Trabajo Fin de Máster
Máster en Ingenería Electrónica, Robótica y Automática

Laboratorio Virtual Remoto para el Control de la Planta Multiprocesos

Autor: Víctor Manuel Gracia Villegas
Tutor: Daniel Limón Marruedo
Cotutor: David Muñoz de la Peña Sequedo

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Máster
Máster en Ingeniería Electrónica, Robótica y Automática

Laboratorio Virtual Remoto para el Control de la Planta Multiprocesos

Autor:
Víctor Manuel Gracia Villegas

Tutor:
Daniel Limón Marruedo
Catedrático

Cotutor:
David Muñoz de la Peña Sequedo
Catedrático

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Máster: Laboratorio Virtual Remoto para el Control de la Planta Multiprocesos

Autor: Víctor Manuel Gracia Villegas

Tutor: Daniel Limón Marruedo

Cotutor: David Muñoz de la Peña Sequedo

El tribunal nombrado para juzgar el Trabajo arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

A mi familia

A mi María

A mis maestros

Agradecimientos

Me gustaría dar las gracias a todas aquellas personas que han estado apoyándome durante esta etapa, sin las cuales este proyecto no habría sido posible.

Especialmente a Daniel Limón, por estar siempre dispuesto a prestarme su ayuda amablemente sin importar las circunstancias ni el momento, así como demostrarme que con paciencia todo es posible. También a David Muñoz, por interesarse tanto por el proyecto y por su apoyo en el desarrollo del mismo.

Debo agradecer el apoyo a mi familia, en especial a María, por ser mi confidente y creer en mí. A mi madre, por ser la que siempre me ha impulsado a seguir adelante y considerarla además de mi madre, mi amiga. Tampoco puedo olvidarme de quien ya no está, espero que sigas mi camino desde donde quiera que me estés viendo, y que estés orgulloso de mí.

Víctor Manuel Gracia Villegas

Sevilla, 2021

Resumen

La planta multiprocesos, como su propio nombre indica, es un sistema en el cual se pueden llevar a cabo varios procesos a la vez, tales como el control de temperatura mediante resistencias térmicas e intercambiadores de calor, o el control de nivel. Se trata de un sistema muy atractivo desde el punto de vista académico, pues pone de manifiesto comportamientos propios de sistemas de control presentes en la industria actualmente, además de dar la posibilidad a los alumnos de poner en práctica técnicas de control multivariable, y trabajar con un sistema que presenta retardo en ciertos procesos.

El proyecto que aquí se describe pretende continuar el Trabajo de Fin de Grado "Laboratorio Virtual para el Control de la Planta Multiprocesos Basado en LabView". En él, se consiguió controlar el sistema presentado mediante el Software LabView, y se realizó la creación de una interfaz gráfica tanto para identificación como para el control del sistema.

En este trabajo, el objetivo es poder controlar el sistema de una forma similar mediante el navegador web de forma remota. Para ello, se hará uso de la herramienta Easy JavaScript Simulations (EJSS), un software desarrollado específicamente para la implementación de laboratorios remotos.

De forma previa a la conexión con LabView, y también al trabajo en EJSS, se ha desarrollado un modelo de parámetros distribuidos en Matlab para modelar el comportamiento del sistema. Este modelo consiste en un conjunto de ecuaciones diferenciales que son capaces de modelar el fenómeno de transporte de masa asociado a la medida de temperatura, así como su comportamiento dinámico. Por otro lado, también modela el nivel del depósito ante los cambios en la válvula que determina el caudal de agua que circula por la planta.

Uno de los objetivos del trabajo es que los usuarios finales puedan trabajar de una forma totalmente local, es decir, que puedan simular y controlar el sistema solamente con EJSS, sin necesidad de LabView ni de la planta real. Por ello, una vez ajustados los parámetros del modelo en Matlab, éste se ha implementado en EJSS. Para validar su correcto funcionamiento, se han comparado los resultados obtenidos tanto en EJSS como en Matlab para comprobar que coinciden.

Una de las ventajas principales de EJSS es que permite conectarse de una forma relativamente sencilla con LabView mediante un protocolo de comunicación también desarrollado con el mismo fin. El protocolo "Remote Interoperability Protocol" permite comunicar variables de lectura y escritura entre LabView y EJSS.

Los programas desarrollados en el anterior proyecto en LabView serán configurados para ejercer como servidores en la conexión, mientras que EJSS será el cliente que se conecte al servidor para leer y escribir variables del mismo. En un primer momento, EJSS se ha conectado con el LabView basado en modelos, para comprobar que funciona de la forma esperada. Después, se realizará la conexión del EJSS con el LabView corriendo en la planta, completando así el objetivo de controlar la planta mediante EJSS de forma remota.

Abstract

The multiprocess plant, as its own name says, is a system in which several processes can occur at the same time, such as the temperature control by using termic resistances and heat exchangers, or the level control. It is a very attractive system from the academic point of view, as it shows typical behaviours from control systems present in the industry today, in addition to giving the possibility to the students to practice multivariable control techniques, and working with a system which shows delays in some processes.

The proyect described here pretends to continue the Final Degree Project "Laboratorio Virtual para el Control de la Planta Multiprocesos Basado en LabView". In it, it was achieved to control de system through LabView Software, and a graphic interface was made for identification as well as for control.

In this work, the objective is to be able to control the system in a similar way through the web browser remotely. For this, the tool Easy JavaScript Simulation (EJSS) will be used, which is a software specifically developed for the implementation of remote laboratories.

Previous to the conection with Labview, and also to the work in EJSS, a distributed parameters model has been developed in Matlab to model the behaviour of the system. This model consists of a set of diferential equations which are able to model the mass transportation phenomenon associated to the temperature measurement, as well as its dynamical behaviour. On the other hand, the level in the tank, when subject to movements in the valve which determines the water flow, is also modelled.

One of the objectives of the work is that end-users can work in a completely local mode, this is, that they can simulate and control the system with EJSS only, without the need of LabView or the real plant. For this reason, once the paremeters in Matlab have been chosen, the model has been implemented in EJSS. To validate its correct operation, the results in EJSS have been compared to the ones in Matlab to make sure they are the same.

One of the main advantages of EJSS is that it allows to be connected with LabView in a relatively easy way through a communication protocol also developed with the same aim. The protocol 'Remote Interoperability Protocol' allows to communicate read and write variables between LabView and EJSS.

The programs developed in the last proyect in LabView will be configured to run as servers in the connection, while EJSS will be the client which connects to the server to read and write variables. At first, EJSS has been connected to Labview in which the models are programmed, to make sure it works as expected. Later, the connection between EJSS and LabView running in the plant will be done, achieving this way the objective of controlling the system through EJSS remotely.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

Este proyecto ha sido realizado sobre la planta de control multiprocesos, disponible en el Laboratorio de Control, que pertenece al Departamento de Ingeniería de Sistemas y Automática de la Universidad de Sevilla.

Se recomienda que, para una mejor comprensión, se lea el Trabajo de Fin de Grado "Laboratorio Virtual para el control de la Planta Multiprocesos basado en LabView", el cual es el trabajo previo al aquí expuesto. No obstante, cierta información será redundante entre ambos documentos, ya que se pretende que este proyecto sea autocontenido.

1.1. Motivación para la realización del proyecto.

La planta multiprocesos es un sistema muy interesante para poner en práctica los conocimientos adquiridos sobre el control automático, pues posee características propias de sistemas implementados en la industria actualmente.

La principal motivación para la realización de este proyecto es la creación de un laboratorio remoto integrado en un programa de laboratorios remotos promovido por la CEA-IFAC. Esto permitirá la realización de prácticas remotas de plantas experimentales para el aprendizaje sobre sistemas de control.

De este modo, mejorará la accesibilidad de la planta al hacer posible que los alumnos del Departamento que necesiten realizar práticas en la misma prescindan de estar físicamente en el laboratorio.

1.2. Objetivos y alcance del proyecto.

Los objetivos de este trabajo son los siguientes:

- Desarrollar un modelo de parámetros distribuidos de la planta en Matlab, ajustando los parámetros de forma óptima para que sea fiel al sistema real.

- Implementar el modelo en el software Easy JavaScript Simulations (EJSS), y validar los resultados comparándolos con los obtenidos en Matlab.

- Crear una interfaz gráfica con vista en formato HTML (navegador web) que permita el manejo de los modelos y del sistema real de forma sencilla e intuitiva, además de la visualización y obtención de datos experimentales.

- Diseñar un programa con funcionamiento totalmente local en EJSS. Esto supone una gran ventaja para los alumnos al poder utilizar un programa idéntico al que manejarán en el laboratorio. Ello implica que los alumnos podrán familiarizarse con la interfaz gráfica, además de realizar los mismos ensayos que se podrán realizar sobre la planta, pero en este caso, sobre los modelos. Serán preferiblemente de forma previa a su asistencia al laboratorio, de modo que se agilizará la adaptación de los usuarios al sistema real.

  Además, su uso será tan sencillo como instalar y ejecutar el programa de EJSS en el ordenador. No será necesario LabView, ni la planta, ni ningún conocimiento previo.

- Modificar los programas en LabView creados en el Trabajo de Fin de Grado (TFG) para adaptarlos a la conexión con EJSS.

- Conectar el programa en LabView basado en modelos a EJSS mediante el protocolo de comunicación Remote Interoperability Protocol (RIP), donde LabView será el servidor y EJSS el cliente.

- Validar el correcto funcionamiento del programa en EJSS cuando está conectado a LabView.

- Conectar el programa en LabView basado en la planta real a EJSS mediante RIP, validando que la planta se controla correctamente desde el navegador web (EJSS).

1.3. Descripción de la planta.

La planta multiprocesos es un equipo de laboratorio orientado a la enseñanza de sistemas de control de procesos de variables fundamentales como nivel, caudal, temperatura o presión. Está basada en una planta desarrollada por la empresa GUNT, pero ampliada y modificada en el Departamento para mejorar su funcionalidad (véase la Figura 1.1). La instrumentación de la planta en de tipo industrial y el sistema de control está basado en un PLC Schneider M340 emulando un sistema de control realista. Además, cuenta con una pantalla HMI Magelis para su operación local, y permite su control remoto desde un PC.

Figura 1.1 – Fotografía de la planta multiprocesos.

1.3.1. Esquema, variables y configuraciones.

El esquema de la planta se muestra en la Figura 1.2 en la configuración que se va a operar la planta. En este

esquema también se muestra la instrumentación instalada en la misma.

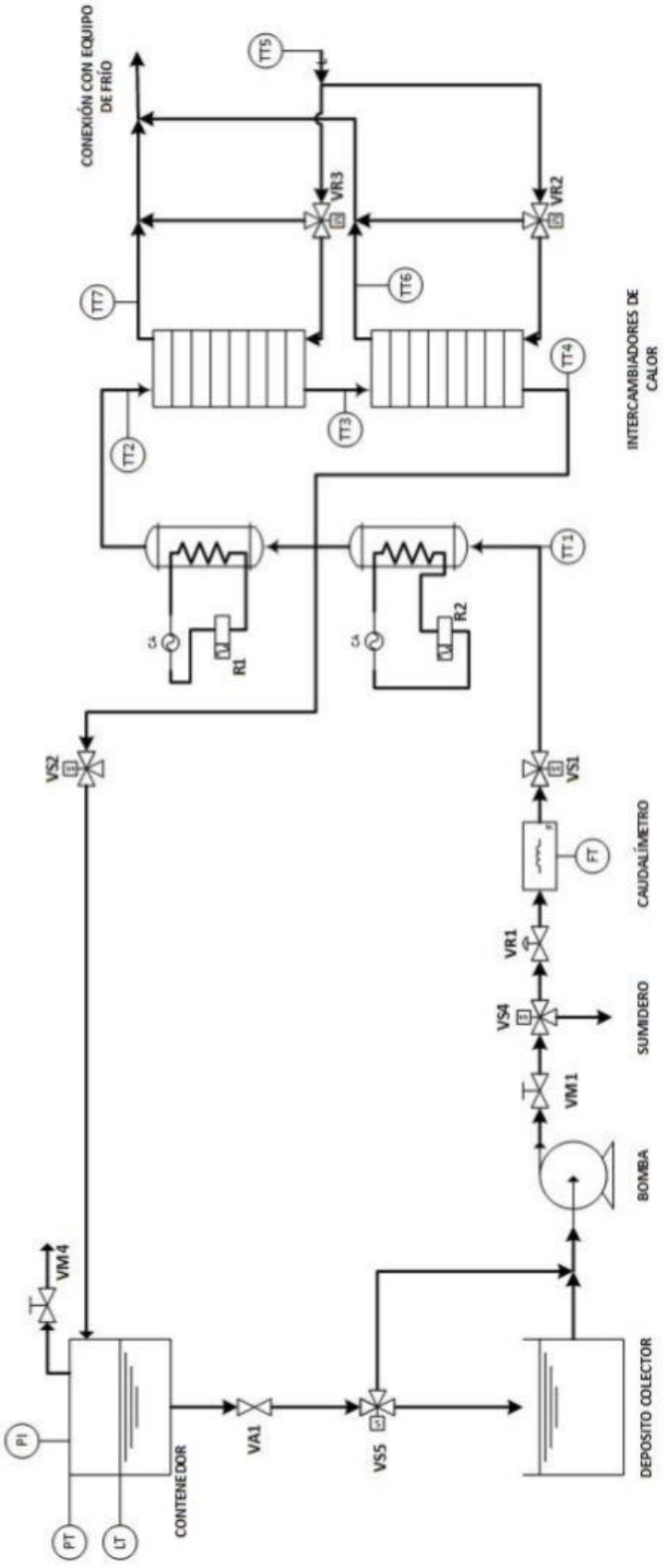


Figura 1.2 – Esquema de la planta multiprocesos y su instrumentación.

La relación de instrumentos instalados en la planta y su nomenclatura se detalla en la siguiente tabla:

FT	Caudalímetro
PT	Sensor de presión
LT	Sensor de nivel
TT1	Sensor de temperatura (PT100 Endress-Hauser)
TT2	Sensor de temperatura (PT100 Endress-Hauser)
TT3	Sensor de temperatura (PT100 Endress-Hauser)
TT4	Sensor de temperatura (Sonda CAREL)
TT5	Sensor de temperatura (Sonda CAREL)
TT6	Sensor de temperatura (Sonda CAREL)
TT7	Sensor de temperatura (Sonda CAREL)
TT8	Sensor de temperatura (PT100 Endress-Hauser)
TT9	Sensor de temperatura (PT100 Endress-Hauser compartida)
VM1	Válvula manual de la bomba
VM2	Válvula manual de evacuación de agua
VM3	Válvula de entrada al rotámetro
VM4	Válvula de cierre del depósito contenedor al ambiente
VSi	Válvulas de solenoide de configuración
VR1	Válvula de regulación (Samson neumática de asiento lineal)
VR2	Válvula eléctrica regulable (Johnson Control eléctrica 3 vías)
VR3	Válvula eléctrica regulable (Johnson Control eléctrica 3 vías)
R1	Resistencia de 2 kW
R2	Resistencia de 4 kW

Tabla 1.1 – Relación de instrumentos instalados en la planta y nomenclatura.

La planta tiene como posibilidad tres configuraciones para su operación. Esto da lugar a distintas estrategias de control posibles.

Para el presente proyecto, y al igual que en el anterior TFG, nos centraremos en la segunda configuración. Ésta consiste en lo siguiente:

- Configuración 2 (C2): El sistema permite controlar tanto el nivel del tanque (LT) como la temperatura del agua que retorna al depósito (TT2). Para este último fin, el agua circulará por las tuberías que conducen hacia las resistencias térmicas R1 y R2 para calentarla, y hacia los intercambiadores de calor para enfriarla mediante el intercambio de calor con el agua procedente de la planta de frío.

1.3.2. Descripción del funcionamiento de la planta.

La bomba aspira el agua del depósito colector y la impulsa a través de una válvula de regulación de asiento (VR1) pasando por un caudalímetro magnético (FT) hacia el resto de la planta donde se llevan a cabo los procesos que se pretenden controlar. El agua de salida del caudalímetro magnético FT circula a través de 2 resistencias eléctricas, R1 y R2, las cuales están dispuestas en serie, y cuya potencia eléctrica es manipulable mediante la modulación de anchura de pulso (PWM) de la tensión de cada resistencia. El agua calentada pasa a través del circuito primario de 2 intercambiadores de calor también en serie, donde se enfría el agua. Desde allí, el agua se conduce al tanque acumulándose, mientras se vacía por gravedad sobre el depósito colector completando un circuito cerrado.

El caudal de agua fría de entrada en el secundario de cada intercambiador se manipula por una válvula de 3 vías de regulación, VR2 y VR3, que divide el agua proveniente de una fuente de agua fría entre una tubería de bypass al intercambiador, y la entrada al secundario del intercambiador. La fuente de agua fría es una enfriadora de agua, que enfría el agua de un depósito desde el cual se impulsa a caudal constante hacia los intercambiadores.

Toda la información técnica necesaria sobre la planta se puede encontrar en la Guía de Usuario de la planta que complementa este documento.

1.4. Modo de operación de la planta.

La planta debe trabajar de la misma forma en la que lo hacía en el anterior TFG, pues es lógico al ser este proyecto una continuación del mismo.

Tras un extenso análisis del comportamiento de la planta, se llegó a la conclusión de que lo más conveniente era trabajar en el siguiente punto de operación:

- Punto de operación de la válvula de caudal VR1: 30%.
- Nivel (LT) en el punto de operación: 24.5cm.
- Posición de la válvula de descarga VA1: 1.9.
- Punto de operación del Duty Cycle de las resistencias R1 y R2: 55%.
- Salto de temperatura (TT2-TT1) en el punto de operación: 6.42ºC.

Cabe destacar que el control del lazo auxiliar se realizó mediante una estructura FeedForward. Se controla la temperatura del agua que retorna al depósito mediante la apertura de las válvulas VR2 y VR3, que determinan la cantidad de agua procedente de la planta de frío que entra a los intercambiadores de calor. Es interesante mencionar que la referencia de temperatura para este contralador es la temperatura ambiente del laboratorio al comenzar los experimentos, de modo que el calor aportado por las resistencias se resta antes de llegar al depósito mediante los intercambiadores de calor.

Por otro lado, y como también se comentó en el anterior trabajo, la temperatura controlada será el salto de temperatura entre TT2 y TT1. De esta forma, al ser una temperatura relativa, se evita al usuario percibir las oscilaciones producidas por el control por relé de la planta de frío que sí se pueden apreciar en las temperaturas absolutas TT1 y TT2.

1.5. Control previo de la planta mediante LabView.

Este apartado tiene como objetivo hacer un resumen del alcance que tuvo el TFG, y proyectar el rumbo que tomará el presente trabajo.

