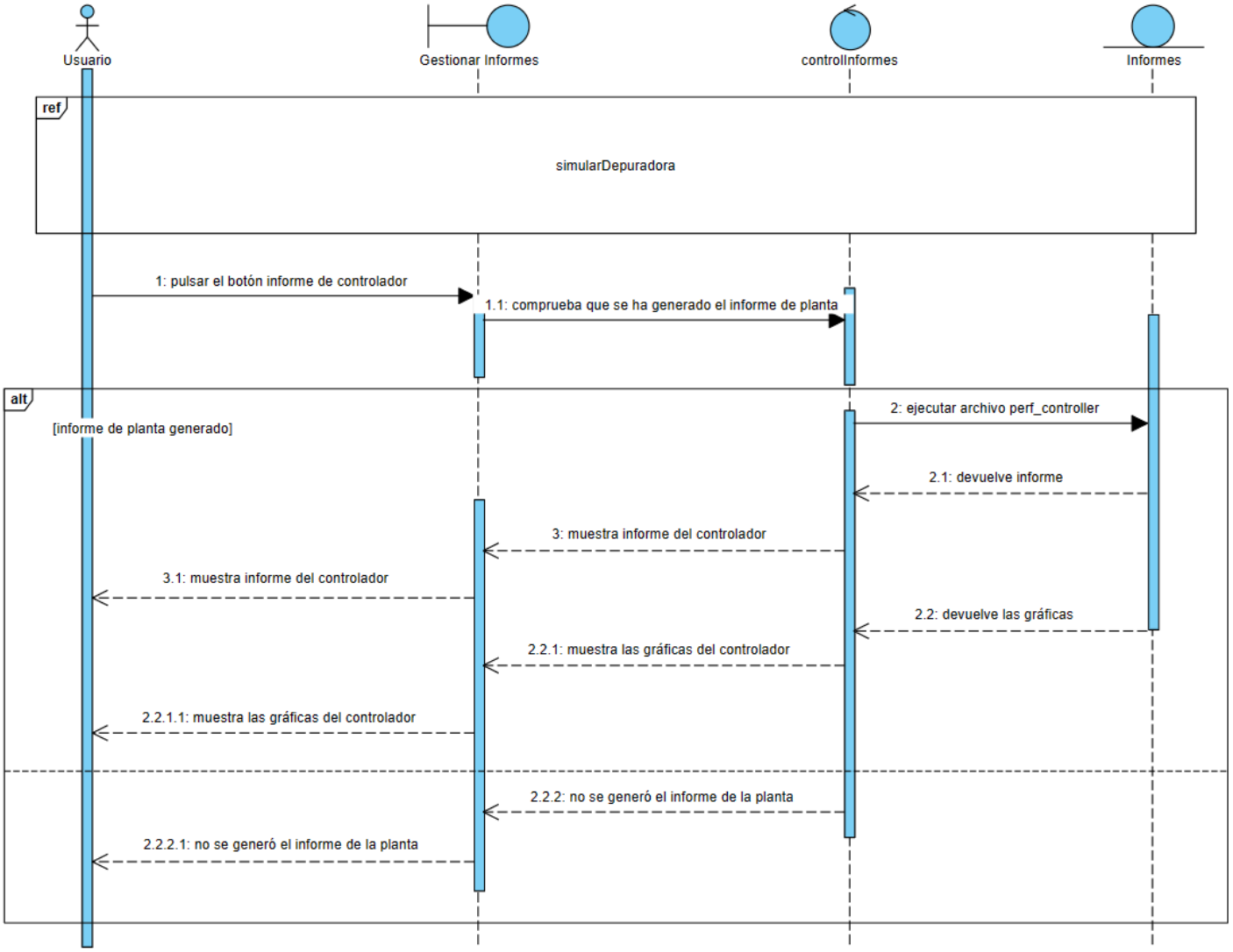


Figura 34 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso 'simularCircuito': muestra la secuencia de acciones para que se ejecute el archivo 'circuitoRCprincipal' al pulsar el botón de simular.

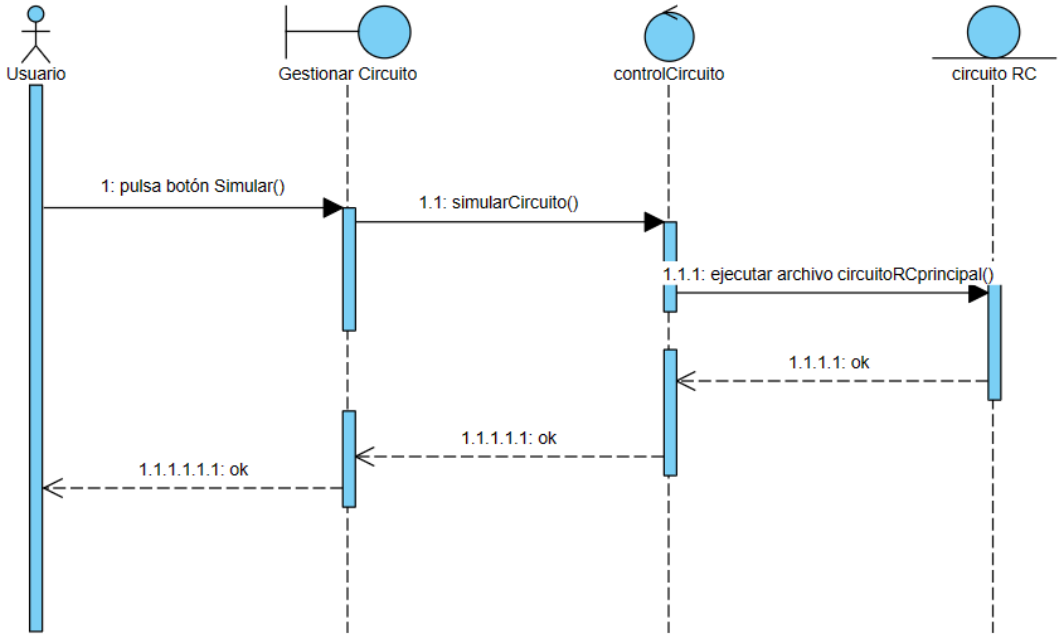


Figura 35 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso 'mostrarGraficaIntensidad': este diagrama del caso de uso de mostrar la gráfica de intensidad, tiene la condición, evidentemente, de que se haya simulado anteriormente, comprueba que en el modelo existan las variables de la simulación y si es así toma los datos necesarios para generar la gráfica en la propia ventana. Si no se ha simulado anteriormente, no hace nada.

