

## Flujo de Trabajo MBD

**Objetivo del Proyecto:** El propósito de esta práctica es validar las competencias en el flujo de trabajo de Diseño Basado en Modelos (MBD). Para aislar la complejidad matemática y centrarse en el manejo de Simulink, el sistema físico se limitará a un acumulador básico y su control se realizará mediante un simple Controlador Proporcional (P).

Todo el trabajo debe realizarse en un proyecto de Matlab que esté alojado en github.

### 1. Definición del Gemelo Digital (La Planta)

El sistema a modelar representa la dinámica de **Nivel de un Tanque de Agua** muy simplificado. El caudal que entra se acumula a lo largo del tiempo para generar un nivel.

- **Requerimientos:** Implementar la planta matemática utilizando un bloque integrador (Integrator).
- **Procedimiento:** La entrada del integrador será el *Caudal de llenado* (proveniente del controlador) y su salida será el *Nivel actual* del tanque.

### 2. Modelado de Condiciones No Ideales y Perturbaciones

Para evaluar la robustez del sistema, se deben introducir imperfecciones propias de los sensores y actuadores físicos.

- **Requerimientos:** Alterar el modelo ideal añadiendo los siguientes componentes en el lazo de control:
  1. **Ruido de medición:** Insertar un bloque de ruido (*Band-Limited White Noise*) sumado a la señal del *Nivel actual*, simulando la imprecisión del sensor ultrasónico.
  2. **Retardo de actuación:** Insertar un bloque de retardo (*Transport Delay*) en la señal de *Caudal de llenado* antes de entrar a la planta, simulando el retardo mecánico de la bomba de agua.
  3. **Perturbaciones:** Utilizar un bloque escalón (*Step*) acoplado a un bloque de resta justo antes del integrador, simulando la aparición de una fuga repentina en el tanque que extrae caudal del sistema.

### 3. Encapsulamiento del Controlador y Configuración de Interfaz (Masking)

Se requiere diseñar un sistema de control de lazo cerrado que regule el nivel del tanque utilizando un control Proporcional (P) puro.

- **Requerimientos:** El algoritmo debe calcular el error (Referencia - Nivel Medido) y multiplicarlo por una ganancia.
- **Procedimiento:**
  1. Construir la lógica matemática utilizando un bloque de suma (Sum configurado como  $+$ -) y un bloque de ganancia (Gain).
  2. Agrupar exclusivamente estos dos bloques en un subsistema (*Subsystem*).

3. Crear una máscara (*Mask*) sobre dicho subsistema.

4. Configurar la interfaz gráfica de la máscara para que exponga un único parámetro editable por el usuario: la Ganancia Proporcional ( $K_p$ ).

#### 4. Generación de Código C++

El bloque del controlador Proporcional debe ser pasado a C++ y verificado.

- **Requerimientos:** Traducir el subsistema del controlador a código C/C++ y validar su salida estricta.
- **Procedimiento:** Utilizar la herramienta *Simulink Coder* configurando el *System Target File*. Mediante el *Simulation Data Inspector*, superponer la señal de salida de control del modelo nativo frente a la generada por el código C++ y demostrar su equivalencia.

#### 5. Protección de la Propiedad Intelectual del Modelo

El modelo final debe empaquetarse para ser entregado a una persona externa, quien evaluará su comportamiento, pero no tendrá acceso a la configuración técnica.

- **Requerimientos:** Exportar el sistema completo (Controlador P + Planta + Perturbaciones) como un modelo protegido.
- **Procedimiento:** Generar un *Simulink Protected Model* (formato. slxp). El entregable final debe permitir al usuario externo modificar únicamente la señal de *Nivel Deseado* (Referencia) y visualizar el *Nivel actual* resultante en un osciloscopio (*Scope*). El sistema deberá denegar por diseño el acceso al interior de los subsistemas y la visualización de la ganancia  $K_p$ .