A modo de recordatorio, la planta se conectó a LabView siguiendo el siguiente esquema:

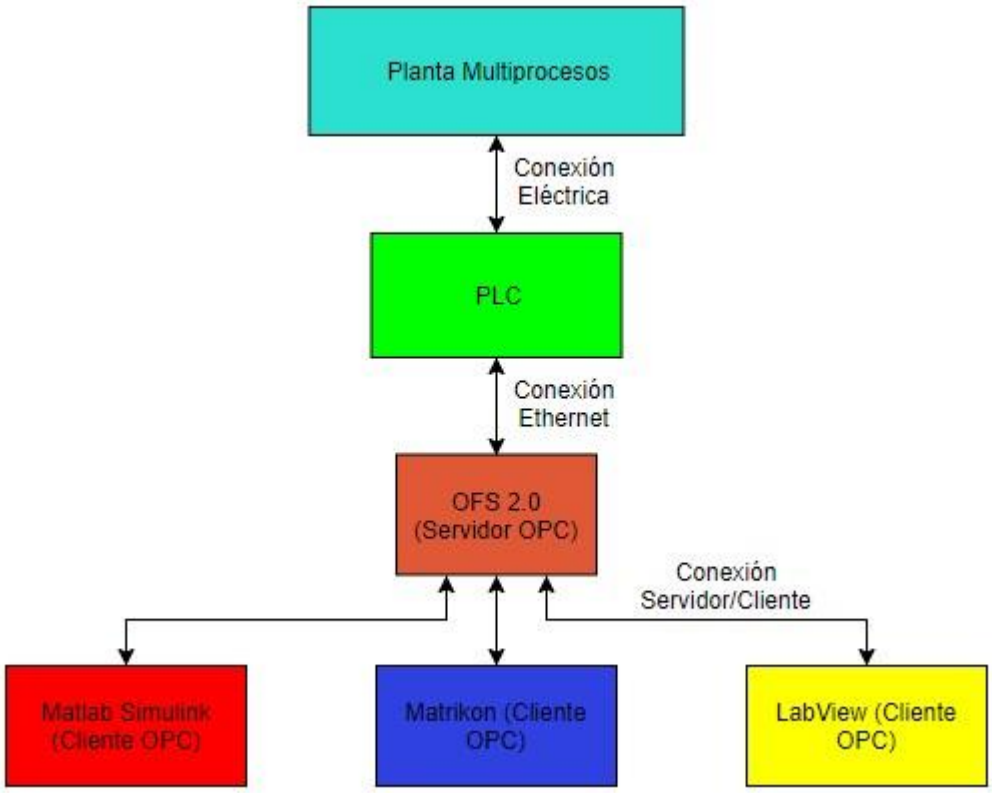


Figura 1.3 – Esquema de jerarquía de conexiones.

Como se puede apreciar en la Figura 1.3, nos conectamos a la planta desde LabView como clientes de la conexión OPC.

1.5.1. Programa en LabView: Control Simulado.

Tras la identificación que se llevó a cabo sobre la planta, se llamó "Control Simulado" a la implementación en LabView de las herramientas de identificación y control sobre los modelos también programados en este entorno.

Este programa no se conecta con el sistema real, sino que simula modelos de primer orden basados en la planta (uno de ellos con retardo). Esto permite al usuario lo siguiente:

- Modo Identificación: Realizar ensayos de identificación mediante entradas en escalón y señales Chirp.
- Modo Control: Controlar de forma descentralizada mediante PIDs con filtro de referencias, modelos sencillos de primer orden (uno de ellos con retardo) basados en la planta. Además, se permite alternar entre los modos Manual/Automático de los controladores.
- Visualizar gráficas en tiempo real con los valores de las entradas, salidas y referencias.

- Guardar los datos de los ensayos en un archivo que permite visuarlizarlos en Matlab para su análisis.

A continuación, se muestra una imagen de la interfaz gráfica en modo identificación:

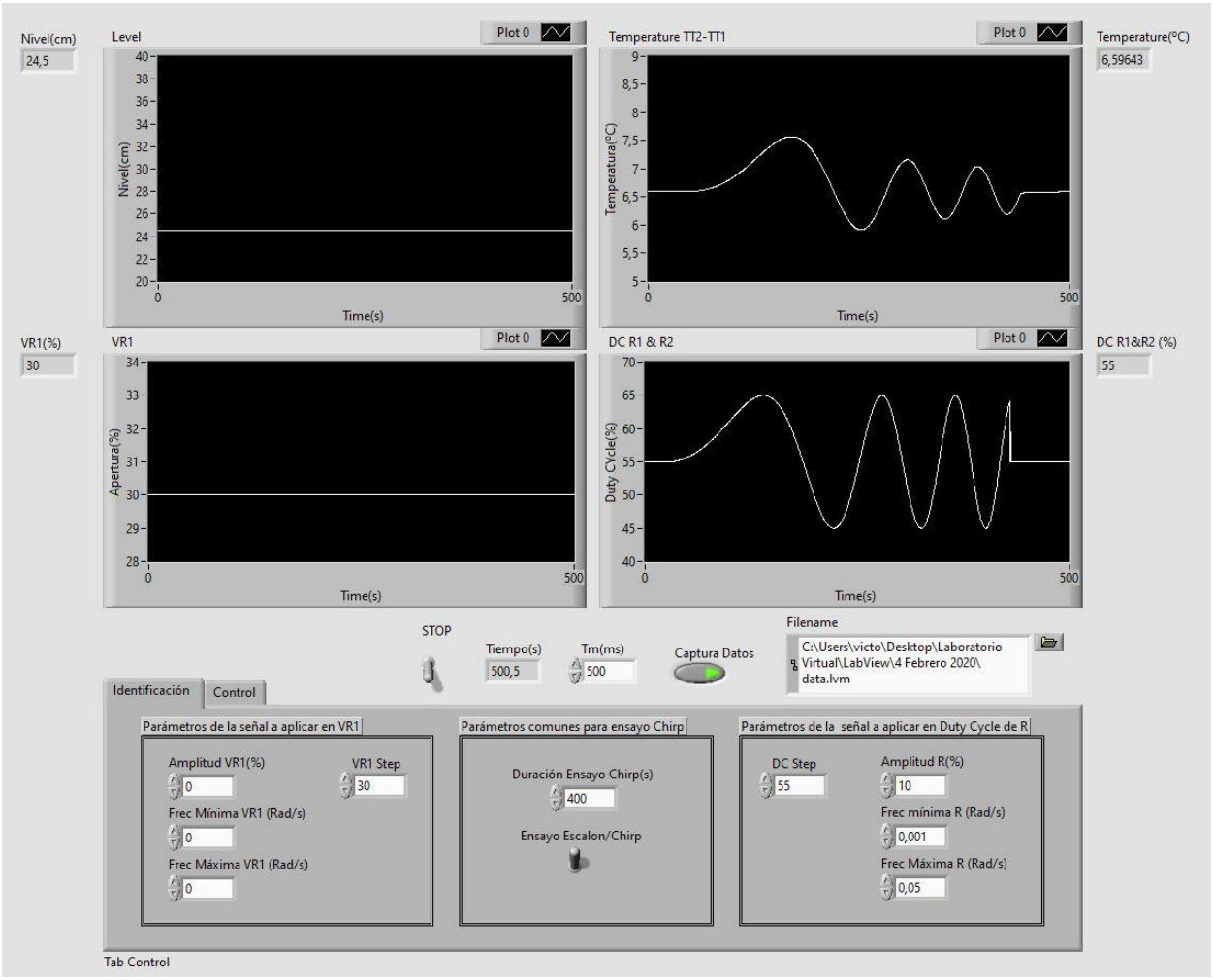


Figura 1.4 – Interfaz gráfica en LabView: Control Simulado en modo identificación.

Por otro lado, también se muestra el modo control:

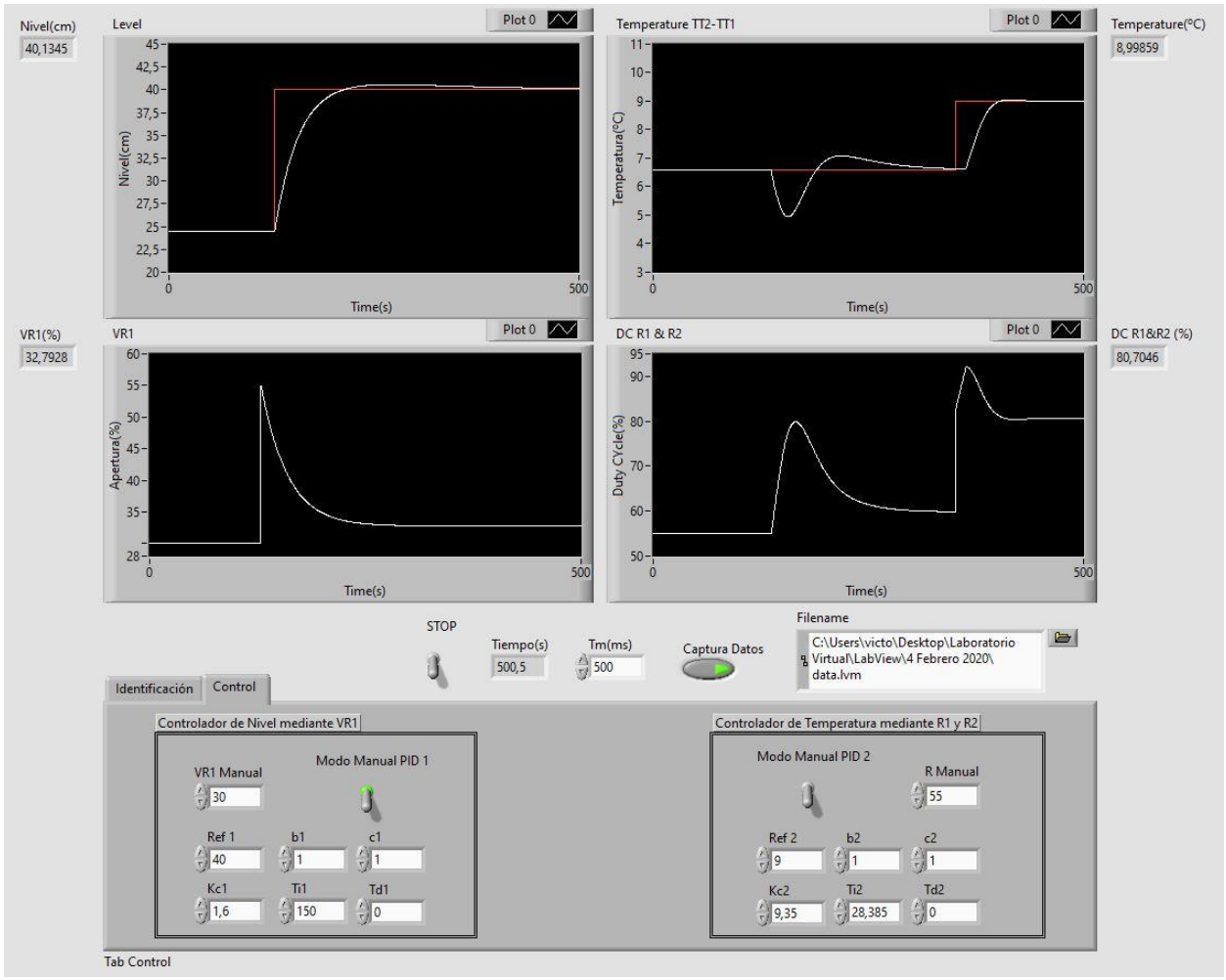


Figura 1.5 – Interfaz gráfica en LabView: Control Simulado en modo control.

Se puede comprobar en las figuras 1.4 y 1.5 que las salidas presentan comportamientos acordes a modelos de primer orden.

1.5.2. Programa en LabView: Control sobre Planta Real.

A diferencia del programa Control Simulado, el programa "Control sobre Planta Real" sí se conecta con el sistema real.

En cuanto a la interfaz gráfica, esta fue programada con la idea que se parezca mucho a la anterior, como se puede ver en las figuras 1.6 y 1.7:

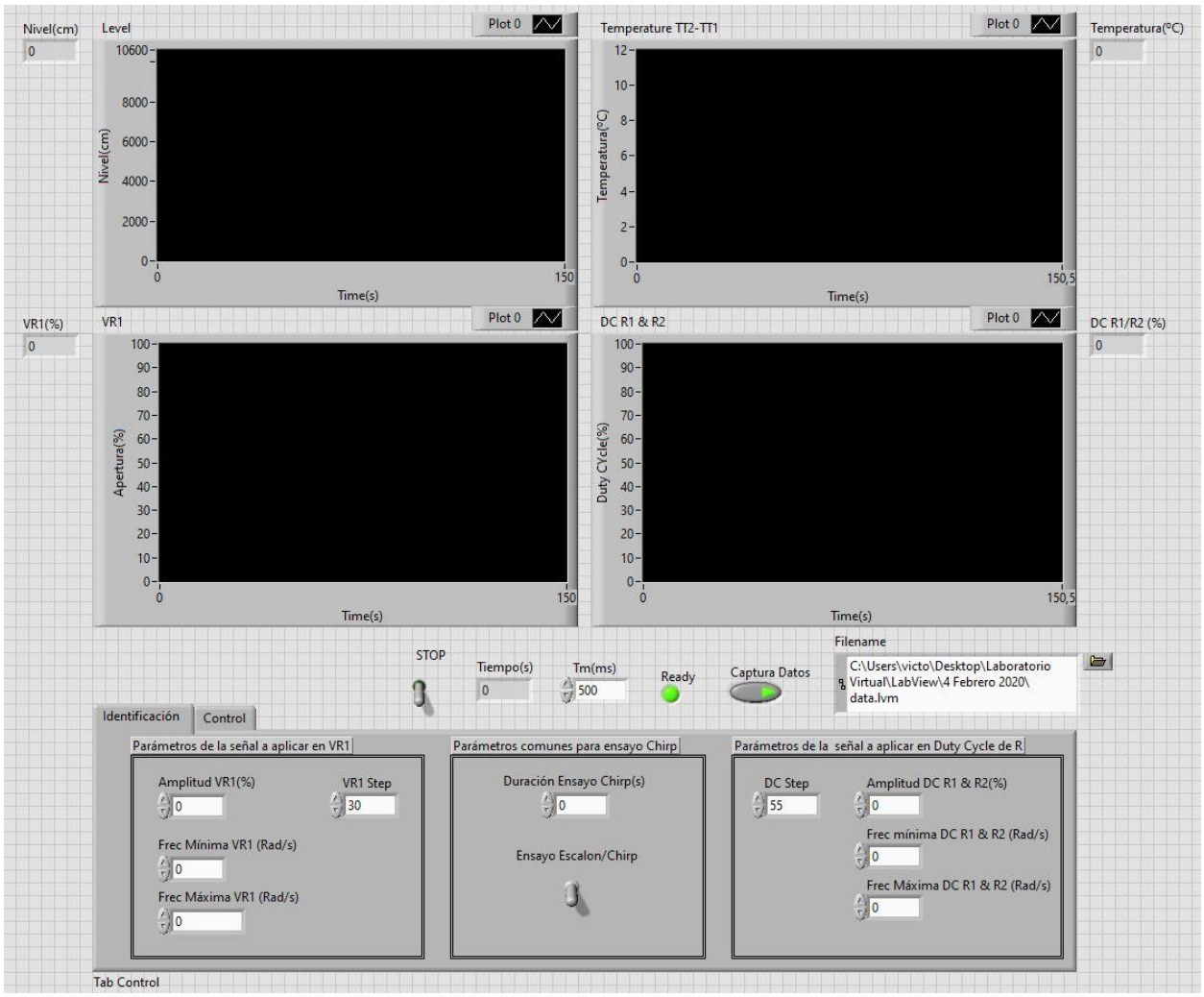


Figura 1.6 – Interfaz gráfica en LabView: Control sobre Planta Real en modo identificación.

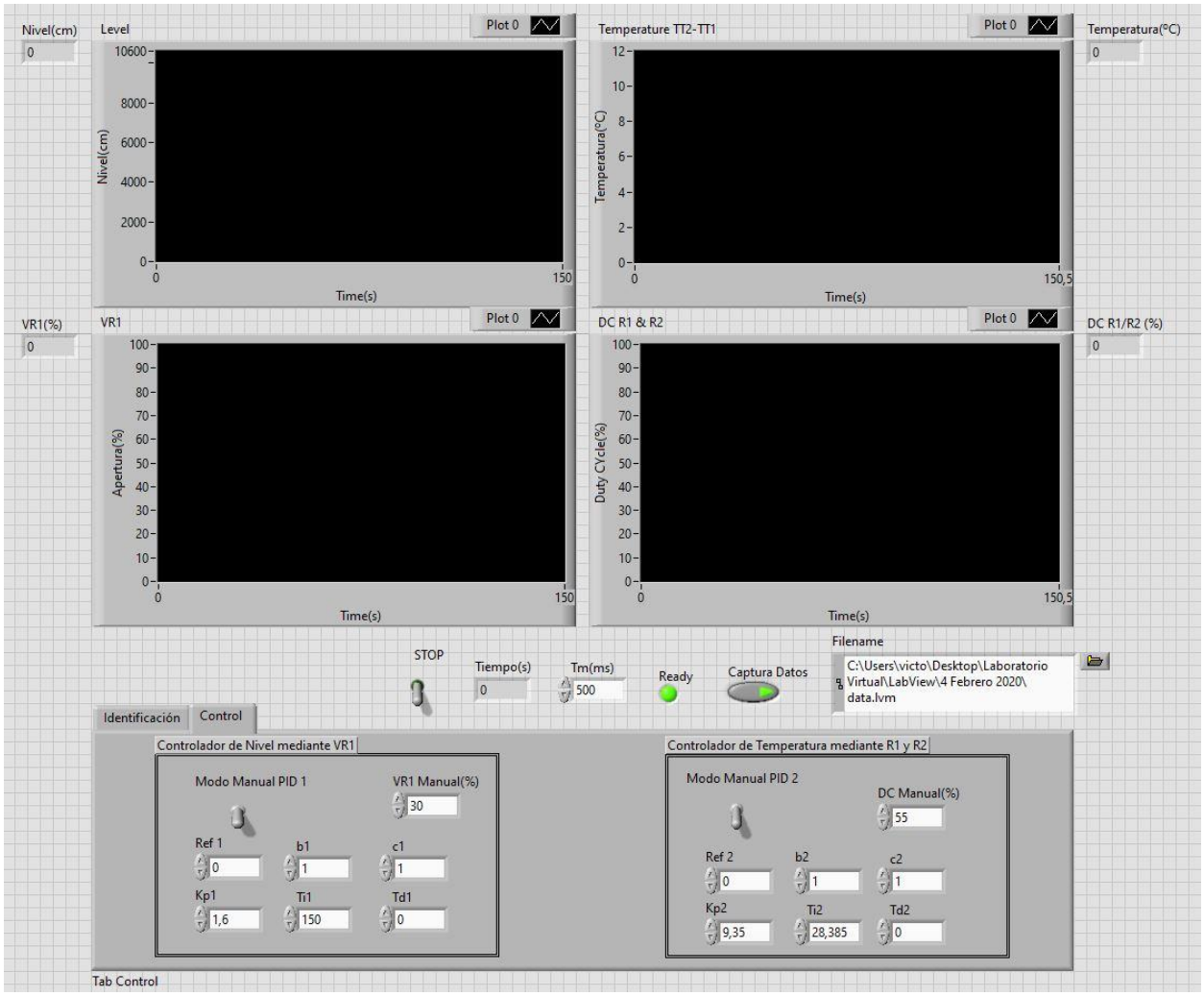


Figura 1.7 – Interfaz gráfica en LabView: Control sobre Planta Real en modo control.

Con el programa Control sobre Planta Real se consiguió realizar ensayos de identificación y control sobre la planta real a través del cliente LabView.