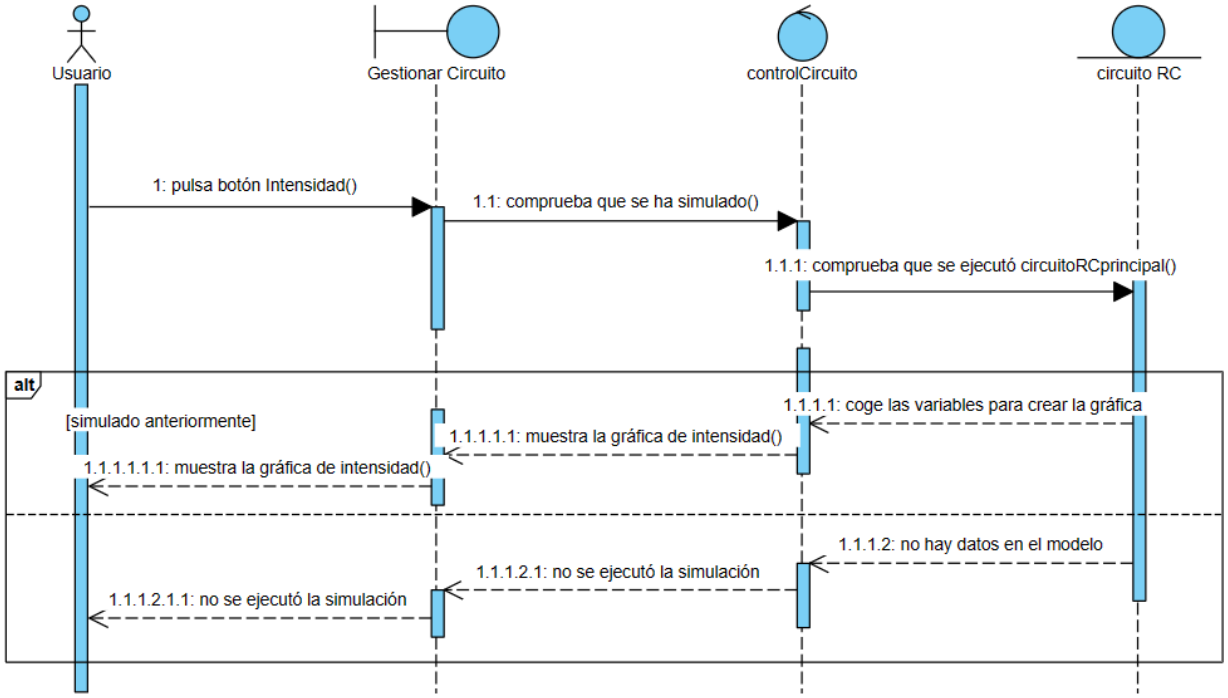


Figura 36 Diagrama de Secuencia

- Diagrama de Secuencia caso de uso 'mostrarGraficaAlturah1': este diagrama es muy parecido al anterior, pero en este caso se trata del modelo de tanques acoplados y se quiere mostrar la gráfica de la altura del agua del primer tanque. Como el anterior se pone de condición que se haya simulado anteriormente.

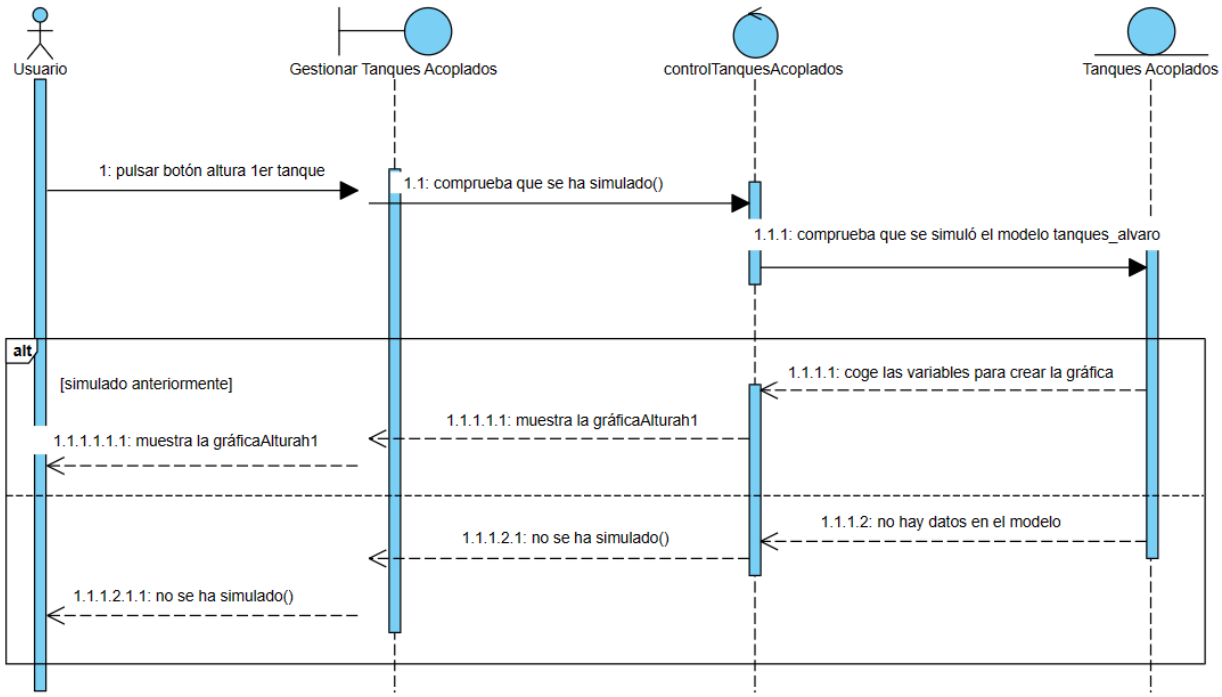


Figura 37 Diagrama de Secuencia

6.6. Etapa de Implementación

Una vez hecha la especificación de requisitos, la etapa de análisis y diseño, ya sólo nos quedaría implementar el sistema basándonos en toda la información recopilada en estas fases anteriores.

Todas estas etapas son estrictamente necesarias para programar un código con una estructura coherente y de una manera eficiente, las funcionalidades ya están claras gracias a los casos de uso, la secuencia de acciones, la arquitectura del sistema, las herramientas a utilizar, etc.

Como se comentó en apartados anteriores, el entorno de trabajo que vamos a utilizar para programar la mayor parte del código es el App Designer de MATLAB, una herramienta proporcionada por la propia empresa que permite diseñar aplicaciones con una conexión directa al propio MATLAB. También hemos utilizado la herramienta Simulink, proporcionada por MATLAB, la cual nos ha permitido crear otro tipo de interfaces llamadas 'Máscaras' que permiten la interacción con el usuario.

Las interfaces de la aplicación se dividen en:

- Modelos Básicos: se trata de los modelos más simples. Dentro de estos encontraríamos: el Circuito RC, el péndulo y los Tanques Acoplados.
- Modelos Avanzados: en los modelos avanzados nos encontramos con el modelo BSM1, la Depuradora de Aguas residuales. Este modelo al ser el más complejo, tiene más funcionalidades, por ende, tendrá tres interfaces más: comparar resultados, gráficas e índices de desempeño.

Además de las interfaces encontradas en la propia aplicación, también se mostrarán los esquemas de proceso de los modelos:

- Tanques Acoplados: este modelo tendrá la funcionalidad de abrir el esquema del proceso, el archivo de Simulink 'tanques_alvaro.slx', y permitirá al usuario la modificación de variables a través de las interfaces de los subsistemas.
- Depuradora de Agua: al igual que el modelo de tanques acoplados, este tendrá las funcionalidades de observar y modificar las variables del esquema del proceso a través las interfaces del archivo de Simulink 'benchmark.mdl'.

6.6.1. Menú Principal

Esta sería la página principal de la aplicación que te muestra las diferentes opciones que tiene el usuario: Modelos Básicos, Modelos Avanzados y cerrar aplicación.