Como veremos más adelante, LabView será el servidor para la conexión con EJSS mediante el protocolo RIP. De este modo se pretende, en última instancia, controlar la planta real mediante EJSS.

1.6. Especificaciones del Laboratorio Virtual.

En este apartado nos centraremos en los objetivos del trabajo de cara a su implementación como práticas de laboratorio para el alumnado. Las posibilidades que ofrecerá se mencionan a continuación:

- Manejar la interfaz de forma sencilla desde el navegador Web, ya sea con el programa local basado en modelos en EJSS, o conectados a la planta real mediante las conexiones oportunas.
- Realizar ensayos de identificación:
  - Ensayo en escalón: El usuario podrá introducir escalones cuando lo desee en cualquiera de las dos entradas (VR1 y R).
  - Ensayo con señal Chirp a la entrada: Se podrán especificar las características de la señal Chirp a introducir en cualquiera de las dos entradas. Estas características son: Amplitud, frecuencias mínima y máxima (en radianes por segundo), y duración (en segundos).
- Realizar ensayos de control mediante controlador PID con Anti-Windup opcional:

- De forma predeterminada, se podrá controlar de forma descentralizada la temperatura mediante R, y el nivel mediante VR1 al ser un sistema triangular.
- El usuario podrá elegir el modo en el que trabajarán los controladores, ya sea en modo manual o en modo automático. En modo manual, se puede introducir la entrada según se desee. En modo automático, los controladores deberán estar parametrizados previamente para poder realizar el control.
- Los parámetros del controlador a introducir serán:
  - Ganancia del controlador (Kc).
  - Tiempo integral (Ti).
  - Tiempo derivativo (Td).
  - Término b del filtro de referencias.
  - Término c del filtro de referencias.
  - Anti-Windup o no.
- Se dará la opción de guardar los datos para que puedan ser mostrados y procesados en Matlab, por ejemplo.

Como se ha comentado previamente, la idea es que los usuarios dispongan de un programa local en EJSS, que no necesite de LabView ni de otro programa que no sea EJSS. Éste permitirá al usuario conocer la interfaz gráfica y un comportamiento aproximado del sistema real antes de acudir al laboratorio. Además, permite que todos los alumnos puedan trabajar con los modelos de forma independiente, teniendo que turnarse solamente para el sistema real.

De este modo, se busca agilizar el proceso del uso de la planta y disminuir el tiempo de espera entre alumnos

2 MODELADO DE LA PLANTA: MODELO DE PARÁMETROS DISTRIBUIDOS

2.1. Análisis del modelo propuesto.

2.1.1 Parte del sistema a modelar.

Como se ha comentado anteriormente, el usuario no tendrá que preocuparse del lazo auxiliar de control, pues es transparente a quien usa el programa. Por ello, nos limitaremos a implementar en EJSS los siguientes modelos:

- Modelo de nivel antes cambios en VR1.
- Modelo de salto de temperatura ante cambios en el Duty Cycle de R1 y R2.

Puesto que el usuario final trabajará con los modelos implementados en EJSS y no sobre los de primer orden de LabView, se han desarrollado unos modelos más sofisticados de parámetros distribuidos que se implementarán en Matlab, y, una vez ajustados, en EJSS.

2.1.2 Elección y justificación de ecuaciones diferenciales.

Al ser un modelo de parámetros distribuidos, es sencillo modelar fenómenos físicos como el transporte de masa, el cual induce el retardo observable en la temperatura. El modelo tendrá tantas ecuaciones diferenciales como secciones tenga la resistencia térmica, más una ecuación diferencial asociada al nivel del tanque.

2.1.2.1 Ecuación diferencial del modelo de nivel.

La ecuación diferencial asociada al nivel tiene la siguiente forma:

$$\frac{dh}{dt} = \frac{1000}{A} \cdot (q - K \cdot h^{\alpha}) \qquad (2.1)$$

, donde:

- h: Nivel del tanque [cm].
- q: Caudal a la apertura dada [l/h].
- A: Área transversal del tanque de nivel [cm^2].
- K y α: Coeficientes de descarga a ajustar.

Se han introducido dos coeficientes, K y α, puesto que introducen más grados de libertad que permiten un mejor ajuste del modelo de nivel.

2.1.2.2 Ecuación diferencial del modelo de temperatura.

La transferencia de calor en la resistencia se describe mediante ecuaciones de parámetros distribuidos. Sin embargo, será modelada como una cantidad de N volúmenes de control finitos en serie, cada uno de los cuales está descrito por la ecuación diferencial de parámetros concentrados que se detalla a continuación.

La ecuación diferencial asociada a cada volumen de control, y, por ende, al modelo de temperatura, tiene la siguiente forma:

$$dx(j) = \frac{Q_j}{C_p \cdot V_j} - q \cdot \frac{x(j) - x(j-1)}{V_j} - K_p \cdot \frac{x(j) - T_a}{C_p \cdot V_j} \qquad (2.2)$$

, donde:

- x(j): Temperatura en la sección j [ºC].
- Qj: Potencia de la resistencia entre nº de volúmenes de control [KW].
- q: Caudal de agua a la apertura dada [l/h].
- Vj: Volumen en la sección j [l].
- Kp: Coeficiente de pérdidas asociadas a la temperatura ambiente.
- Cp: Calor específico del agua (4,19J/(g·ºC)).

El índice "j" representa a qué sección se hace referencia, de las N secciones totales. Se deben integrar las N ecuaciones, sin embargo, sólo ciertas secciones poseen potencia eléctrica.

El fundamento sobre el que se basa el modelo es la transferencia de calor desde una sección a la siguiente. Cada sección (excepto la primera) tiene la temperatura de la sección anterior como entrada, por lo que, al estar sometida a cierta potencia calorífica, sufre un aumento de temperatura sobre la temperatura que proviene de la sección anterior. Sin embargo, también sufren pérdidas por la temperatura ambiente que es, en general, menor. Así, el modelo en conjunto está formado por N sistemas de primer orden en serie, donde la entrada de la sección j es la temperatura de la sección j-1, y la salida de la sección j es la entrada de la sección j+1. Esto, además, permite modelar el retardo.

Si analizamos los sumandos de la ecuación diferencial, el primer sumando del segundo término de la ecuación diferencial hace referencia a la temperatura aportada por la resistencia térmica.

El segundo sumando, hace referencia a la temperatura que proviene de la sección anterior, es decir, en j-1.

El tercer sumando está asociado a las pérdidas de temperatura debidas a la temperatura ambiente.

Para comprender mejor el fenómeno que induce el retardo, se ilustra un esquema a continuación:

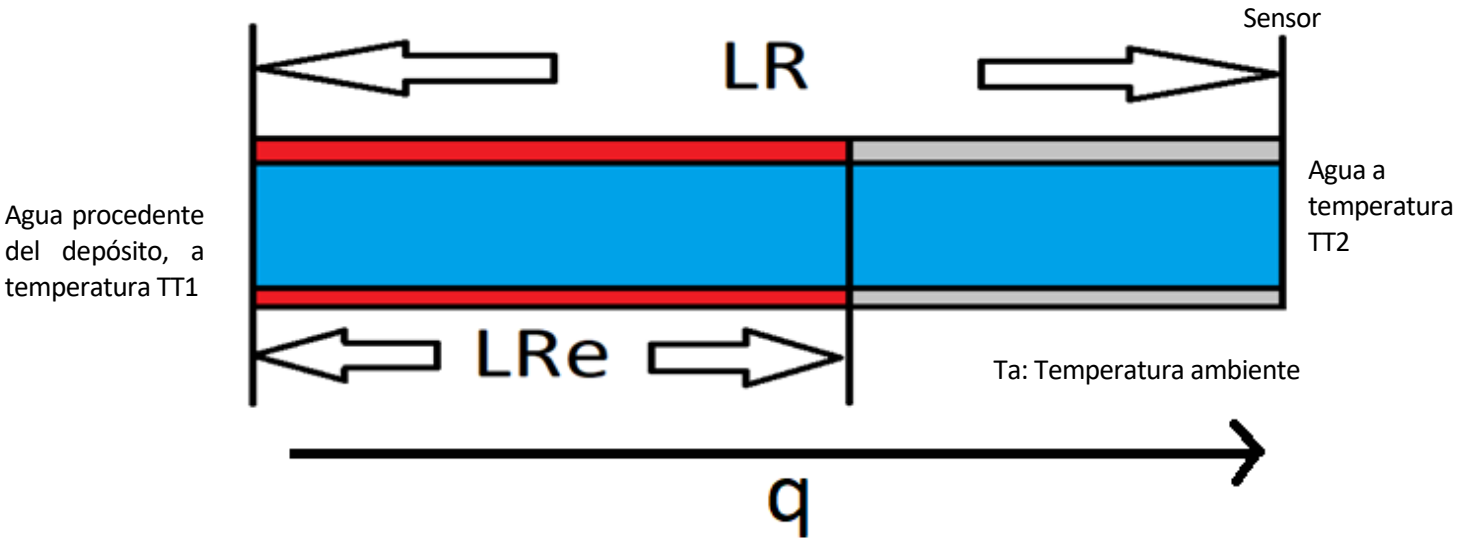


Figura 2.1 – Esquema del modelo de resistencia térmica.

En la figura 2.1, LR es la longitud total de la resistencia, y LRe es la longitud donde se concentran secciones

con potencia eléctrica de la resistencia. Evidentemente, LR-LRe está compuesto por secciones sin potencia eléctrica. Gracias a esta configuración, se consigue modelar el retardo inducido en la temperatura.

2.1.3 Modelado de la válvula VR1.

Como se puede observar en las ecuaciones (2.1) y (2.2), y en la figura 2.1, nos resulta de interés conocer el caudal asociado a la apertura de la válvula dada. Éste fue representado mediante la variable "q".

Para obtener la relación entre q y VR1, se analizó la característica estática de la válvula, midiendo el caudal para distintos porcentajes de apertura de VR1. Dicha característica estática se muestra en la figura 2.2.

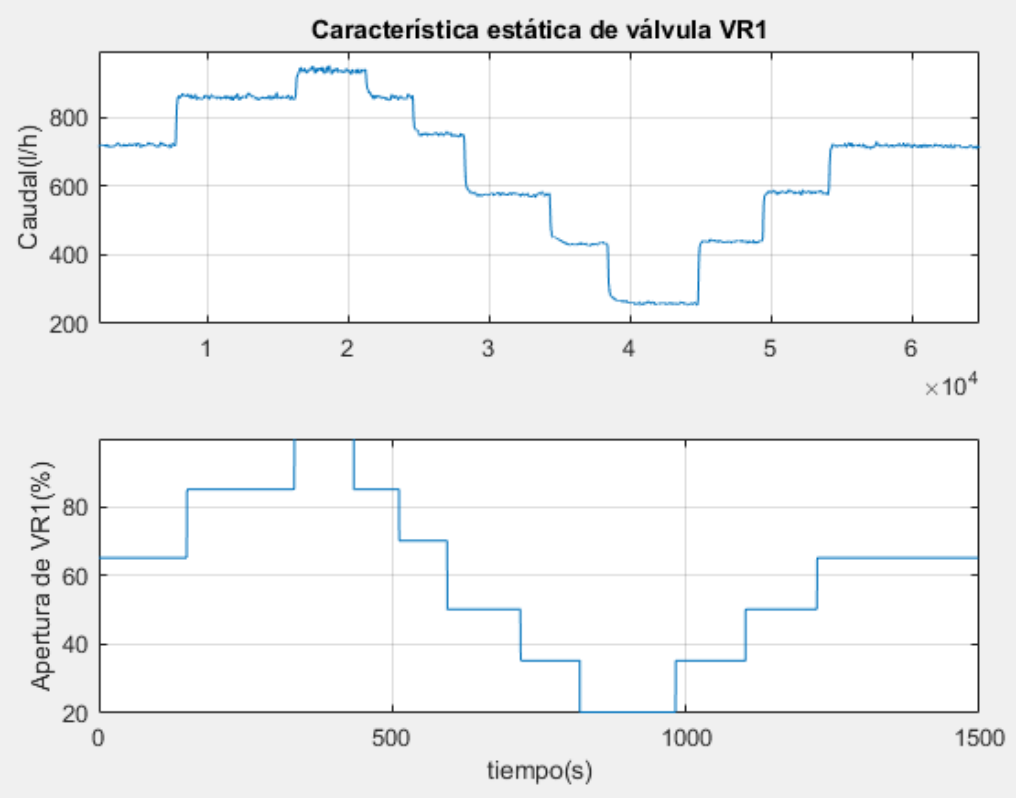


Figura 2.2 – Característica estática de la válvula VR1.

Analizando los valores del caudal para cada apertura, se han escogido los valores más relevantes. Si los representamos en una gráfica podemos observar cuán lineal es esta relación (figura 2.3).

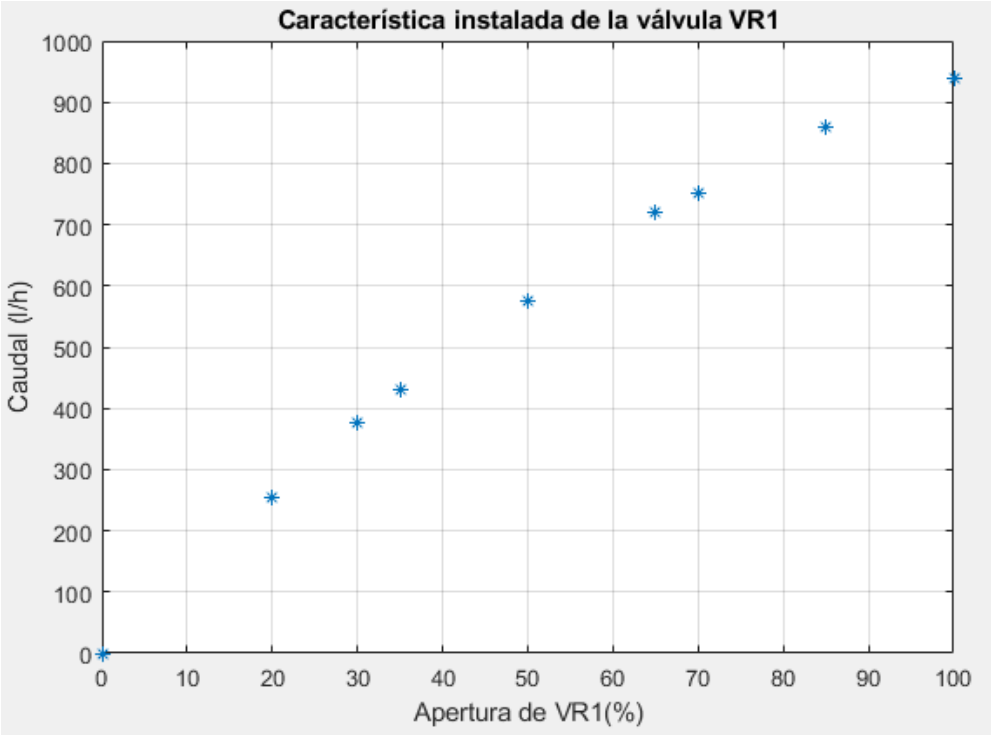


Figura 2.3 – Característica instalada de la válvula VR1.

A partir de la figura 2.3 se deduce que existe una relación caudal-apertura no demasiado lineal. Para obtener el caudal que representa la variable q se ha creado una función que realiza la interpolación lineal entre los puntos para obtener los valores del caudal valores de apertura intermedios. Además, ya que el caudal q se requiere en litros/segundo para las ecuaciones, los valores obenidos de la interpolación se dividen entre 3600.

Por otro lado, para simular el modelo, se han programado en Matlab varios scripts que contienen los parámetros y ecuaciones necesarios para la integración.

El comportamiento dinámico del modelo depende de los parámetros seleccionados, por lo que se deben ajustar para que el comportamiento del modelo sea lo más parecido posible al de la planta real.

Por otro lado, se debe tener en cuenta que es sumamente importante que el número de secciones en el que se divide la resistencia sea lo más pequeño posible, pues de ello depende el número de ecuaciones diferenciales a implementar. Sin embargo, esto no debe sacrificar que el modelo se parezca a la planta, por lo que habrá que encontrar un compromiso.

Los parámetros seleccionados se muestran en la siguiente tabla:

Parámetro	**Valor**
Sección de la resistencia térmica	136 cm^2
Potencia de la resistencia	5.3 KW
Longitud total del volumen de la resistencia	195 cm
Longitud total de la resistencia térmica	95 cm
Coeficiente global de pérdidas en la resistencia	0.002
Caudal máximo aportado por la bomba	940 l/h
Altura máxima del tanque	44 cm

Temperatura del depósito	19.5 ºC
Temperatura ambiente	22 ºC
Número de secciones de la resistencia	10
Coeficiente lineal de descarga de la válvula (K)	0.0211
Coeficiente exponencial de descarga de la válvula (α)	0.1682

Tabla 2.1 – Parámetros físicos del modelo y sus valores.

2.1.1. Análisis del comportamiento dinámico del modelo.

Con los parámetros de la tabla 2.1, se analizará la respuesta del modelo para los casos que se indican a continuación:

- Nivel ante cambios en VR1.
- Temperatura ante cambios en R con VR1 constante.
- Temperatura ante cambios en VR1 con R constante.

2.1.1.1. Modelo de nivel.

Comenzado por el primero de los tres casos, tenemos los siguientes resultados:

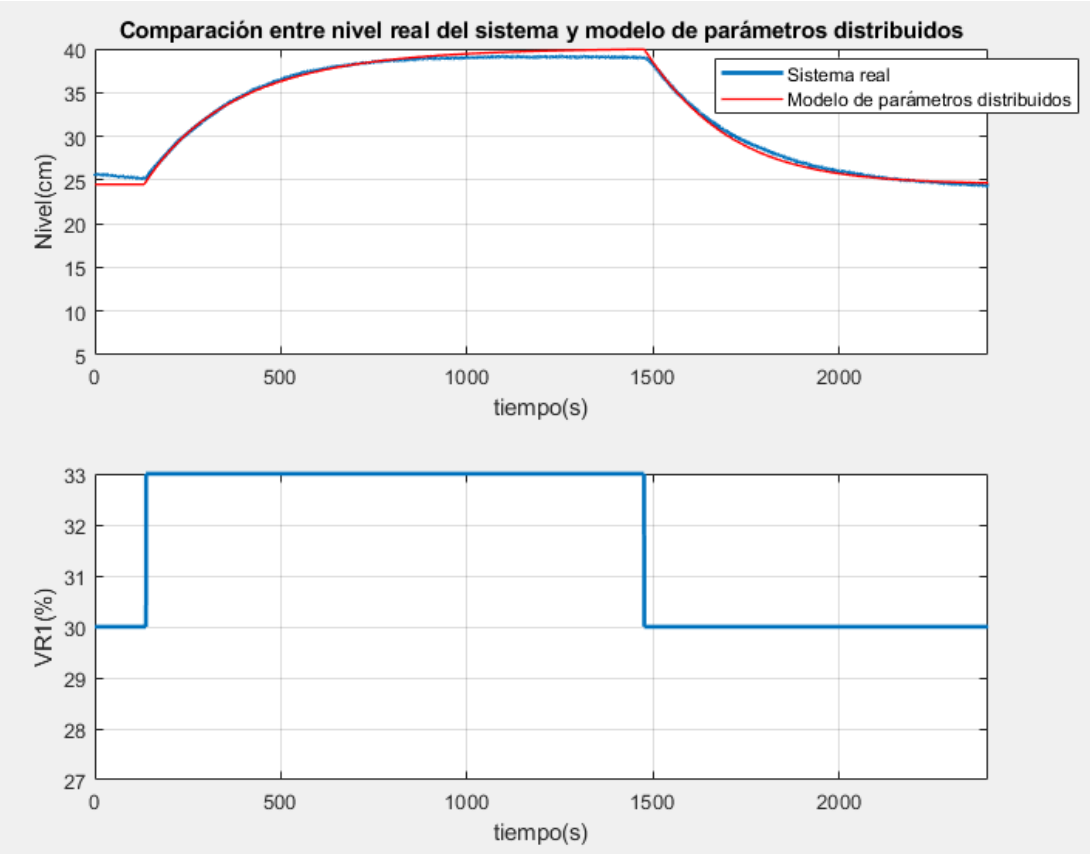


Figura 2.4 – Comparación entre nivel del sistema real y modelo de parámetros distribuidos.

La figura 2.4 muestra que el modelo se comporta de forma muy similar al sistema real en términos de nivel. Recordamos con este experimento que el nivel presenta la respuesta típica de un sistema de primer orden. Se consideran correctamente ajustadas tanto la ganancia estática como la constante de tiempo.

2.1.1.2. Modelo de temperatura ante cambios en R, con VR1=Cte.

Para el segundo caso, en el que se mantiene VR1 al 30% constante y se modifica R, se obtienen los resultados que muestran la siguiente figura:

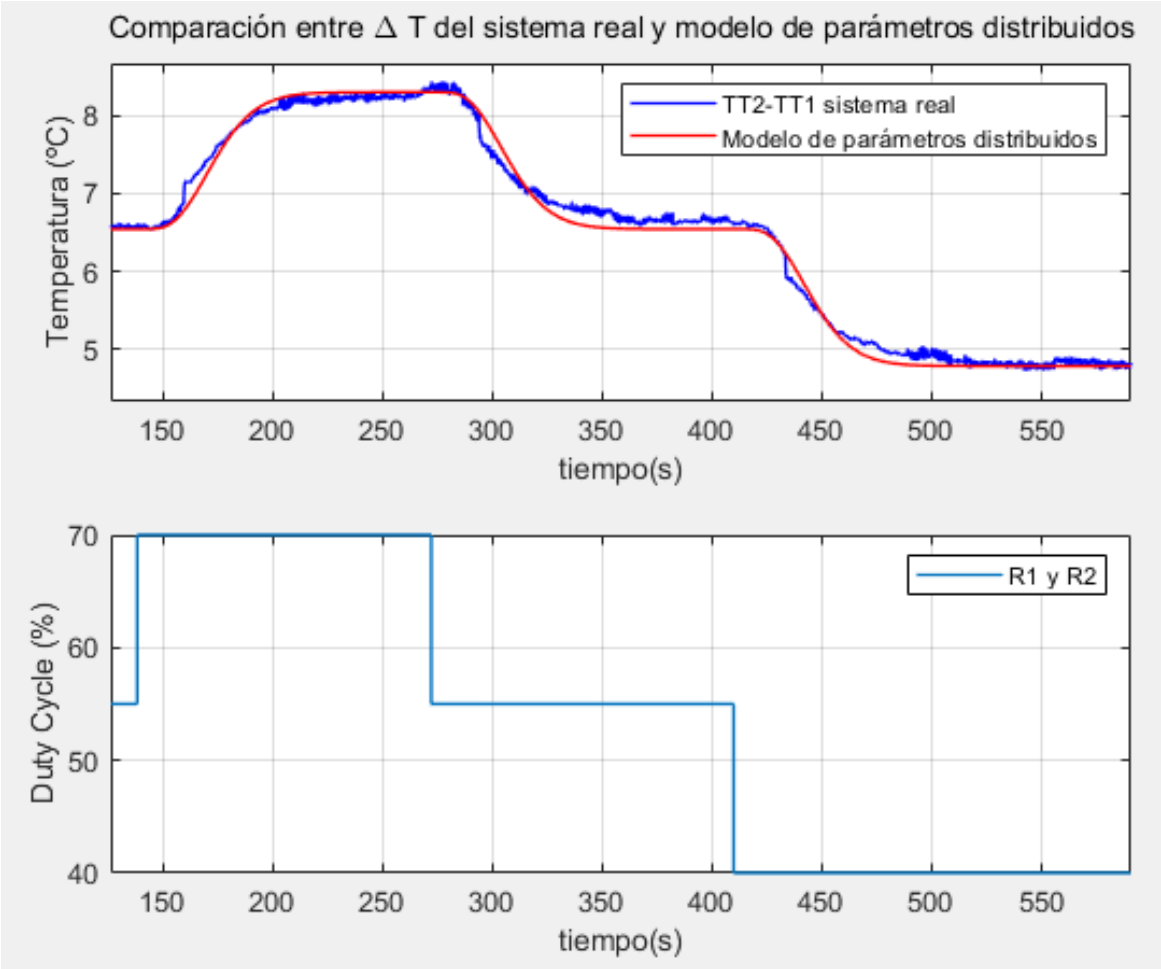


Figura 2.5 – Comparación entre temperatura del sistema real y modelo de parámetros distribuidos con escalones en R, manteniendo VR1=Cte.

La figura 2.5 muestra que la respuesta de la temperatura ante cambios en R con VR1 constante es típica de un sistema de primer orden con retardo. Ajustando los parámetros del modelo se ha conseguido un comportamiento del modelo similar al sistema real en términos de retardo, constante de tiempo y ganancia estática, tanto en escalones positivos como negativos.

2.1.1.3. Modelo de temperatura ante cambios en VR1, con R=Cte.

Para el tercer y último caso, en el que R es constante al 55% y se aplican escalones a la apertura de VR1, se analiza la respuesta de la temperatura:

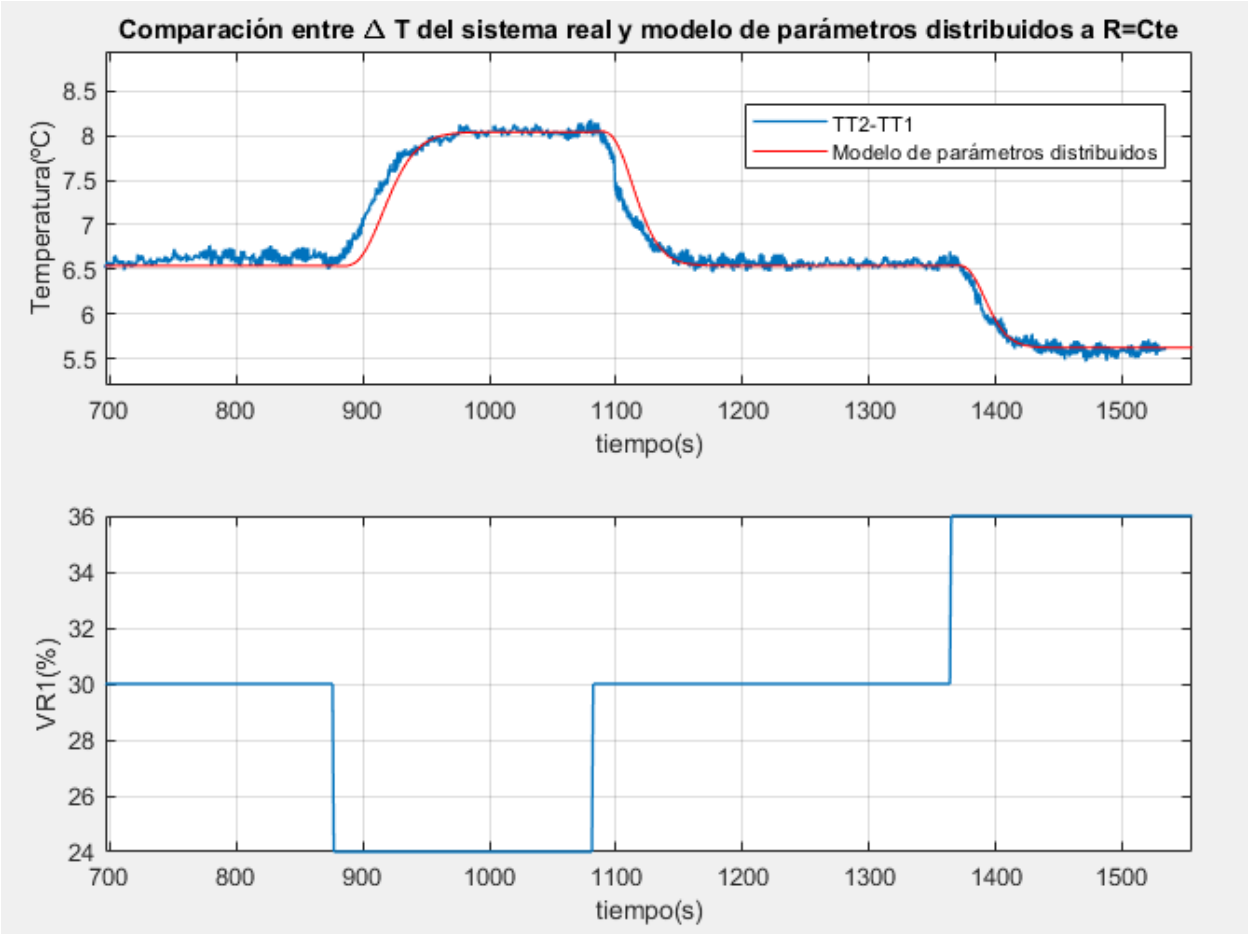


Figura 2.6 – Comparación entre temperatura del sistema real y modelo de parámetros distribuidos con R=55% Cte, y escalones en VR1.

En la figura 2.6 se aprecia que la respuesta del sistema real ante cambios en VR1 con R constante es también propia de un modelo de primer orden con retardo. En este caso, la constante de tiempo y la ganancia son aceptables, sin embargo, el ajuste del retardo no resulta tan satisfactorio. Esto es debido a que, si se intenta mejorar el ajuste del retardo en esta parte del sistema, el retardo de la temperatura ante cambios en R se ve afectado.

Por este motivo, no ha sido posible ajustar perfectamente todo el modelo, pero, en cualquier caso, se ha considerado prioritario el ajuste de los comportamientos mostrados en las figuras 2.4 y 2.5.

3 IMPLEMENTACIÓN EN EASY JAVASCRIPT SIMULATIONS (EJSS)

3.1 Introducción a EJSS.

Easy JavaScript Simulations es un software de código abierto basado en Java o JavaScript (dependiendo del lenguaje elegido) creado con el propósito de proporcionar al usuario una forma sencilla de crear simulaciones interactivas sin necesidad de tener un conocimiento especializado en programación.

Permite implementar modelos de forma muy intuitiva, y simularlos a partir de la integración ecuaciones diferenciales en las que estos se basan.

Otra de sus principales características es que su ejecución requiere de una vista en formato HTML que permite al usuario trabajar con el navegador web. De este modo, resulta adecuado para la implementación de proyectos de control con conexión remota como es el caso del Laboratorio Virtual, en el que se pretende trabajar con la planta de forma remota en Moodle.

Para este proyecto, la versión de EJSS elegida ha sido la 5.3, ya que es una versión estable en la que los desarrolladores han verificado su correcto funcionamiento al trabajar con el protocolo de comunicación RIP. Por otro lado, el lenguaje de programación en el que se basa es JavaScript, también por motivos de compatibilidad.

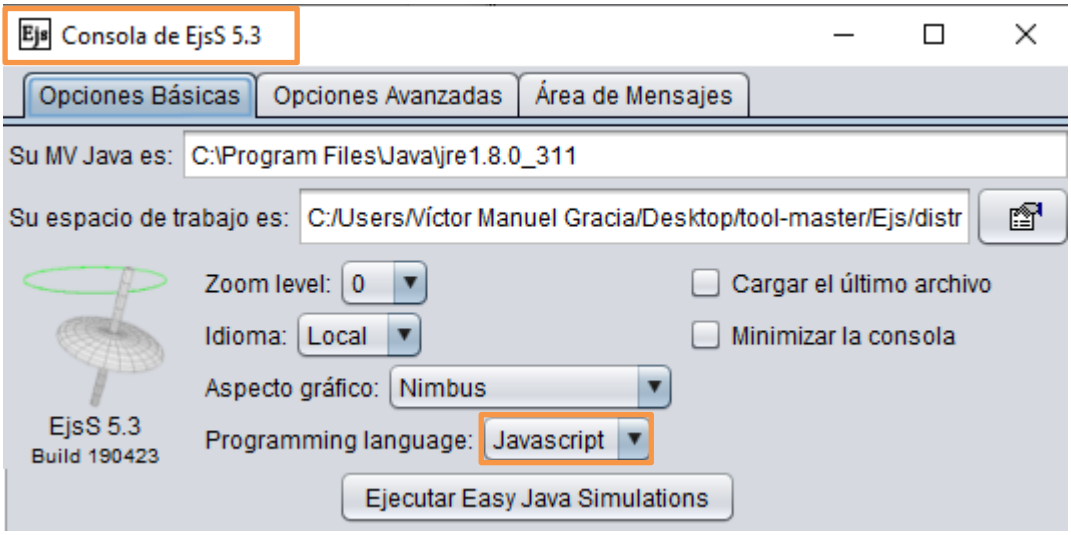


Figura 3.1 – Versión y lenguaje de programación en EJSS.

3.1.1 Pestaña Descripción.

En la pagina de descripción el usuario puede crear una portada y una descripción del proyecto creado. Tal y

como muestra la figura 3.2, presenta un formato de procesador de texto para el desarrollador, donde se pueden insertar imágenes, hipervínculos, etc.

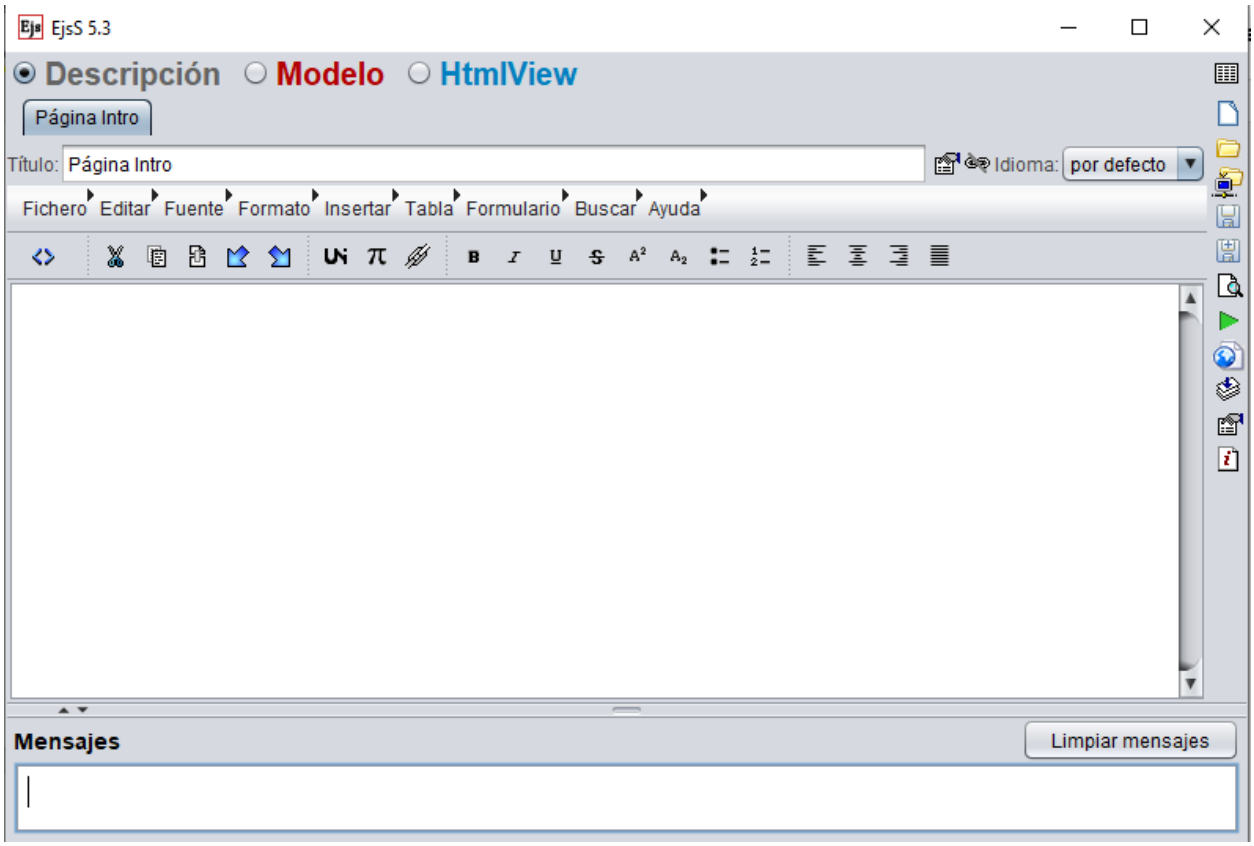


Figura 3.2 – Página de Descripción en EJSS.

3.1.2 Pestaña Modelo.

Esta pestaña concentra toda la parte técnica del proyecto. Su principal función queda determinada en la subpestaña "Evolución", mostrada en la figura 3.3, que permite la integración de las ecuaciones diferenciales mediante el método de integración elegido en la página EDO (Ecuaciones Diferenciales Ordinarias, también conocidas en inglés como ODE) (figura 3.4), o crear páginas de código que permiten escribir el código de evolución libremente, de, por ejmplo, un controlador PID o una señal Chirp. Además, con las opciones "Imágenes por segundo" y "Pasos por visualización" es posible ajustar la velocidad con la que se ejecuta la simulación.

Por otro lado, la subpestaña "Variables" permite (y necesita) la creación de tablas que contienen las variables a utilizar en el proyecto, definiendo también el tipo de cada una de ellas.

La subpestaña "Inicialización", en nuestro caso, será útil para inicializar el protocolo RIP si el programa lo requiere.

Por otro lado, la subpestaña "Relaciones fijas" permite crear expresiones matemáticas fijas entre variables.

La subpestaña "Propio" proporciona la posibilidad de crear funciones propias a utilizar en el proyecto, con sus respectivas entradas y salidas.