Figura 38 Pantalla Inicial

En la propia ventana y en todas las que veremos a continuación, existe un Menú Desplegable que permite al usuario moverse a cualquier ventana independientemente de la ventana en la que se encuentre. *(Figura 38 Menú Desplegable)*

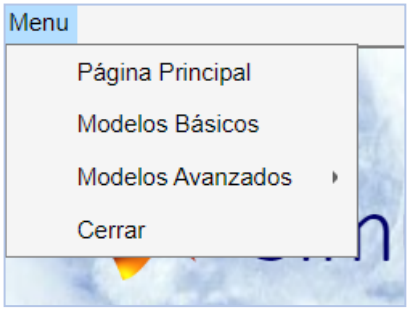


Figura 39 Menú Desplegable

6.6.2. Modelos Básicos

Si el usuario pulsa el botón de Modelos Básicos, la aplicación te redirigirá a la interfaz compuesta por los tres modelos básicos desarrollados: Circuito RC, Péndulo y Tanques Acoplados.

Circuito RC

Esta ventana en cuestión se trata del Circuito que permite al usuario cambiar los valores del tiempo inicial y final de la simulación del modelo y el valor inicial del condensador. Pulsando el botón de Simular, la aplicación resolverá el modelo matemático con la ode45, este método es el utilizado por MATLAB para todos los modelos incluidos en la aplicación.
El ode45 es un algoritmo que divide el intervalo del tiempo en pasos más pequeños y calcula la solución de la ODE en cada paso, utiliza la información de la solución en pasos anteriores para estimar la solución en el siguiente paso.
Una vez generada la simulación el usuario podrá visualizar los resultados en la gráfica mostrada, respecto a la intensidad y al voltaje.

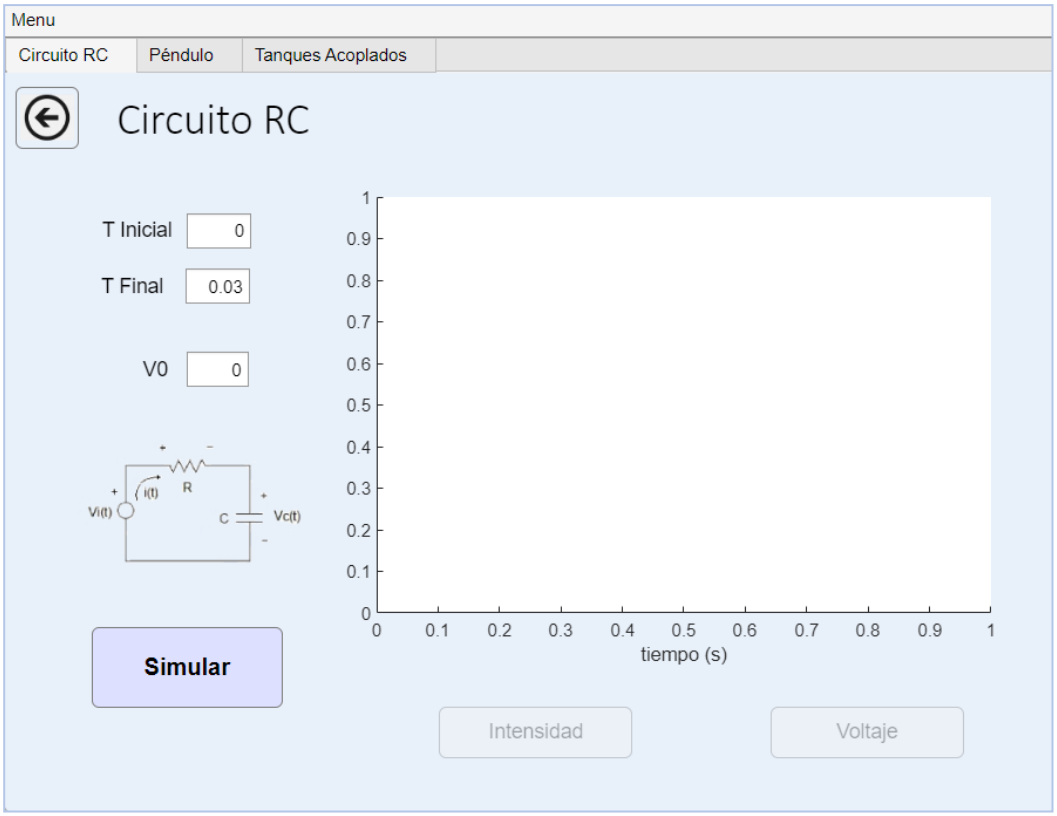


Figura 40 Pantalla Circuito RC

Péndulo

La siguiente ventana seria la del Péndulo, la cual muestra dos gráficas que nos permitirán visualizar los resultados de la simulación del péndulo: la posición del péndulo y la velocidad del péndulo con respecto al tiempo.
Observando el modelo matemático presentado en la sección *4.2.2.Péndulo*, no hay dependencia con la masa m, porque en las ecuaciones se simplifica ese parámetro. Por lo tanto, no hay parámetros modificables.

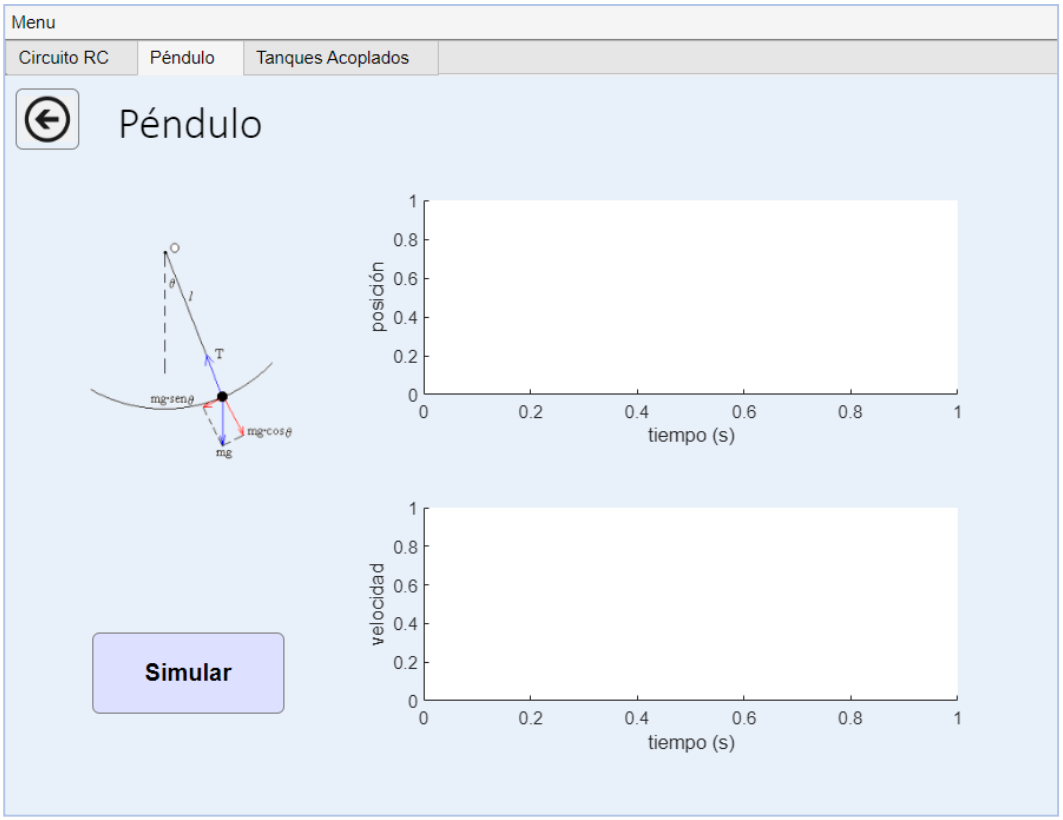


Figura 41 Pantalla del Péndulo

Tanques Acoplados

La última ventana de los Modelos Básicos, Tanques Acoplados. Esta ventana tiene una característica adicional, ya que en los anteriores modelos se utilizan ecuaciones diferenciales para representarlos, en este se utiliza un diagrama de bloques de Simulink. Para la visualización de este esquema basta con pulsar el botón de Esquema de Proceso y que posteriormente se puede cerrar con el botón que se encuentra justo debajo. También se puede cambiar el tiempo de simulación y observar los resultados después de ejecutar la simulación.

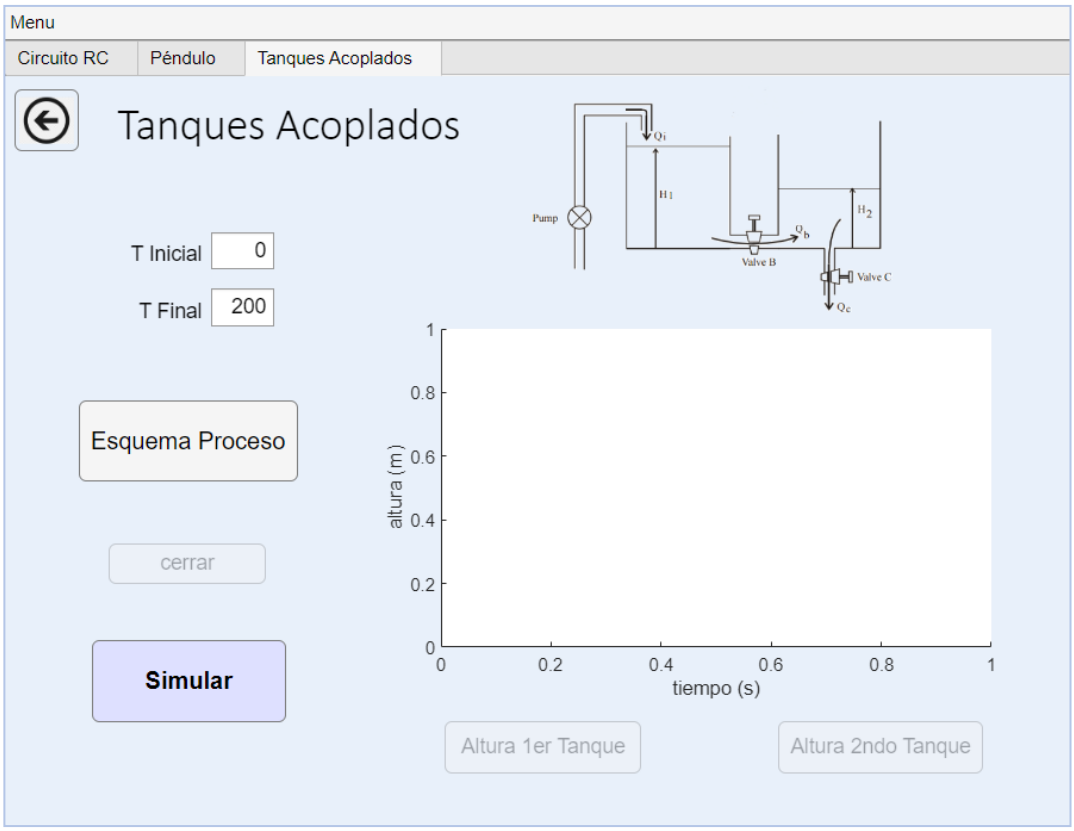


Figura 42 Pantalla de Tanques Acoplados

- Modelo ‘Tanques_alvaro’ de Simulink

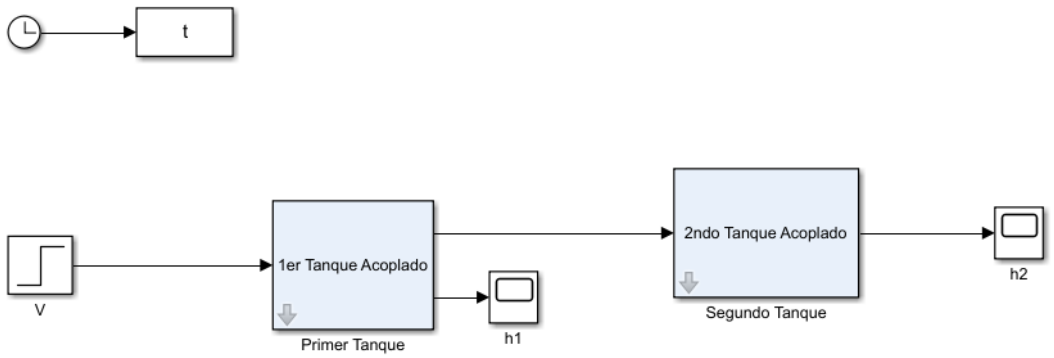


Figura 43 Esquema de Tanques Acoplados (Simulink)

Este sería el esquema de bloques de los Tanques Acoplados, conformado por dos subsistemas (Primer Tanque y Segundo Tanque), cada uno de ellos permiten al usuario modificar valores para su posterior simulación.
Los parámetros modificables son: a superficie del tanque, el radio del orificio de salida. (*Figuras 43 y 44*)

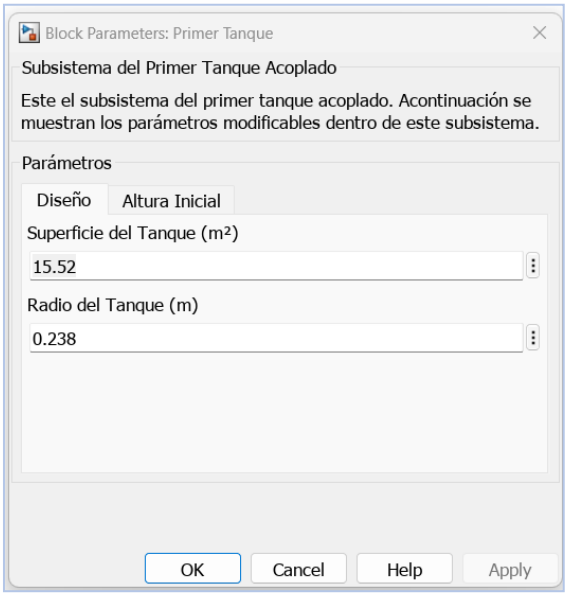


Figura 45 Pestaña de Parámetros del Tanque

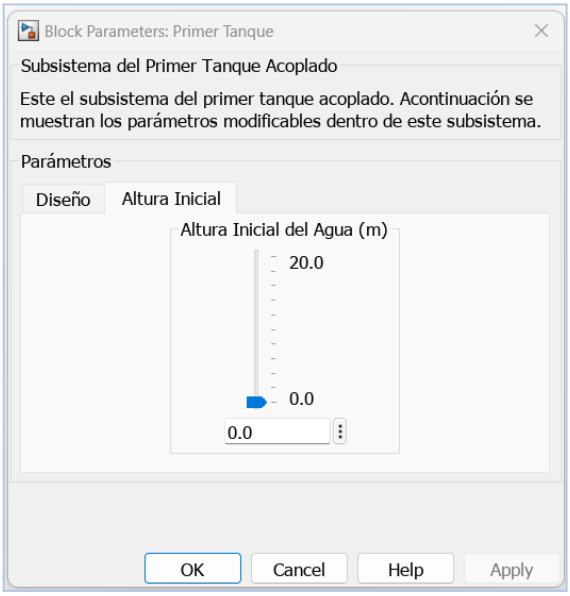


Figura 44 Pestaña de Parámetros del Tanque

Como podemos observar la pestaña de parámetros está dividida en dos pestañas a su vez: Diseño y Altura Inicial.
El diseño se refiere a los parámetros relacionados con el diseño físico del tanque, los cuales se pueden modificar para un resultado alternativo en la

simulación. La Altura Inicial se refiere a la altura del agua inicial que contiene el tanque al inciar la simulación.