Por último, la subpestaña "Elementos" nos será de utilidad en nuestro caso para crear el "Elemento RIP" que establecerá la conexión con LabView desde EJSS:

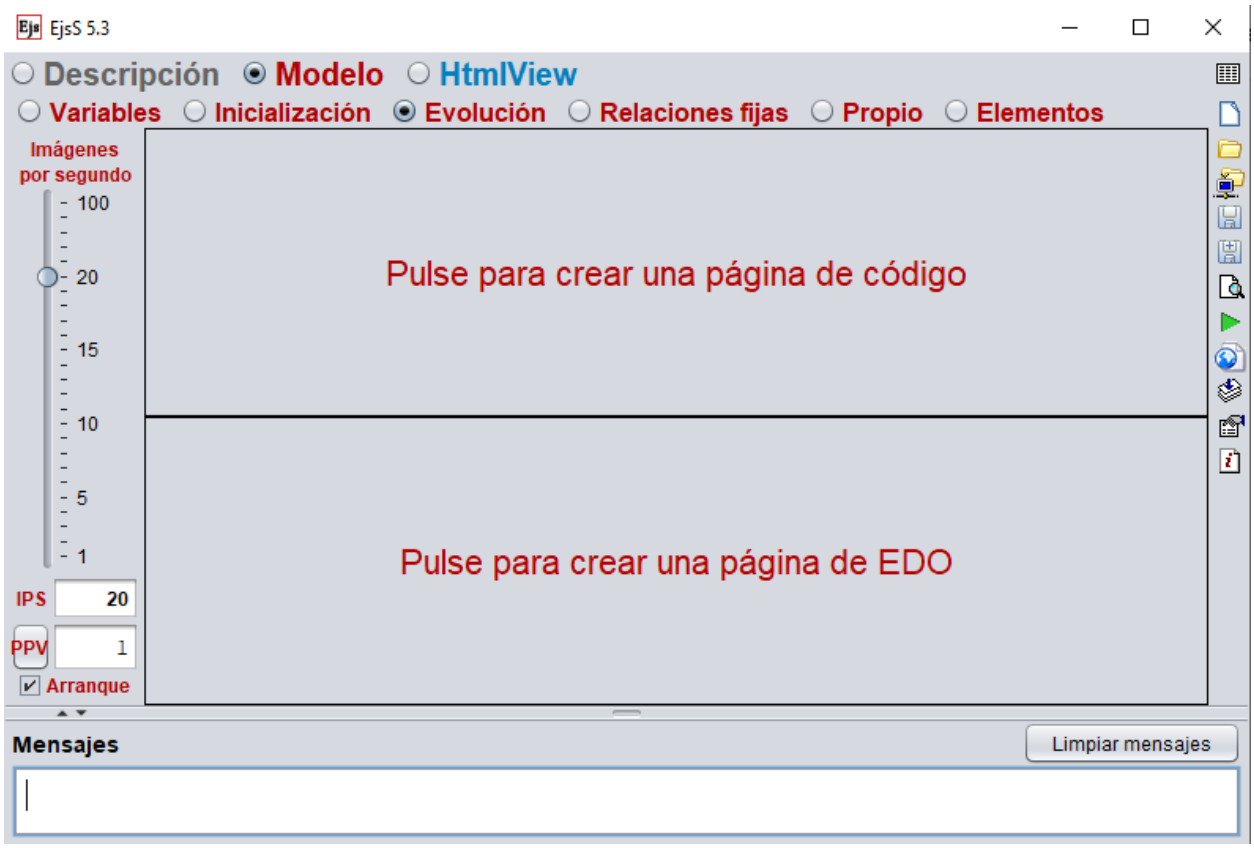


Figura 3.3 – Pestaña de Modelo en EJSS, subpestaña de Evolución.

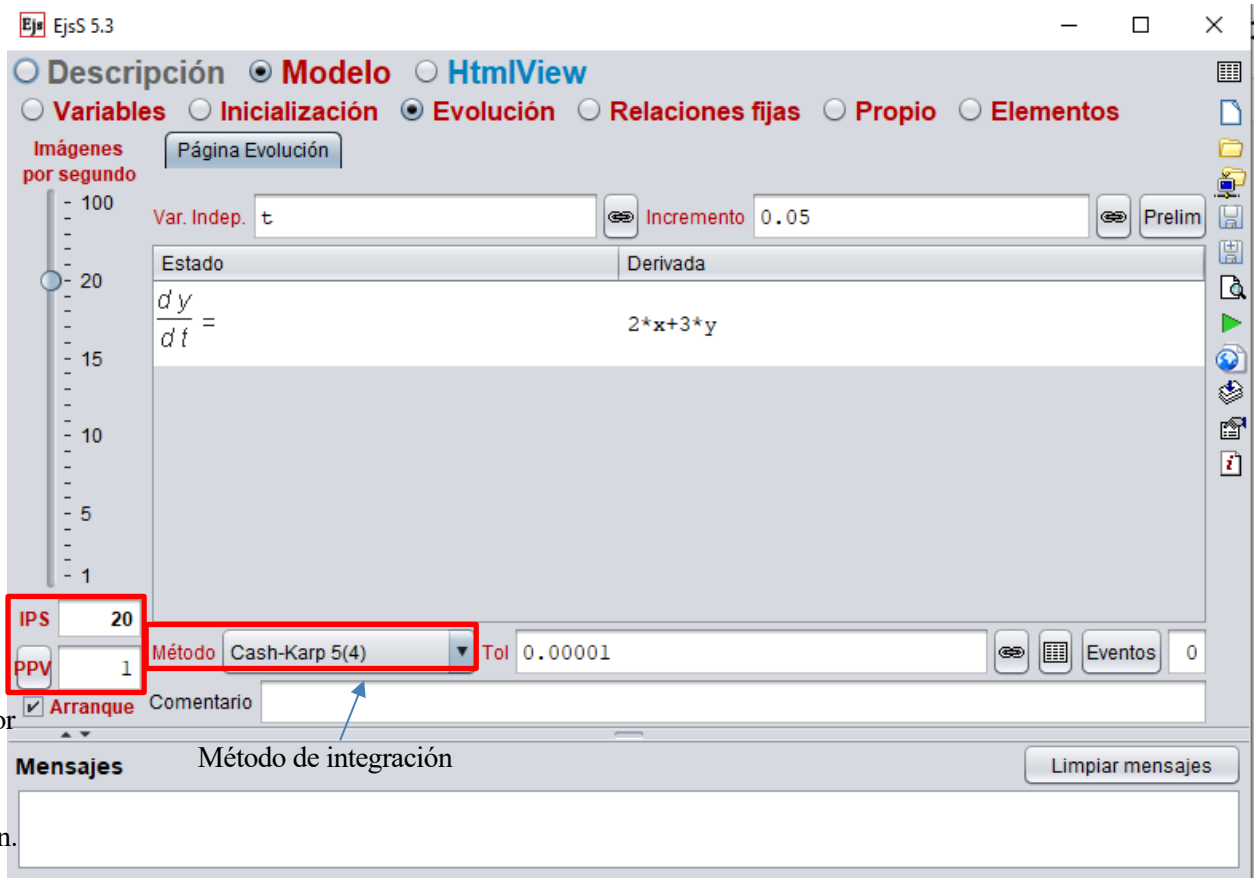


Figura 3.4 – Página ODE en EJSS.

3.1.3 Pestaña HTMLView.

En la pestaña HTMLView da gran variedad de opciones para crear la vista HTML que tendrá la simulación. Da la posibilidad de crear elementos interactivos de todo tipo, tales como elementos de entrada/salida (deslizadores, botones de uno y dos estados, campos numéricos), elementos de visualización (gráficas, figuras 2D y 3D con movimiento dependiente de variables, etc.), elementos multimedia (videos, música, imágenes, conexión con cámara web, etc.).

Como se muestra en la figura 3.5, las páginas presentan un esquema jerárquico en el que los elementos superiores son padres de los elementos inferiores (hijos).

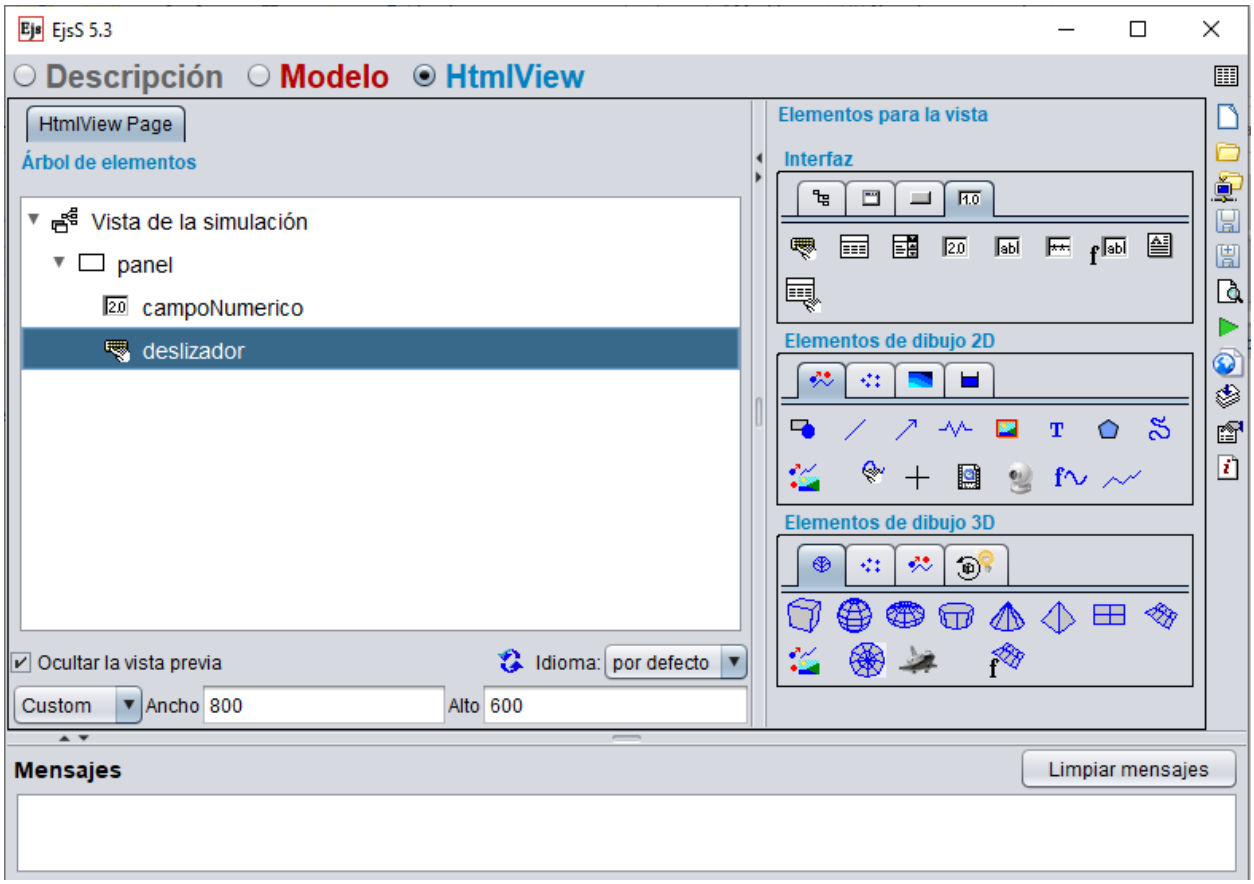


Figura 3.5 – Página HTMLView en EJSS.

3.2 Modelo.

En este apartado nos centraremos en detallar el contenido de la pestaña Modelo de este proyecto.

3.2.1 Subpestaña de Variables.

En las tablas de variables se encuentran todas las variables necesarias para la ejecución. Cabe destacar que se han creado tres tipos de variables de conexión que pueden disitinguirse según el nombre:

- Variables Planta (P): Son las variables que salen y entran al bloque de planta.
- Variables Control (C): Idem para bloque de control.
- Variables SCADA (SC): Idem para bloque de visualización (SCADA).

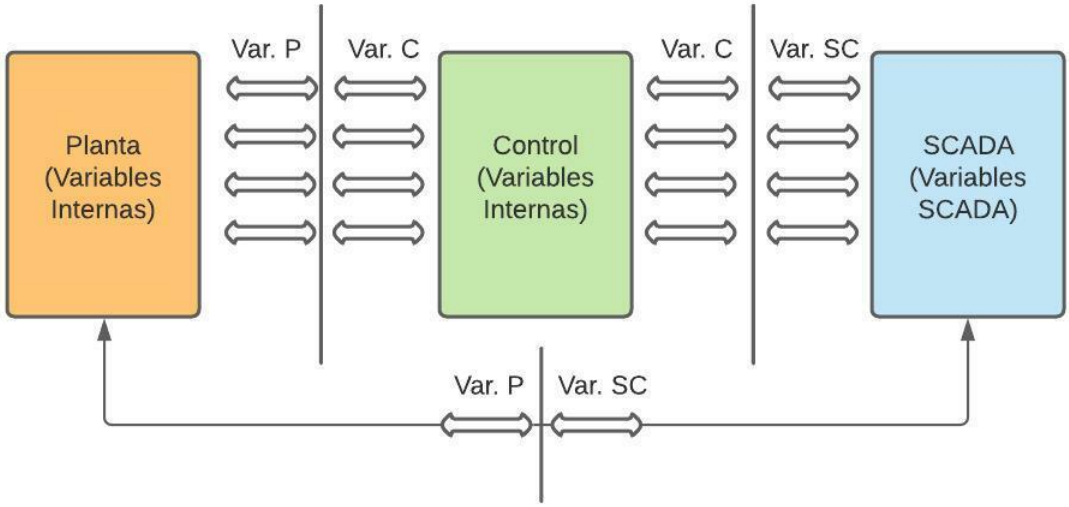


Figura 3.6 – Esquema: Tipos de variables según su función.

Gracias a las variables de conexión, es más sencillo entender la función que ejerce cada variable y realizar modificaciones si fuese necesario, además de agilizar la conexión posteriormente con LabView.

3.2.2 Subpestaña de Evolución.

En la subpestaña de evolución se han creado cinco páginas de código de evolución, y una página ODE.

3.2.2.1 Páginas de código.

En las páginas de código se ha implementado el código de las señales Chirp de ambas entradas (figuras 3.7 y 3.8), el código de los dos controladores PID, y una página para la impresión de los valores por la consola del navegador web.

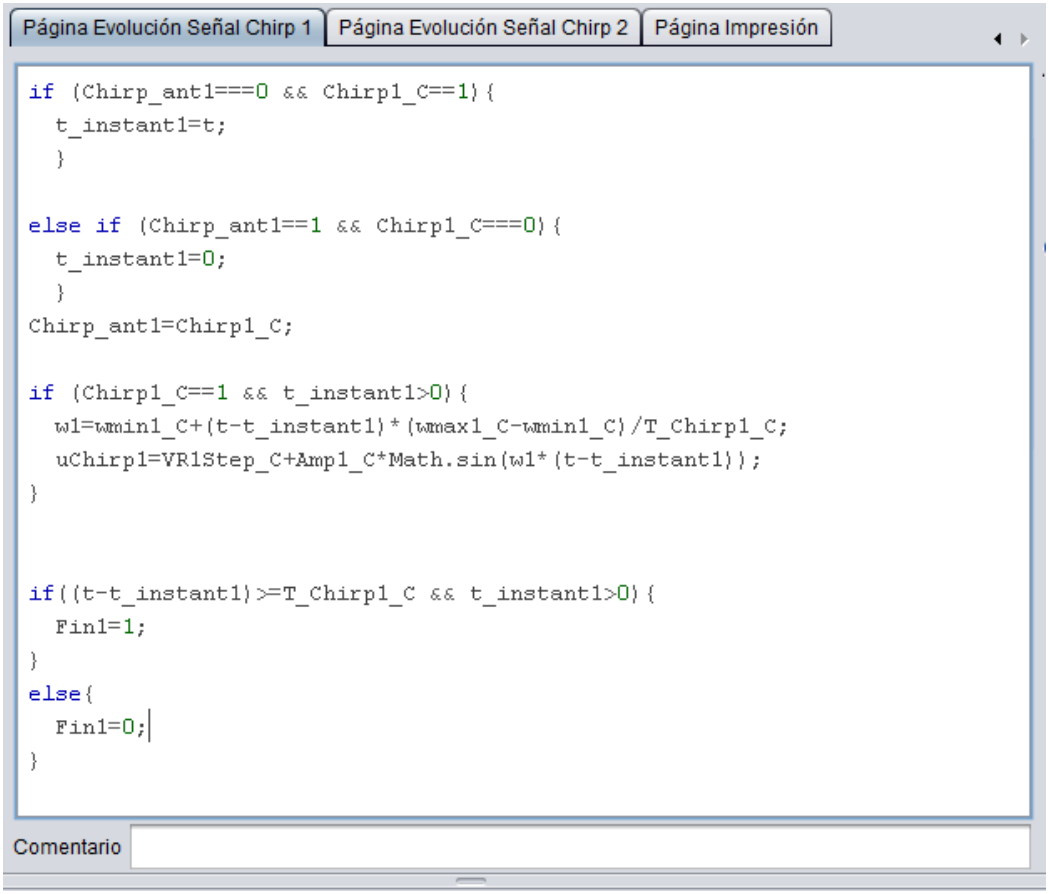


Figura 3.7 – Página de código de evolución: Señal Chirp para la entrada VR1.

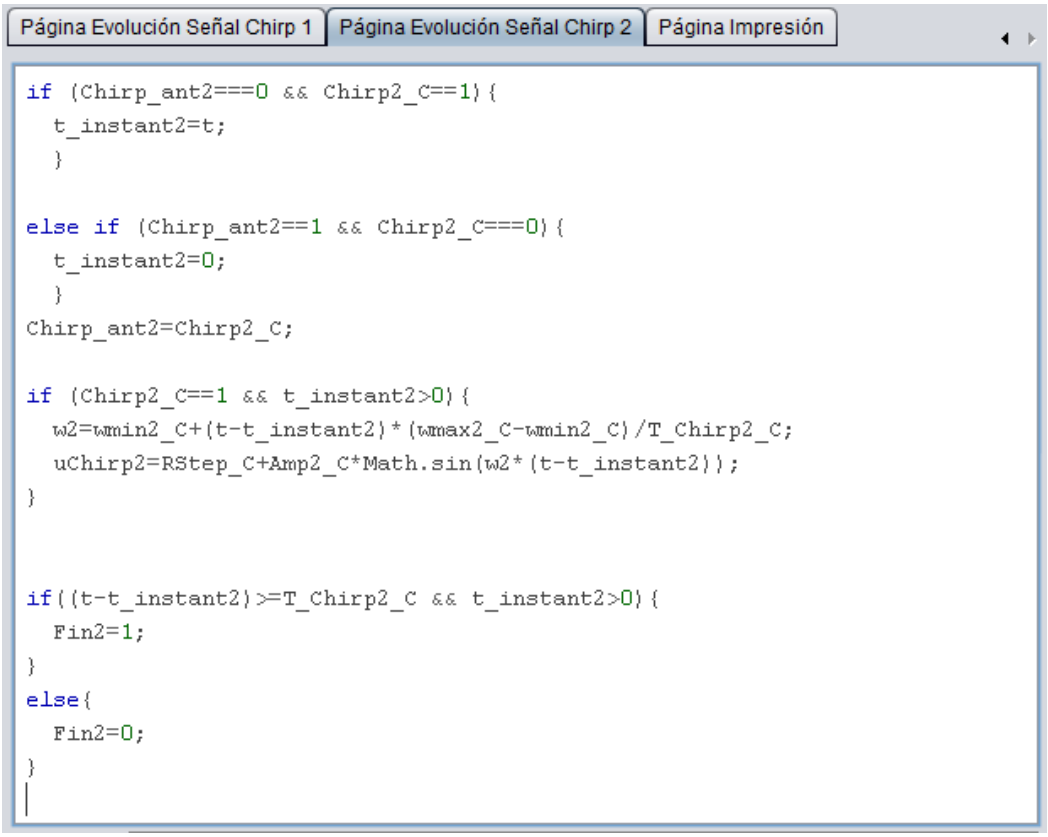


Figura 3.8 – Página de código de evolución: Señal Chirp para la entrada R.