6.6.3. Modelos Avanzados

Depuradora de Agua

Si en el menú principal pulsamos el botón de Modelos Avanzados o en un defecto pulsamos en el Menú desplegable, que se encuentra en la esquina superior izquierda, la aplicación nos redirigirá a esta ventana.
Aquí podremos usar las funcionalidades de la Depuradora de Agua.
En una primera instancia tendremos que decidir si queremos iniciar una simulación desde cero o cargar un archivo *.mat con los parámetros de otra simulación ejecutada y salvada anteriormente.

- En caso de la primera opción abriremos el esquema del proceso 'benchmark.mdl' de Simulink y ahí modificaremos los parámetros que consideremos importantes para la simulación.
- En caso de la segunda opción simplemente cargaremos el archivo *.mat y ya podremos visualizar las gráficas y los índices de desempeño de esa simulación.

Una vez se haya ejecutado una simulación o, en su defecto, se haya cargado un archivo, el sistema dejará de restringir la ventana de Gráficas e Índices de Desempeño, que están inaccesibles al usuario hasta que el sistema posee los datos de la simulación.

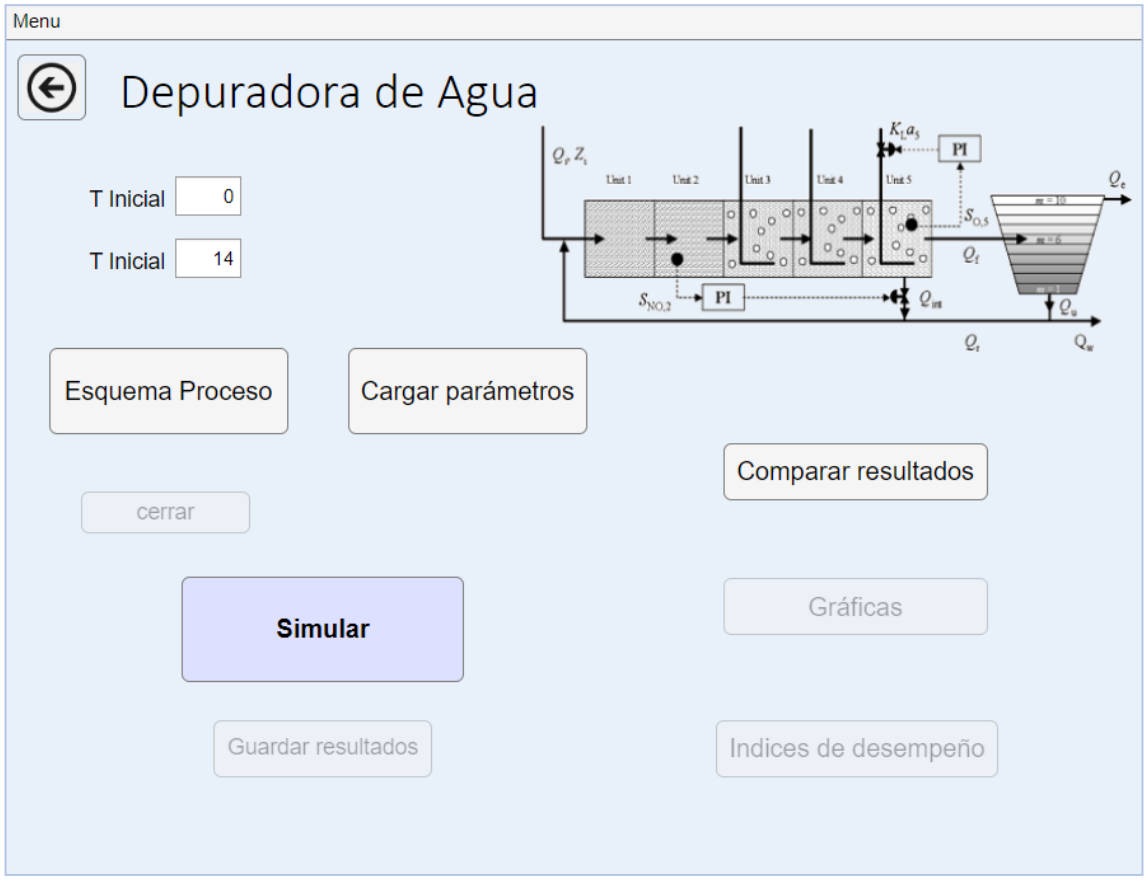


Figura 46 Pantalla de Depuradora de Agua

➢ Comparar Resultados

Esta ventana permite al usuario la comparación de diferentes simulaciones usando como método de visualización la lista de gráficas usadas en este modelo. Esta funcionalidad sólo requiere de los archivos de las simulaciones efectuadas para la propia comparación.
Primero se selecciona el dato que queremos comparar desplegando la lista de gráficas y seleccionando la deseada por el usuario.
Posteriormente se empieza a cargar los archivos de las simulaciones y la aplicación mostrará en una ventana emergente la comparativa de esas simulaciones en ese dato específico.
Si el usuario que compara otro dato, simplemente pulsará el botón reiniciar y entonces podrá volver a seleccionar la gráfica que quiere comparar.

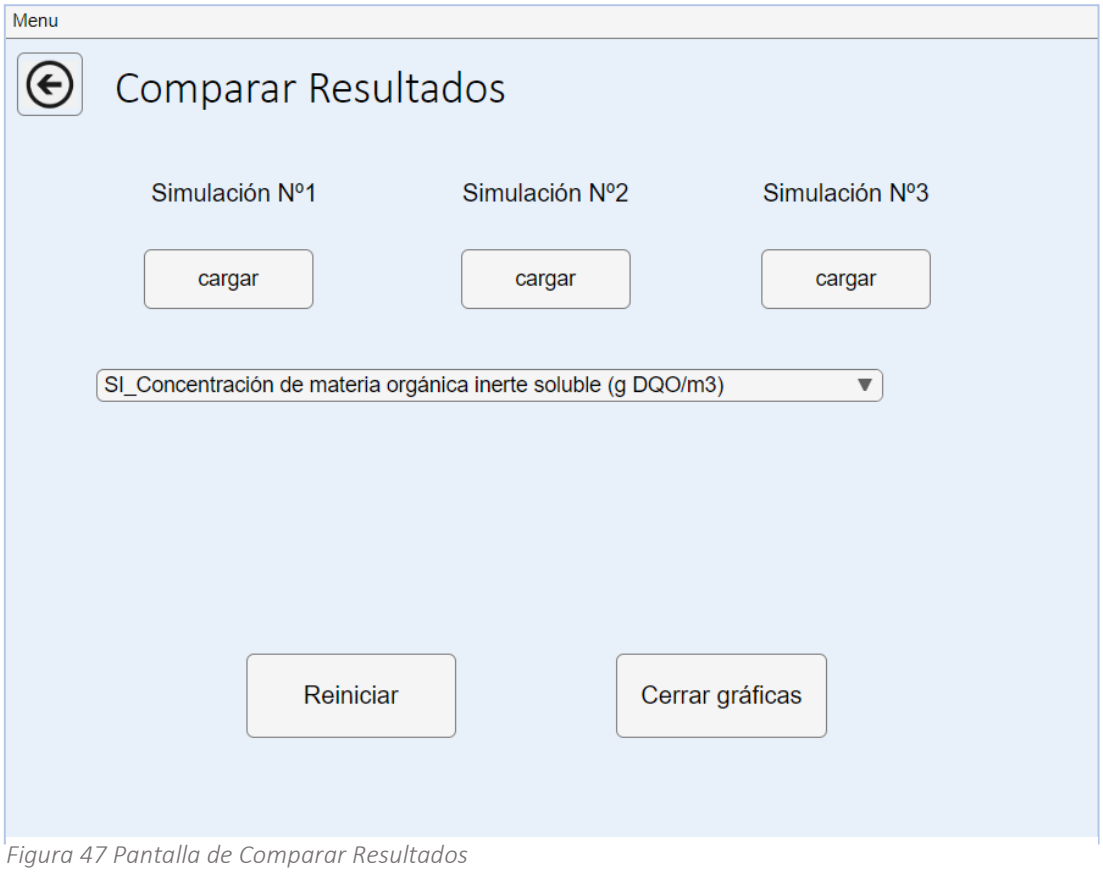


Figura 47 Pantalla de Comparar Resultados

- Gráficas

Esta ventana permite al usuario visualizar los diferentes datos de la simulación pulsando en cada uno de los mostrados en la ventana.
Cuando el usuario pulsa en alguna de ellas, el sistema abrirá una ventana alternativa con la gráfica en cuestión y el usuario podrá abrir las que necesite, cerrando todas las gráficas al instante pulsando el botón cerrar gráficas.

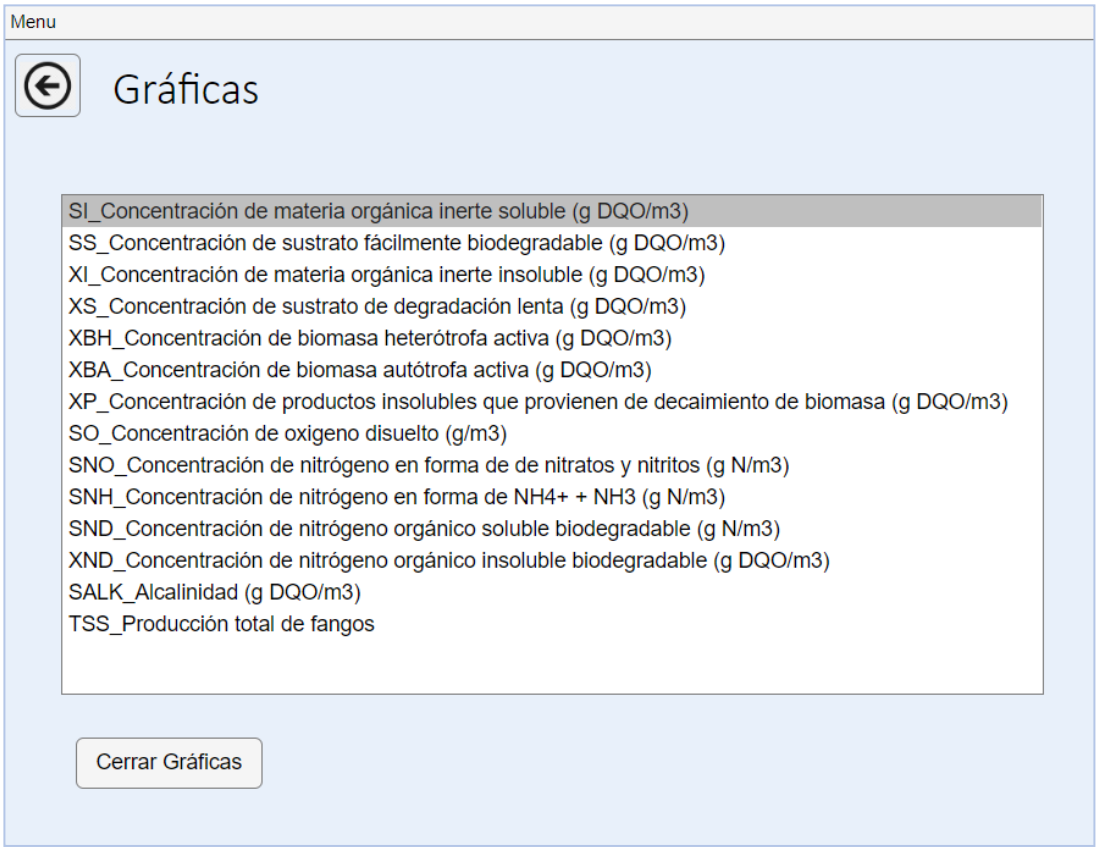


Figura 48 Pantalla de Gráficas

- Informes

La ventana de Índices de Desempeño permite al usuario ver los datos relacionados con la planta y con el controlador en un intervalo de tiempo marcado por el propio usuario. El sistema genera el informe deseado por el usuario en el recuadro que se encuentra en la propia ventana, además de sacar por pantalla las gráficas relacionadas con estos informes.