Página EDO | Página Evolución PID 1 | Página Evolución PID 2 | Página Evolución Señal Chirp 1 | Página Evolución Señal Chirp 2 | Página

```
if (h>45)
h=45; //Saturación altura del tanque
else if (h<0)
h=0;

//PID con Anti-Windup
ed1_ant=ed1;
I1_ant=I1;
uPID1_ant=uPID1;
IAW1_ant=IAW1;
u1_ant=uk1;

if (MAN1_C===0){
  ep1=b1_C*Ref1_C-h_C; //Errores para el controlador
  ed1=c1_C*Ref1_C-h_C;
  ei1=Ref1_C-h_C;
}

else {
  ep1=0;
  ed1=0;
  ei1=0;
  ed1_ant=0;
}
```

Figura 3.9 – Página de código de evolución: PID para control de altura mediante VR1 (Parte 1).

Página EDO | Página Evolución PID 1 | Página Evolución PID 2 | Página Evolución Señal Chirp 1 | Página Evolución Señal Chirp 2 | Página

```
uP1=Kc1_C*ep1; //Parte proporcional de la señal de control
uD1=Kc1_C*Td1_C*(ed1-ed1_ant)/dt;
if(MAN1_C==1){
  I1=uk1/(Kc1_C/Ti1_C); //Asegura una transición suave entre los modos manual y automático
}

else{
  I1=I1_ant+ei1*dt;
}

uI1=Kc1_C/Ti1_C*I1;

Tr1=0.1*Ti1_C;

IAW1=IAW1_ant+dt*(u1_ant-uPID1_ant);

uAW1=1/Tr1*IAW1;

if (AW1_C===0){ //Si el anti-windup del controlador de nivel no está activo
  uAW1=0;
}

uPID1=uP1+uD1+uI1+uAW1;

if (MAN1_C==1){
```

Figura 3.10 – Página de código de evolución: PID para control de altura mediante VR1 (Parte 2).

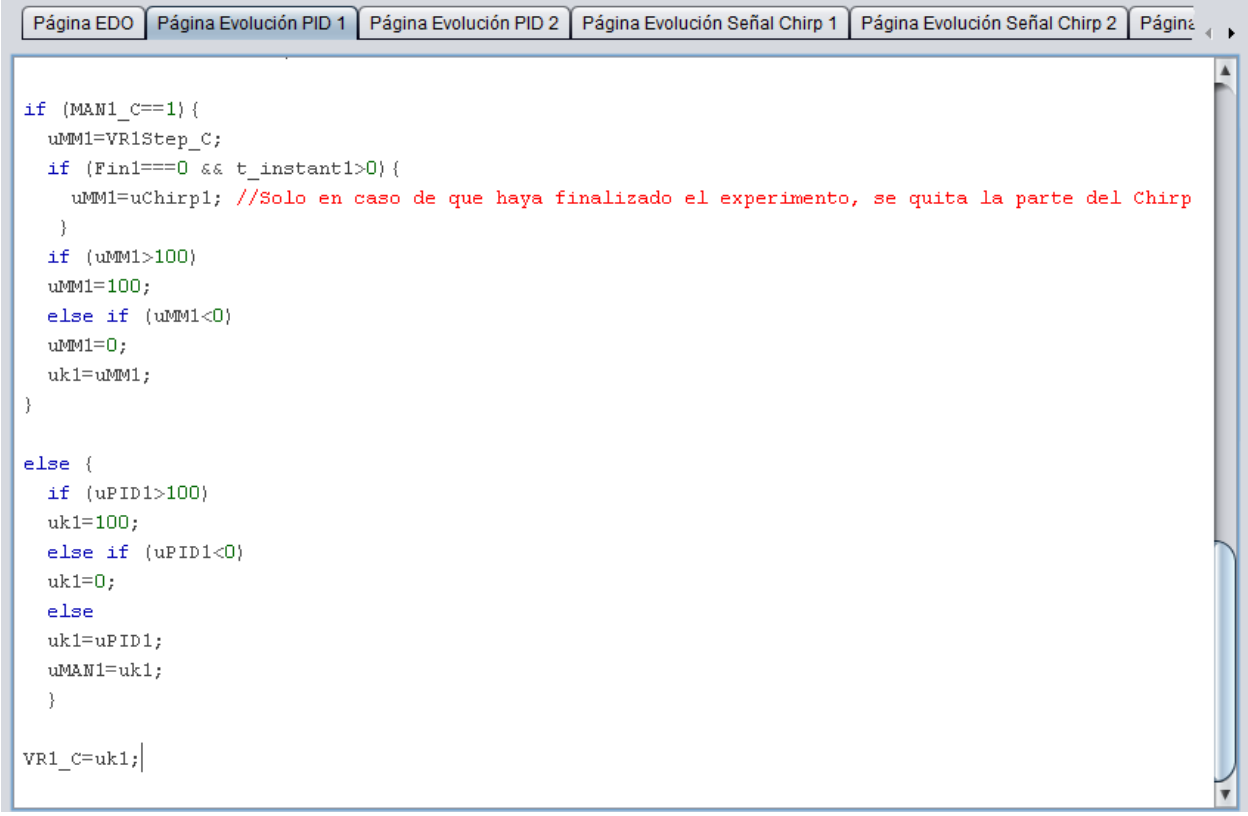


Figura 3.11 – Página de código de evolución: PID para control de altura mediante VR1 (Parte 3).

Se omitirá el código del controlador de temperatura mediante R, ya que es análogo al visto en las figuras 3.9 a 3.11, con la única diferencia de que la salida 2 (TT2-TT1) no tiene límites de saturación al no tener una limitación física como sí lo es la altura del tanque de nivel.

Por último, se muestra la página de impresión de datos, que se encargará de imprimir los datos de los experimentos por la consola del navegador web. Ésta se muestra en la figura 3.11.

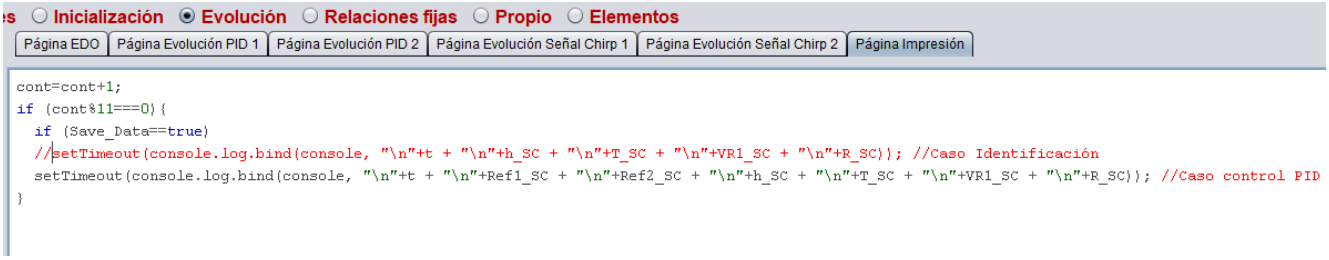


Figura 3.12 – Página de código de evolución: Página de impresión de datos.

El código propuesto en la figura 3.12 implica la necesidad de comentar una de las dos líneas de impresión en función de si nos encontramos en un experimento de identificación (no es necesario imprimir la referencia) o de control (referencias necesarias).

3.2.2.2 Páginas ODE.

Como se ha comentado, sólo se ha creado una página de ecuaciones diferenciales. En ella se han implementado tanto las ecuaciones de la resistencia, como la del modelo de nivel. Esto se puede apreciar en las figuras 3.13 y 3.14.

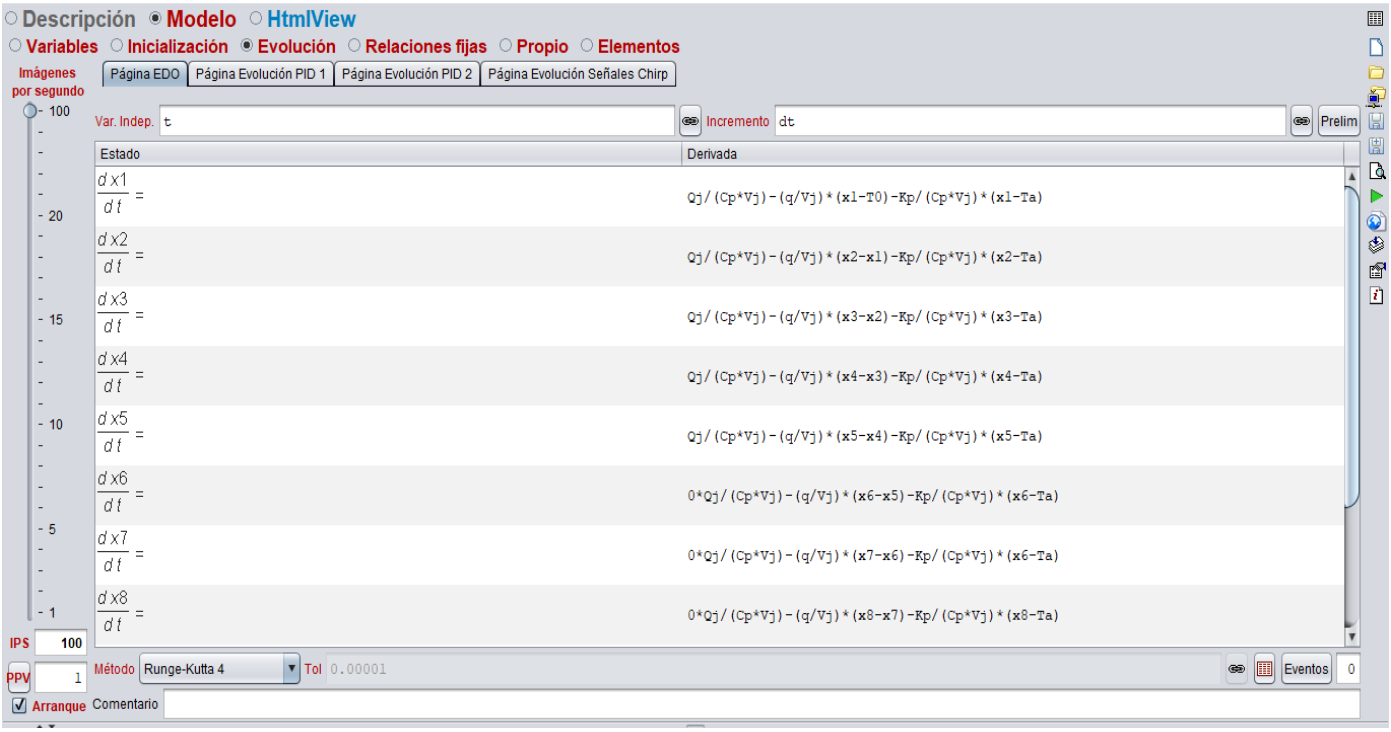


Figura 3.13 – Página ODE: Ecuaciones diferenciales implementadas en EJSS (Parte 1).

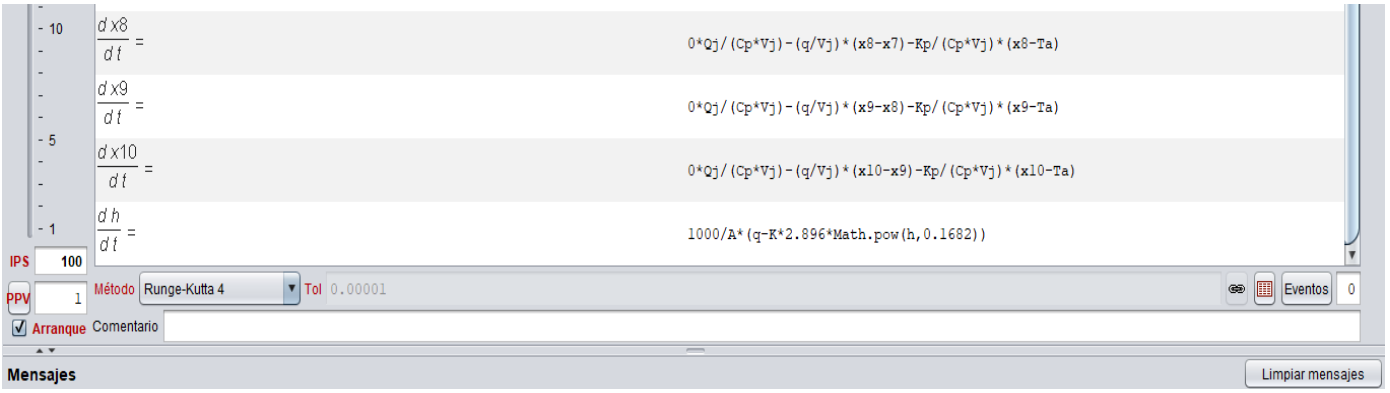


Figura 3.14 – Página ODE: Ecuaciones diferenciales implementadas en EJSS (Parte 2).

Se puede comprobar a partir de las dos figuras anteriores, que existen 10 ecuaciones diferenciales asociadas a la temperatura, es decir, tantas como número de secciones elegimos en Matlab. Por otro lado, también se puede apreciar en la figura 3.14 la ecuación diferencial que modela el nivel.

Se debe destacar que el método de integración utilizado ha sido Runge-Kutta de orden 4.

3.2.3 Subpestaña de Relaciones fijas.

En este caso, se han creado dos páginas de relaciones fijas. La primera de ellas (figura 3.15), está dedicada a algunas relaciones fijas necesarias para el modelo, tales como parámetros a partir de otros parámetros.

También se ha creado una ecuación para calcular el color del líquido según la temperatura que se utiliza para la animación del tanque de nivel. Ésta hace que el color pase de azul oscuro a rojo de forma gradual según la temperatura TT2-TT1.

Conexión Variables | Página RelFijas

```
T_P=x10-T0; //Incremento de temperatura

Vj=S*LR/(NR*1000); //Volumen en j
Qj=(PR*R_P/100)/NRe; //Potencia que da la resistencia por volumen de control

// Relación caudal[l/s] - VR1[%], en tres tramos, cada uno con ajuste por mínimos cuadrados
if (VR1_P<=30)
q=(12.5643*VR1_P+0.9286)/3600;
else if (VR1_P>30 && VR1_P<=70)
q=(9.4960*VR1_P+95.8)/3600;
else
q=(6.2667*VR1_P+318)/3600;

//Para modificar el color de la tubería según la temperatura del agua
color=0.481*T_SC+8;
if (T_SC>8.32)
color=12;
```

Figura 3.15 – Página ODE de relaciones fijas.

3.2.3.1 Relación Caudal-VR1.

Ya se comentó en el apartado 2.1.3 cómo se lleva cabo la obtención del caudal q en el modelo implementado en Matlab, ya que ésta presentaba cierta no linealidad. En la figura 3.16 se puede apreciar qué tan no lineal es ayudándonos de la inserción de una recta en la gráfica para comprobarlo.

Sin embargo, por simplicidad se ha decido en EJSS elegir una función obtenida del ajuste por mínimos cuadrados en tres tramos distintos, por lo que la relación implementada sigue tres tramos lineales, como se muestra en la figura 3.17.

El ajuste que siguen es el siguiente:

$$\text{Tramo 1: } q = 12.5643 \cdot VR1 + 0.9286$$

$$\text{Tramo 2: } q = 9.4960 \cdot VR1 + 95.8$$

$$\text{Tramo 3: } q = 6.2667 \cdot VR1 + 318$$

Hay que destacar que lo que se obtiene de estas ecuaciones es el caudal en litros/hora. Si se quiere pasar a litros/segundo (el modelo requiere esta unidad) se debe dividir entre 3600, como se puede observar en la figura 3.15.

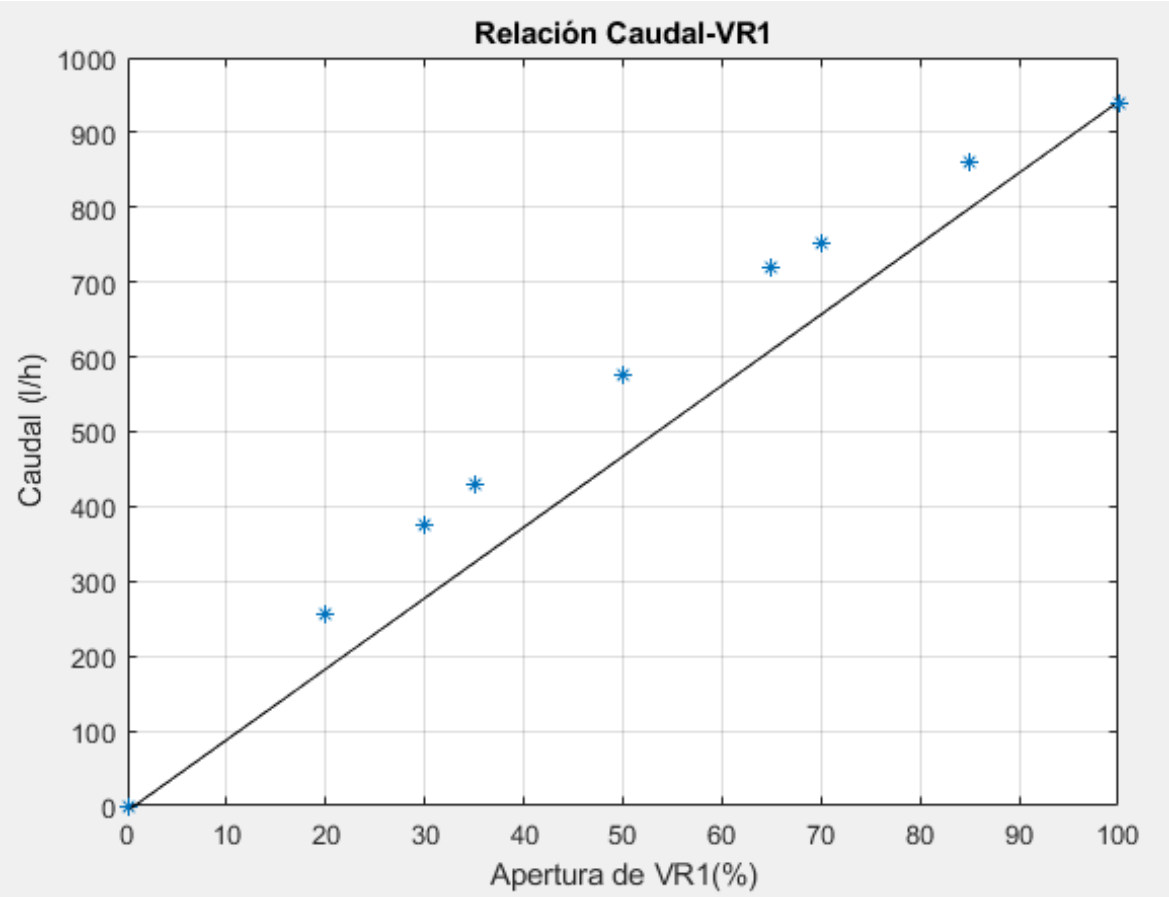


Figura 3.16 – Relación Caudal-VR1.

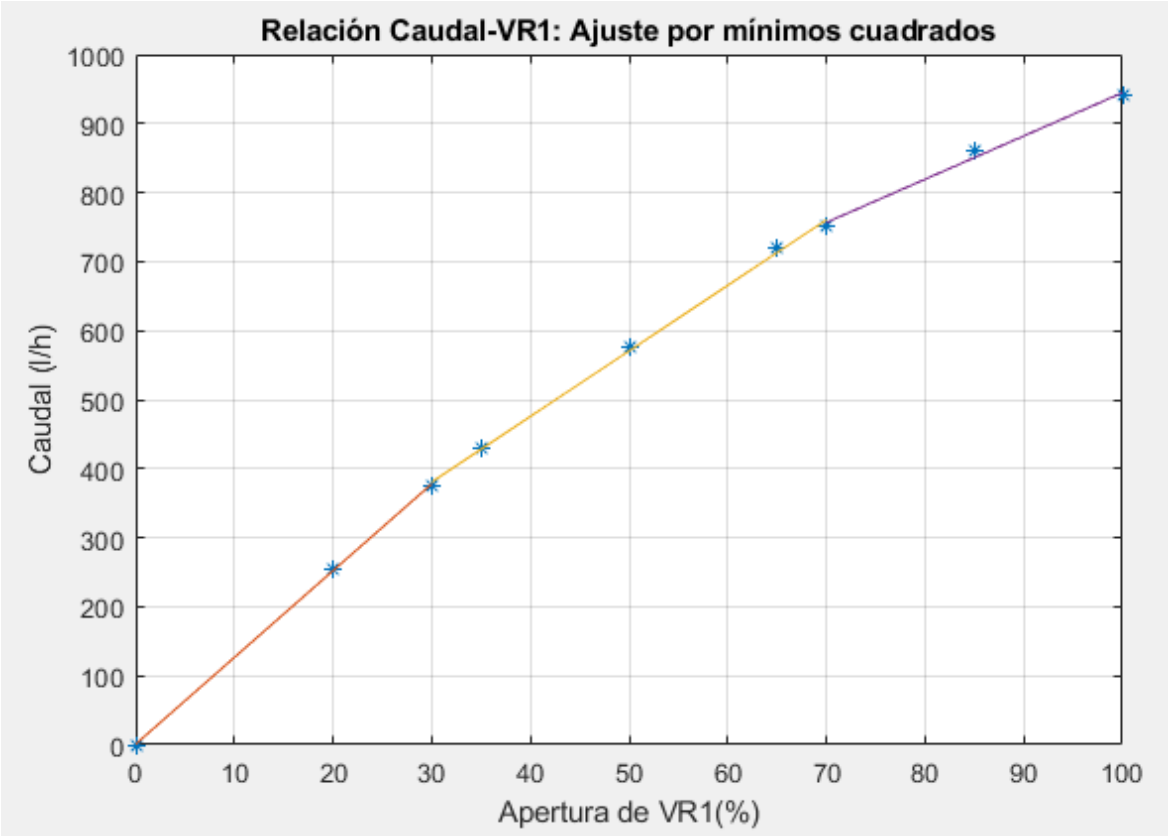


Figura 3.17 – Relación Caudal-VR1 con ajuste por mínimos cuadrados en tres tramos.

En la figura 3.17 se puede apreciar que los resultados obtenidos por la aproximación son aceptables.

Por otro lado, se ha creado otra página de evolución para la conexión de las variables, como se mostraba en la figura 3.6.

3.3 Vista HTML.

Este apartado está dedicado a la creación de la vista HTML del Laboratorio Virtual. Ésta se ha dividido en dos paneles: Panel de Visualización y Panel de Configuración.

Para una mejor gestión del espacio, ambos paneles hacen uso de pestañas, de forma que permiten alternar entre vistas de forma sencilla.

Todo lo comentado se puede apreciar en la figura 3.18.

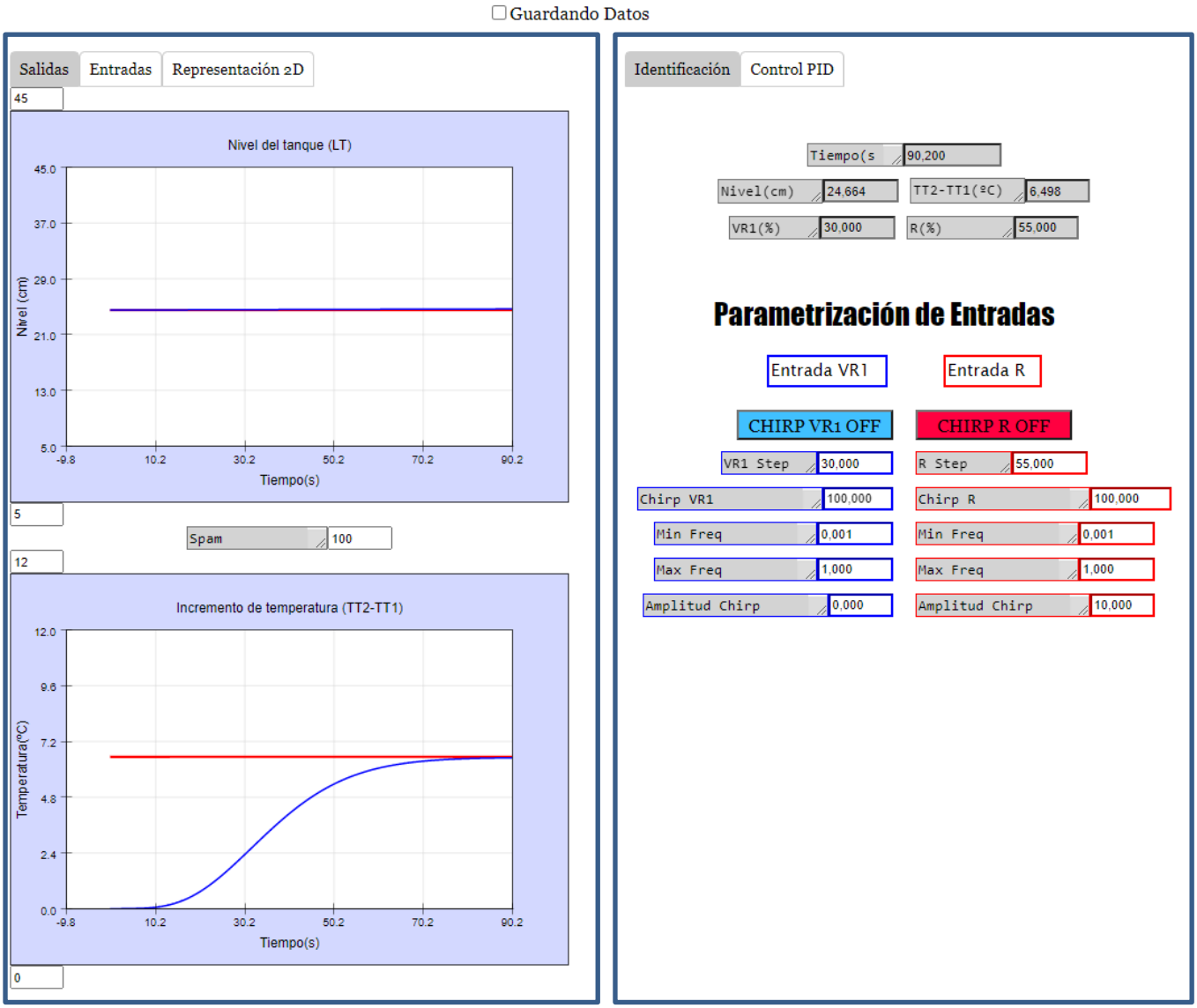


Figura 3.18 – A la izquierda: Panel de Visualización. A la derecha: Panel de Configuración.

3.3.1 Panel de Visualización.

Este panel tiene como objetivo representar entradas y salidas del sistema. Está compuesto por tres pestañas:

- Salidas: Muestra las salidas del sistema (nivel LT y temperatura TT2-TT1).

- Entradas: Muestra las entradas del sistema (apertura de la válvula VR1 y duty cycle de las resistencias R).
- Representación 2D: Muestra un esquema sencillo del sistema, con altura variable en el depósito, y tuberías y contenido del depósito que cambian de color con la temperatura.

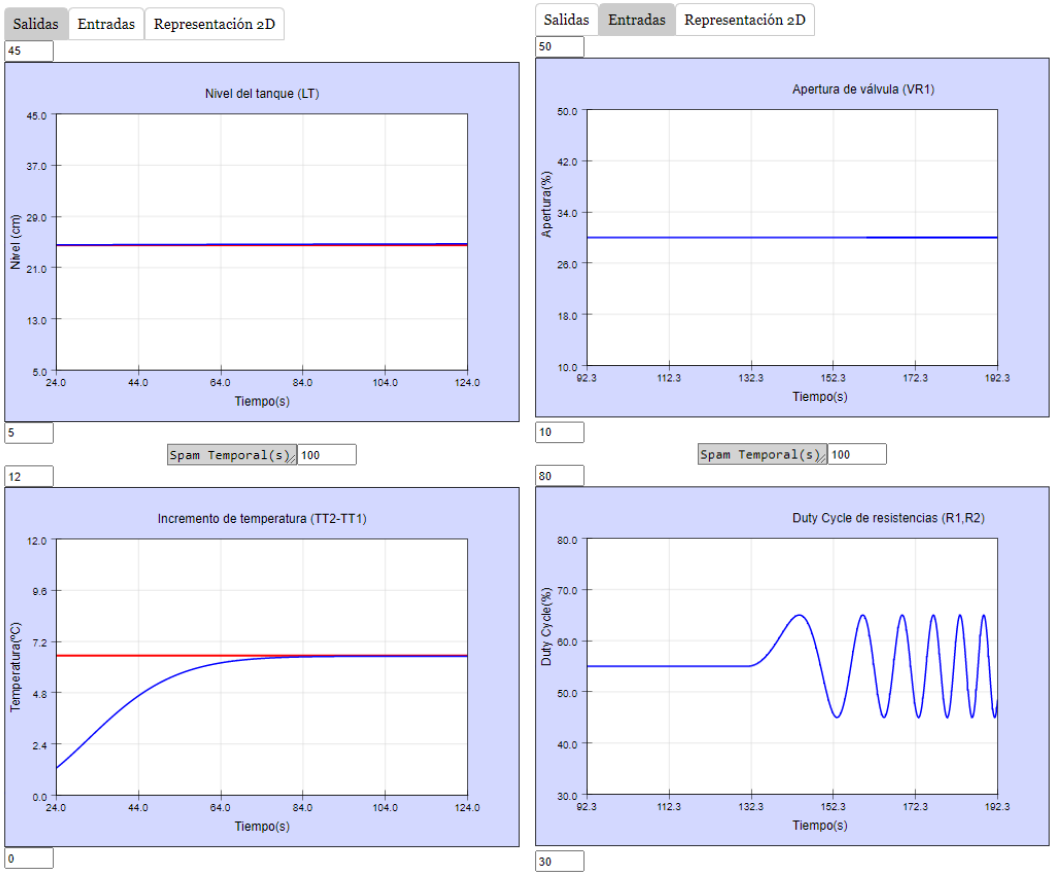


Figura 3.19 – A la izquierda: Pestaña de Salidas. A la derecha: Pestaña de Entradas.

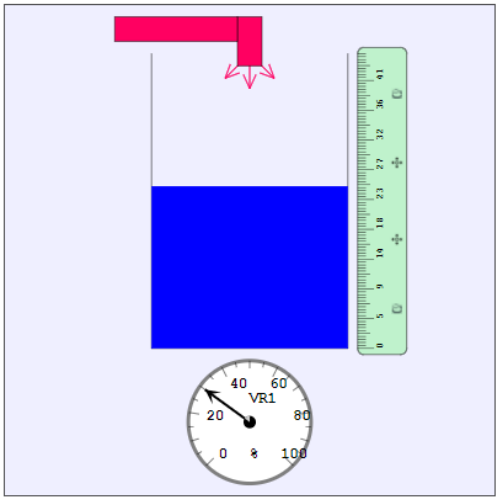


Figura 3.20 – Pestaña de Representación 2D.

3.3.2 Panel de Configuración.

Este panel permite al usuario visualizar los valores numéricos de entradas, salidas y tiempo. Además, da la posibilidad de elegir el tipo de entradas que quiere introducir al modelo, parametrizar las señales y controladores, entre otras cosas.

Las pestañas que componen este panel son dos:

- Identificación: Permite parametrizar las entradas y elegir entre entrada en escalón y Chirp.
- Control PID: Da la posibilidad de alternar entre los modos manual y automático de los controladores, parametrizarlos, y elegir si se desea que los controladores tengan anti-Windup o no.

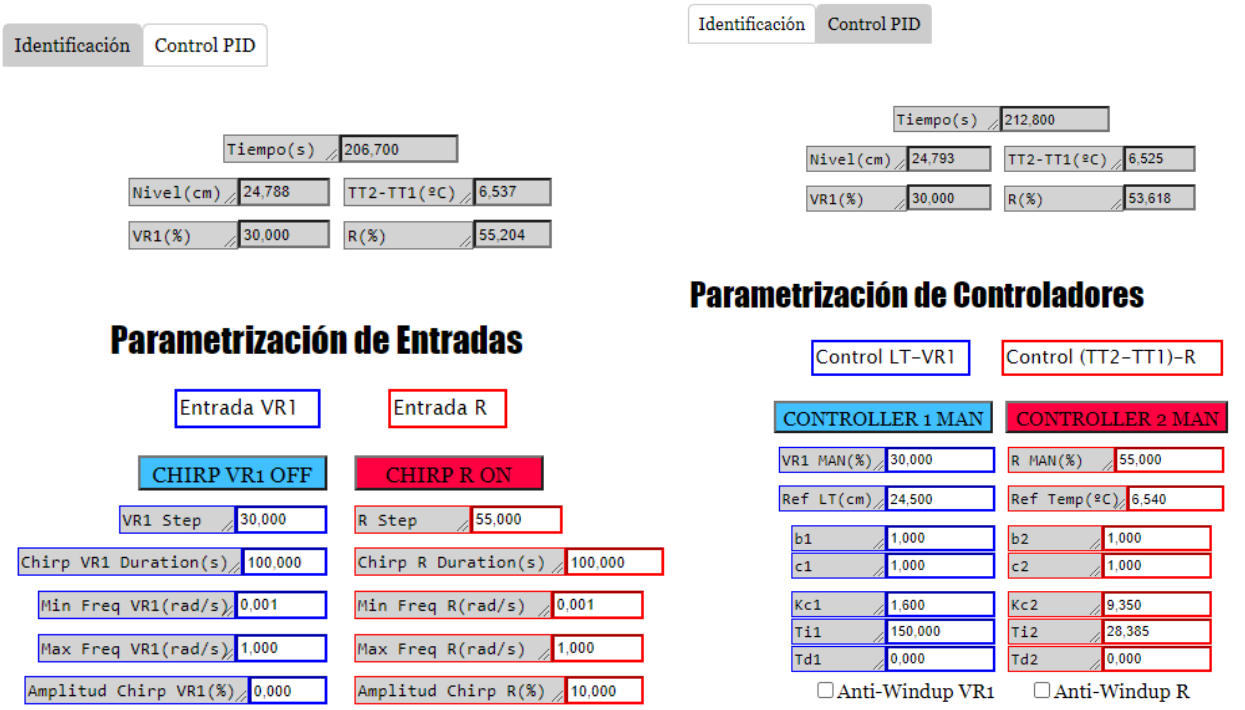


Figura 3.21 – A la izquierda: Pestaña de Identificación. A la derecha: Pestaña de Control PID.

4 MODO LOCAL: EJECUCIÓN SOBRE EL MODELO EN EJSS

Se ha llamado "Modo Local" a la ejecución del programa exclusivamente sobre EJSS. Es decir, este programa será completamente funcional sin necesidad de programas adicionales que no sean Java Environment, y Matlab para la representación de gráficas.

4.1 Validación de resultados con Matlab.

Para validar los resultados, se realizará el mismo experimento en ambos programas para corroborar que la respuesta del modelo es la misma en ambos casos.

Se ha elegido dar un escalón a cada entrada en momentos distintos para validar el modelo completo. Si la implementación en EJSS es correcta, se debe obtener un resultado idéntico al de Matlab.

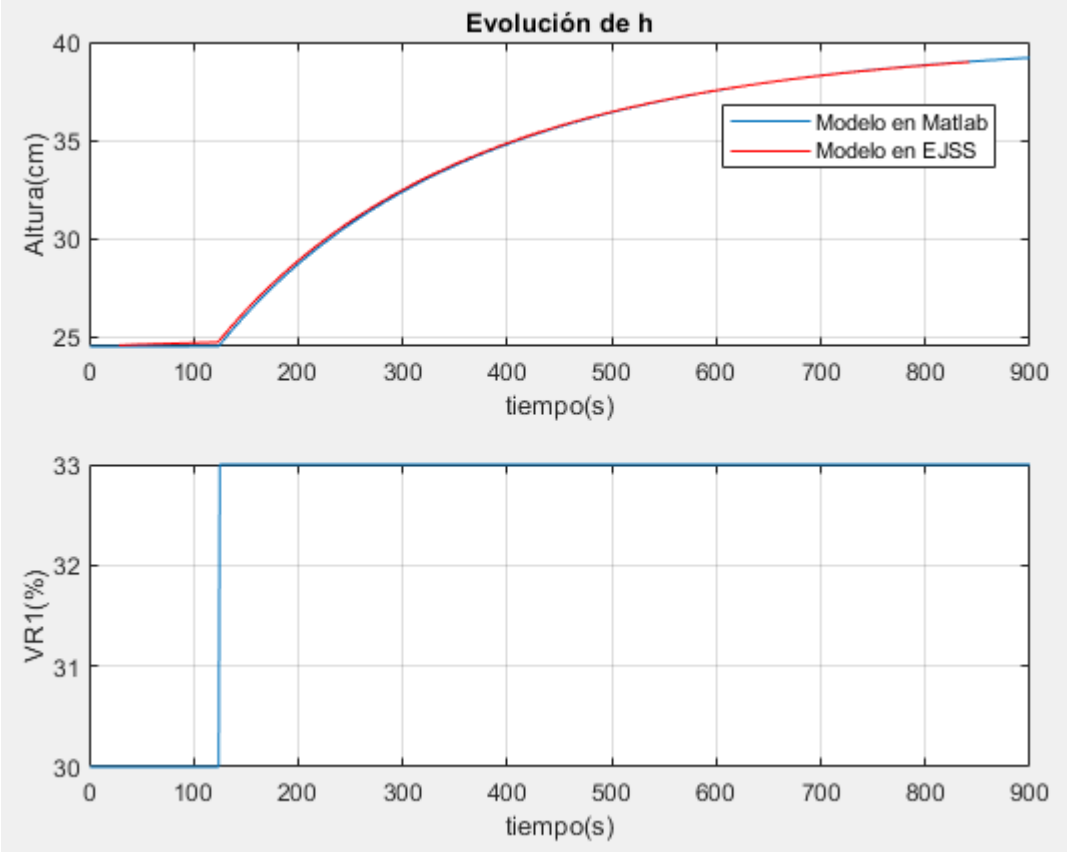


Figura 4.1 – Validación del modelo en EJSS. Evolución del nivel.