Figura 49 Pantalla de Informes

- Modelo ‘benchmark’ de Simulink

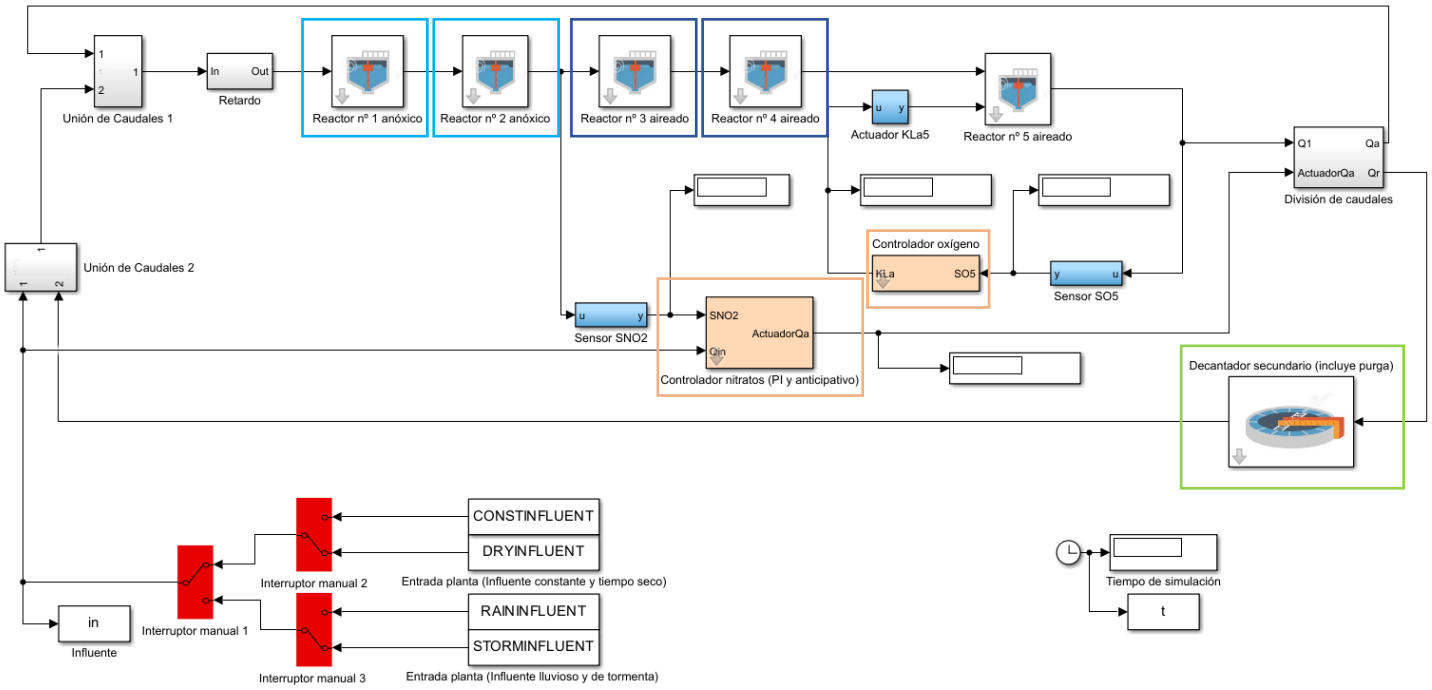


Figura 50 Esquema de la Depuradora de Agua (Simulink)

Este este es el esquema del proceso de la Depuradora de Agua de Simulink que se mostraría al pulsar el botón de Esquema de Proceso, explicado en la ventana de Modelos Avanzados.

Este modelo matemático ‘benchmark’ está compuesto por muchos subsistemas como hemos explicado en la parte de Conceptos Teóricos, pero los que hay que tener más en cuenta son: el primer y segundo Reactor anóxico, el tercero, cuarto y quinto Reactor aireado, el Controlador de Oxígeno, el Controlador de Nitratos y el Decantador Secundario. Estos son los subsistemas accesibles al usuario para modificar sus parámetros.

A continuación, se muestran las pestañas de cada uno:

- Parámetros de Reactores Biológicos: Esta sería la pestaña del Primer reactor que permite modificar los valores de carb1 y KLa1. KLa es el coeficiente de transferencia de oxígeno, que indica la eficiencia en la transmisión de oxígeno del aire al líquido, y que depende de varios factores, entre ellos la agitación del reactor (si el reactor está más agitado, se disolverá más oxígeno). carb indica la cantidad de carbono externo que se añade en los distintos biorreactores. Para que se produzcan adecuadamente las reacciones de desnitrificación, es necesaria la presencia de carbono (materia orgánica). Esto es la materia orgánica que llega del influente, pero algunas veces no es suficiente, y para incrementar la eficiencia de las reacciones químicas, se puede añadir carbono adicional.

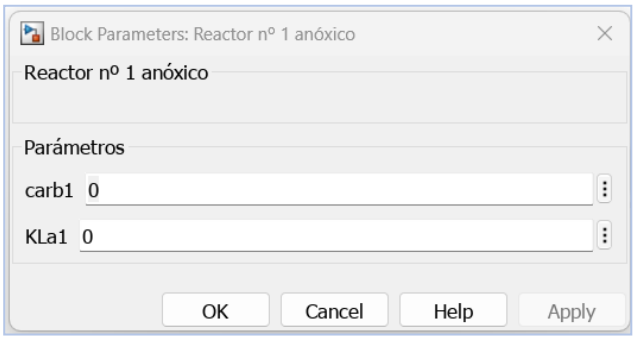


Figura 51 Pestaña de Parámetros de Reactor anóxico

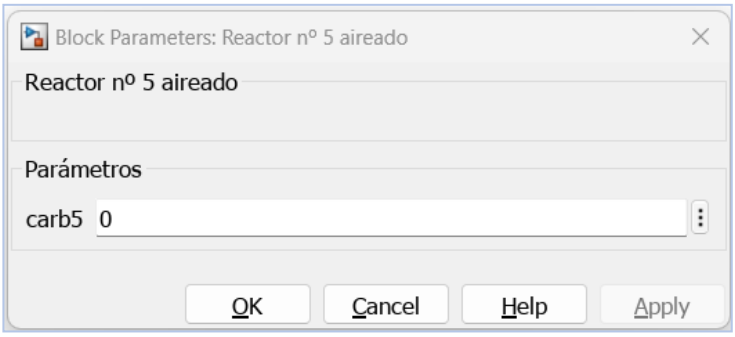


Figura 52 Pestaña de Parámetros de Reactor aireado

- Pestaña de Controlador de Oxígeno: Este sería el controlador de oxígeno que como se observa, está divido en la pestaña de Referencia, que sería el valor de referencia del propio controlador, y la de Parámetros, los parámetros modificables del controlador. El controlador es un PI, por lo que hay que sintonizar definiendo la constante proporcional Kp y el tiempo integral (Ti), y también la constante antiwindup, que limita el efecto integral.

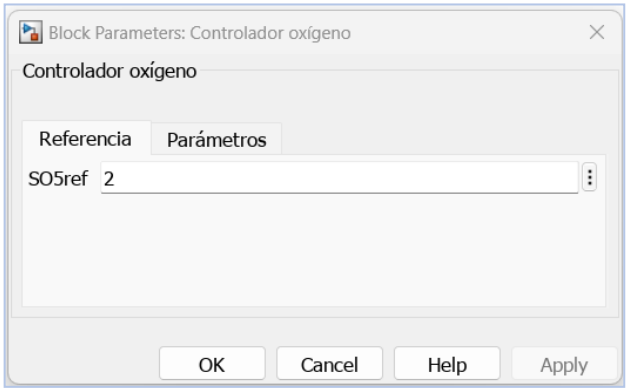


Figura 54 Pestaña de Parámetros del Controlador de Oxígeno

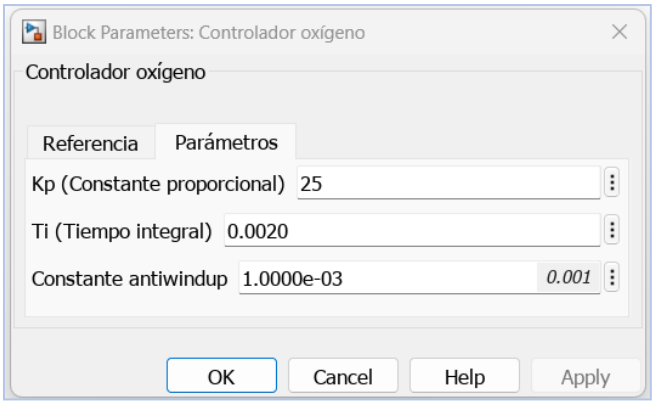


Figura 53 Pestaña de Parámetros del Controlador de Oxígeno

- Pestaña Controlador de Nitratos: Esta sería la pestaña del controlador de nitratos, el diseño es muy simular al controlador de oxígeno con las dos pestañas de Referencia y Parámetros. Con la diferencia que en este contralor tenemos una variable modificable más que es el Kfeedfordward que no teníamos en el otro controlador y que se refiere a la existencia de un controlador por prealimentación (feedforward), que trata de rechazar el efecto de las perturbaciones antes de que afecten al sistema.

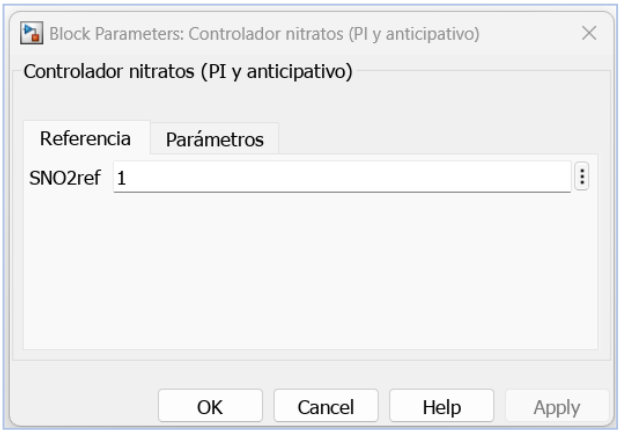


Figura 56 Pestaña de Parámetros del Controlador de Nitratos

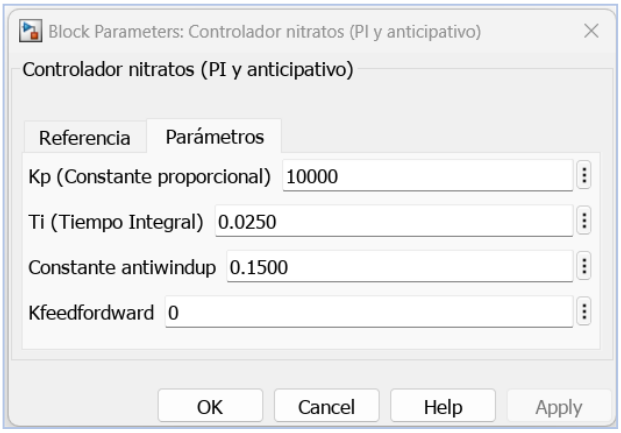


Figura 55 Pestaña de Parámetros del Controlador de Nitratos

- Pestaña del Decantador Secundario: Por último tendríamos la pestaña del Decantado Secundario con sus dos parámetros modificables: caudal de entrada a la planta (influente) y el caudal de purga de fangos.

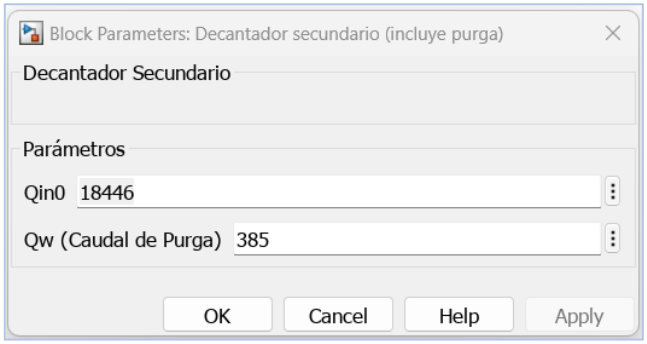


Figura 57 Pestaña de Parámetros del Decantador Secundario

7. Conclusiones y líneas de trabajo futuras

Después de haber terminado la elaboración del proyecto puedo decir que se han cumplido satisfactoriamente los principales objetivos marcados al principio del proyecto, la elaboración final de la aplicación ha sido un éxito y cumple con esas funcionalidades.
Ofrece al alumno hacer las simulaciones de diferentes modelos, desde los más básicos hasta los más complejos, para un tiempo de simulación que decida el usuario, cambiar los valores de diferentes variables para ver su evolución después de la simulación, visualizar los resultados en gráficas incorporadas en la propia ventana de la aplicación, cargar simulaciones realizadas anteriormente, guardar la simulación realizada en el momento, comparar en la gráfica seleccionada las diferentes simulaciones que haya escogido, mostrar los informes de la planta y el controlador en la propia ventana del modelo más complejo (Depuradora de Agua), entre otras funcionalidades.
En resumen, es una aplicación bastante completa, intuitiva y fácil de utilizar que permite al alumno con pocos conocimientos del sector industrial de los sistemas automáticos, familiarizarse más con estos conceptos e interactuar con los diferentes modelos que se plantearon.

Otro aspecto a destacar es la herramienta utilizada para la implementación del software, pues acercar más a los alumnos a otro tipo de herramientas también es muy importante, ya que, aunque en los sectores de la programación no se suele tener en cuenta, MATLAB, es una aplicación que tiene muchas funcionalidades y proporciona muchas herramientas y es gracias a una de ellas, App Designer, se ha podido desarrollar la aplicación de Simulación de Modelos.

Algunas líneas de trabajo futuro pueden ser: la visualización dinámica del propio sistema cuando se ejecuta una simulación, un ejemplo del propio proyecto sería la capacidad de ver los tanques acoplados llenándose de agua al mismo instante que se está ejecutando la simulación, evidentemente la complejidad de añadir esta funcionalidad a cada modelo es muy alta y tendría sentido en un futuro proyecto independiente.
La aplicación está diseñada de forma que puedan añadirse diferentes modelos a la base desarrollada y no comprometer el diseño arquitectónico

en su conjunto, esto es otro plan de futuro que se podría discutir, para ofrecer más modelos con los que experimentar y familiarizarse.

Comentar que el desarrollo de este proyecto ha sido muy positivo para mí, primero, en cuanto a experiencia pues se trataba del primer proyecto a largo plazo que debía desarrollar, y segundo, en lo que a consistencia y trabajo diario se refiere, pues al ser un trabajo a largo plazo, de nada sirve intentar realizarlo en un período corto de tiempo.

Evidentemente el progreso del proyecto me ha llevado por diferentes fases de estados de ánimo a medida que iba evolucionando y avanzando, fases en las que me atascaba y no sabía por dónde dirigir el proyecto, otras fases en las que todo iba como la seda, etc. Pero, en términos generales, se me ha hecho muy ameno el desarrollo del proyecto además sobre un tema que me llama mucho la atención como son los sistemas de control.

Finalmente, agradecer la ayuda de mis tutores, al facilitarme el desarrollo del proyecto con un feedback constante en cada una de las reuniones que hemos tenido, esto es muy importante porque en los momentos de indecisión sobre algunos aspectos del proyecto tenía la posibilidad de pedir consejo a mis tutores

8. Referencias

1. Matlab, S. (2012). Matlab. The MathWorks, Natick, MA. Acceso: https://www.mathworks.com/help/matlab/
2. Xue, D., & Chen, Y. (2013). System simulation techniques with MATLAB and Simulink. John Wiley & Sons.
3. Rodríguez, M. G. (2003). Introducción rápida a Matlab y Simulink para ciencia e ingeniería. Ediciones Díaz de Santos.
4. Cellier, F. E., & Kofman, E. (2006). Continuous system simulation. Springer Science & Business Media.
5. Stuetz, R. M., & Stephenson, T. (Eds.). (2009). Principles of water and wastewater treatment processes. Iwa Publishing.
6. Qasim, S. R. (2017). Wastewater treatment plants: planning, design, and operation. Routledge.
7. Visual Paradigm (s.f). Acceso: https://www.visual-paradigm.com/
8. Microsoft Proyect (s.f). Acceso: https://www.microsoft.com/es-es/microsoft-365/project/project-management-software