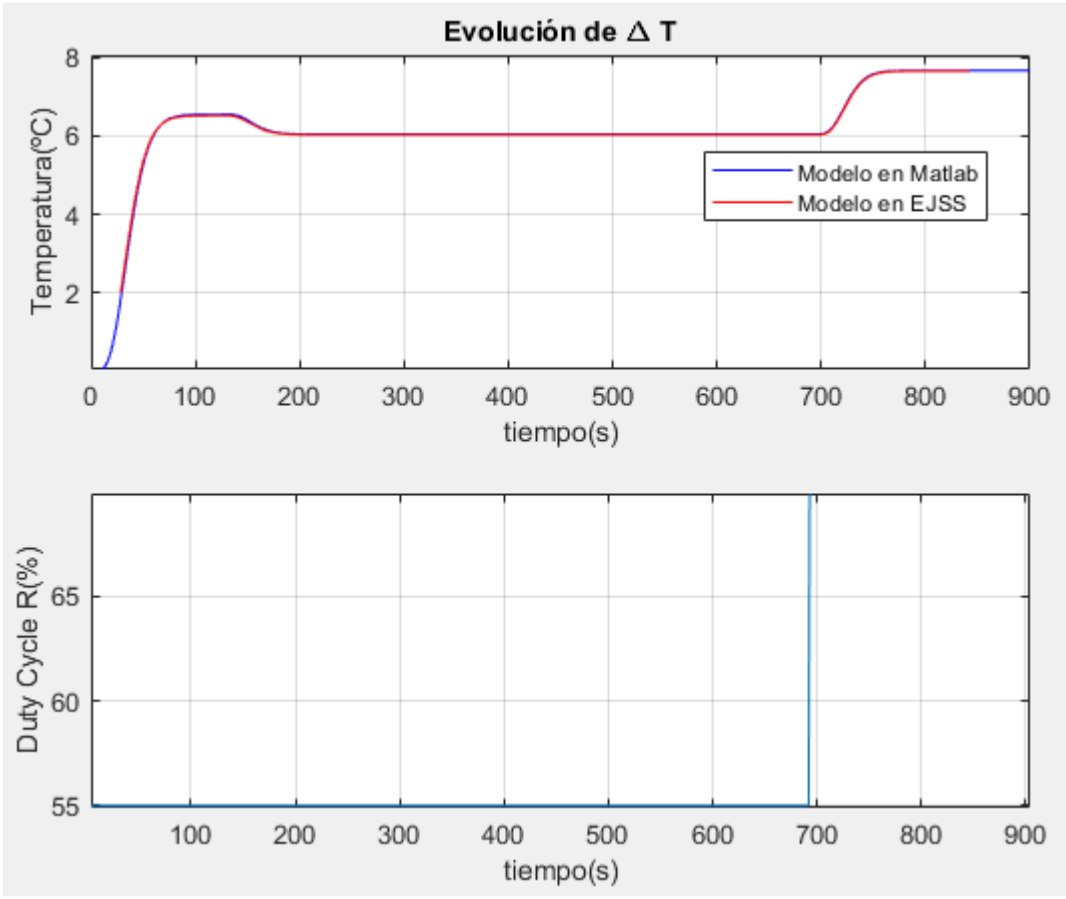


Figura 4.2 – Validación del modelo en EJSS. Evolución de la temperature.

Se puede comprobar a partir de las figuras 4.1 y 4.2 que los modelos se comportan de la misma forma en ambos softwares.

Se puede apreciar una leve diferencia entre ambos debido a ciertas aproximaciones en algunos decimales, y a que en Matlab solo se puede insertar escalones en valores enteros del tiempo por cuestiones de la implementación, a diferencia de EJSS.

4.2 Pruebas sobre EJSS.

Una vez comprobada la correcta implementación del modelo en EJSS, comprobaremos si el resto de la implementación (señales Chirp, botones, controladores PID y sus opciones) funciona adecuadamente.

4.2.1 Pestaña Identificación.

En este apartado verificaremos la funcionalidad de la pestaña de identificación del panel Configuración. Por simplicidad, se dará por validado el funcionamiento de las entradas en forma de escalón, ya que en las figuras 4.1 y 4.2 éstas han sido utilizadas.

Solo falta, por tanto, comprobar la funcionalidad de las señales Chirp y de los botones de activación de las mismas.

En la siguiente figura se puede comprobar un experimento donde ambas entradas, en momentos distintos, tienen una señal chirp de 200 segundos de duración, con frecuencias entre 0.001 y 0.1 rad/s.

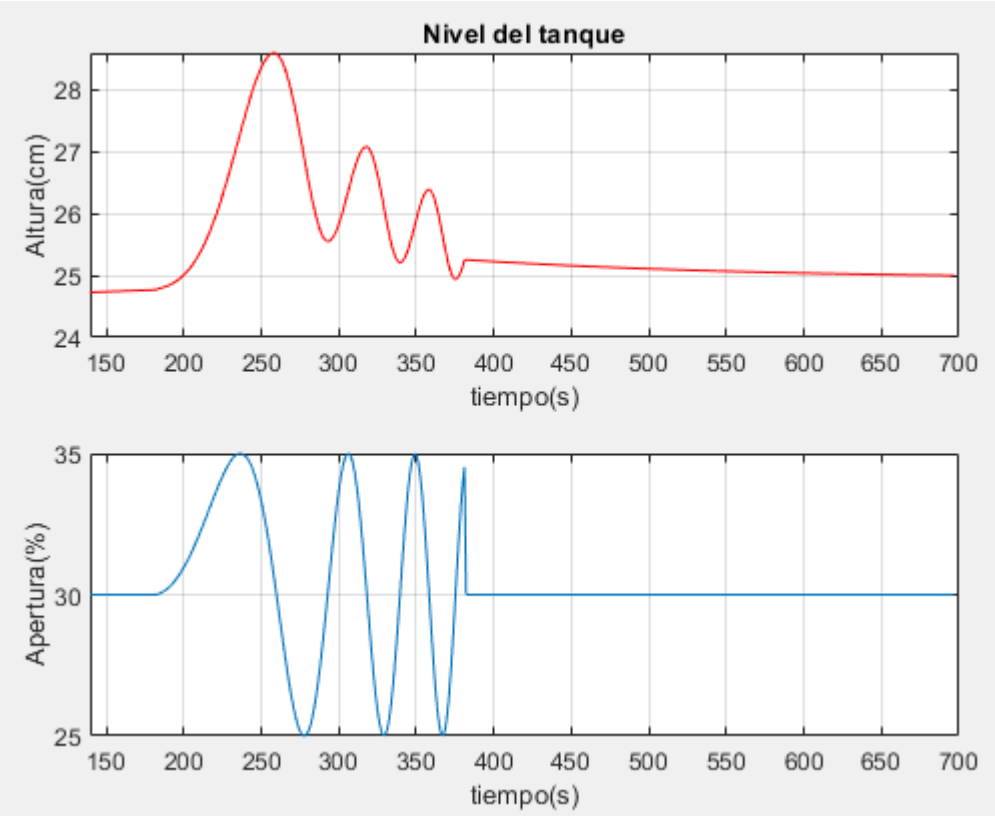


Figura 4.3 – Prueba de señal chirp en EJSS sobre entrada VR1.

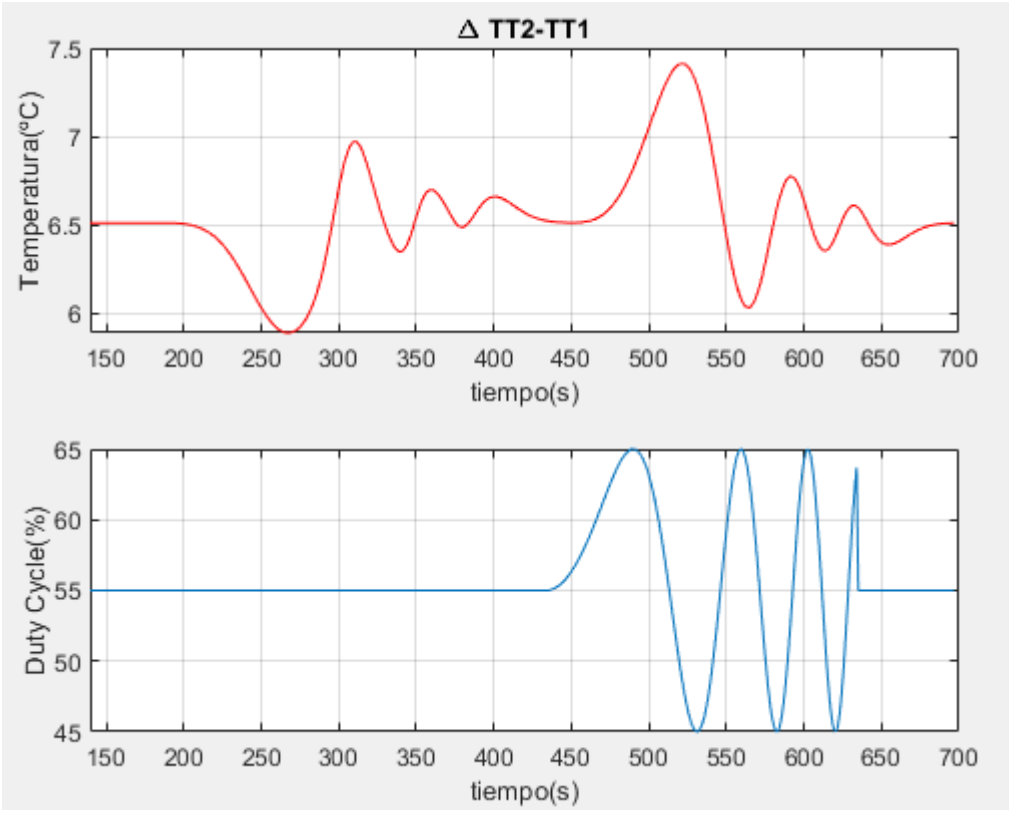


Figura 4.4 – Prueba de señal chirp en EJSS sobre entrada R.

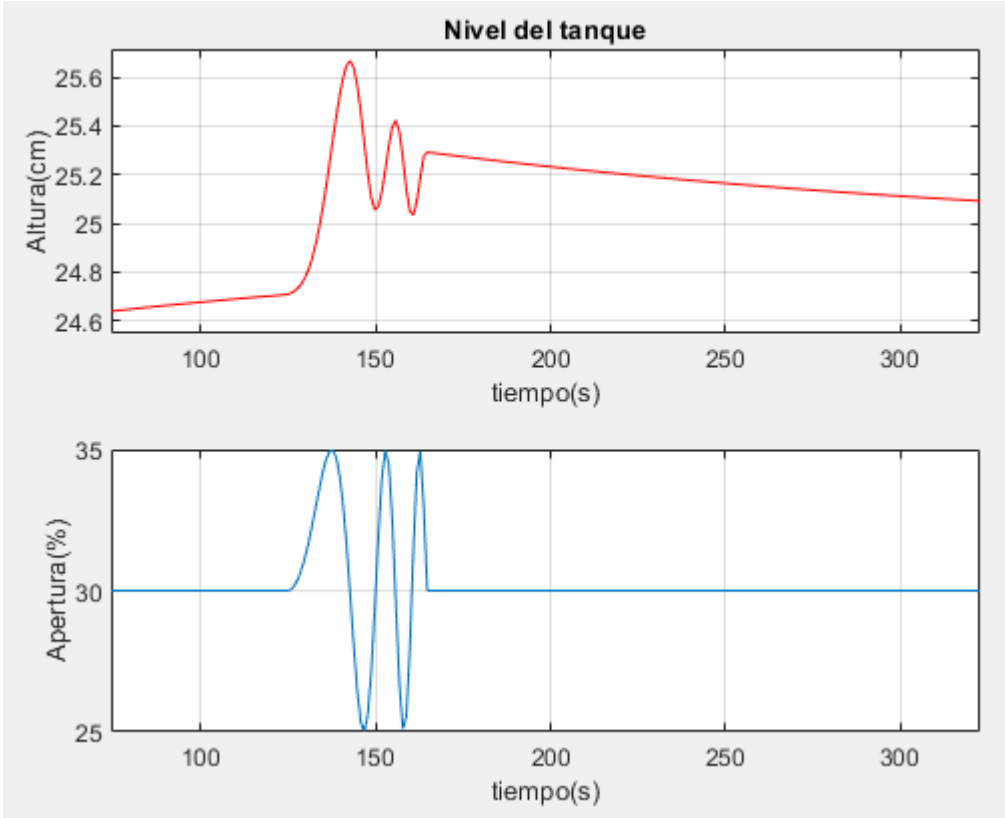


Figura 4.5 – Prueba de botón chirp en EJSS sobre entrada VR1.

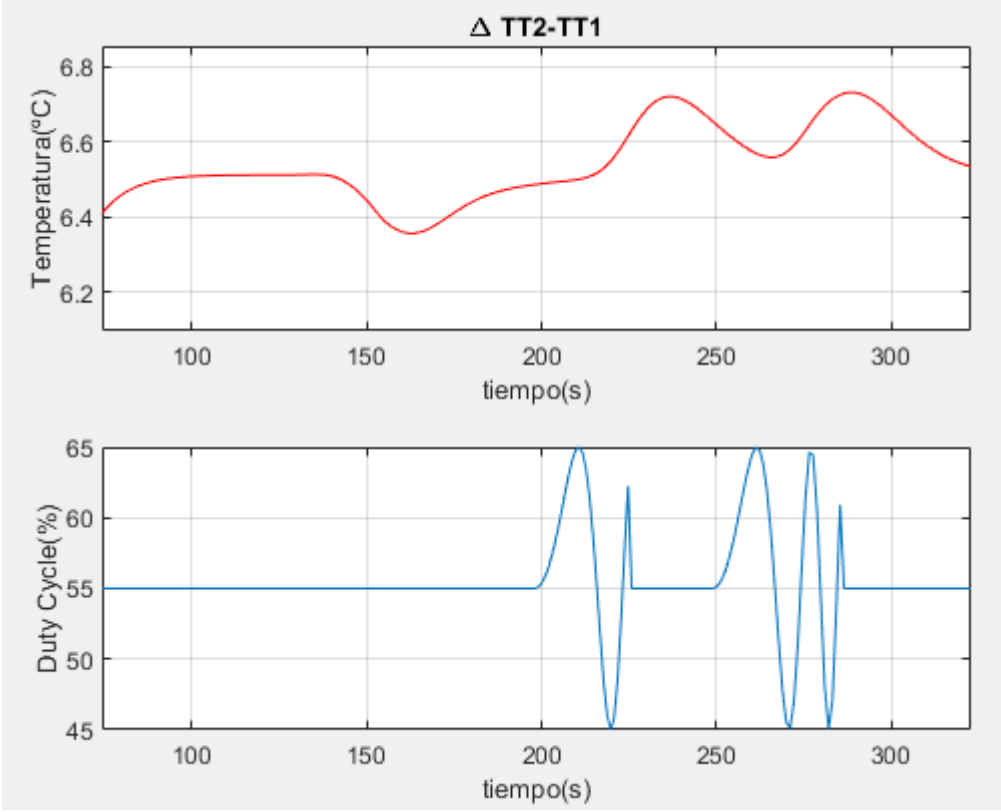


Figura 4.6 – Prueba de botón chirp en EJSS sobre entrada R.

También podemos observar en las figuras 4.5 y 4.6 qué ocurre cuando el usuario decide detener la señal chirp

mediante los botones (antes de que la señal chirp alcance el tiempo establecido en su duración).

4.2.2 Pestaña Control.

En la pestaña de control se deben validar las siguientes funcionalidades:

- Transición entre modos manual/automático.
- Control descentralizado de ambos controladores.
- Funcionamiento de la opción anti-Windup de ambos controladores.

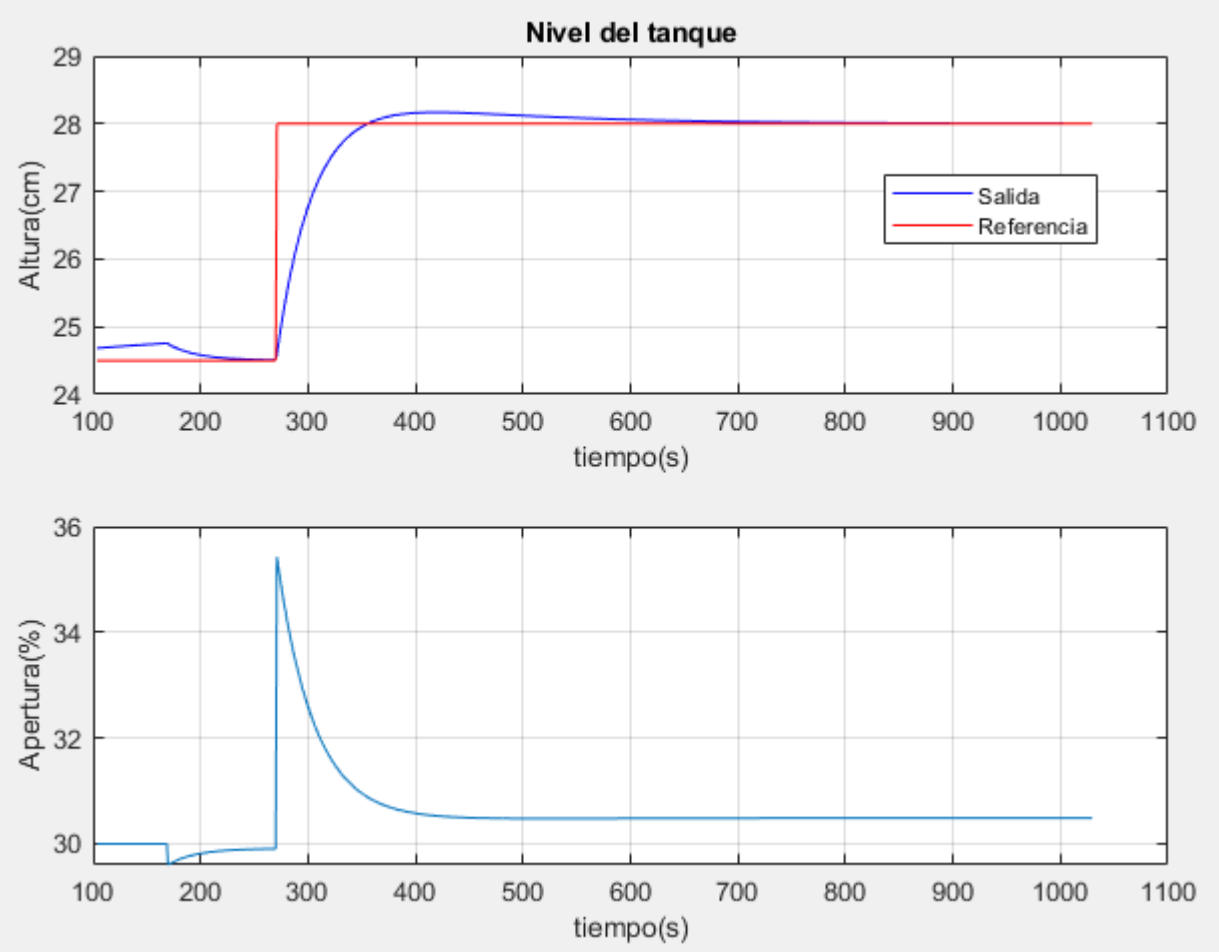


Figura 4.7 – Prueba de controlador de nivel en EJSS y cambio de manual a automático.

En las figuras 4.7 y 4.8 se puede apreciar la transición entre los modos manual/automático en ambos controladores, además de que los controladores son capaces de realizar su función correctamente.

Mejorar el comportamiento de éstos es cuestión del diseño de los parámetros de los mismos.

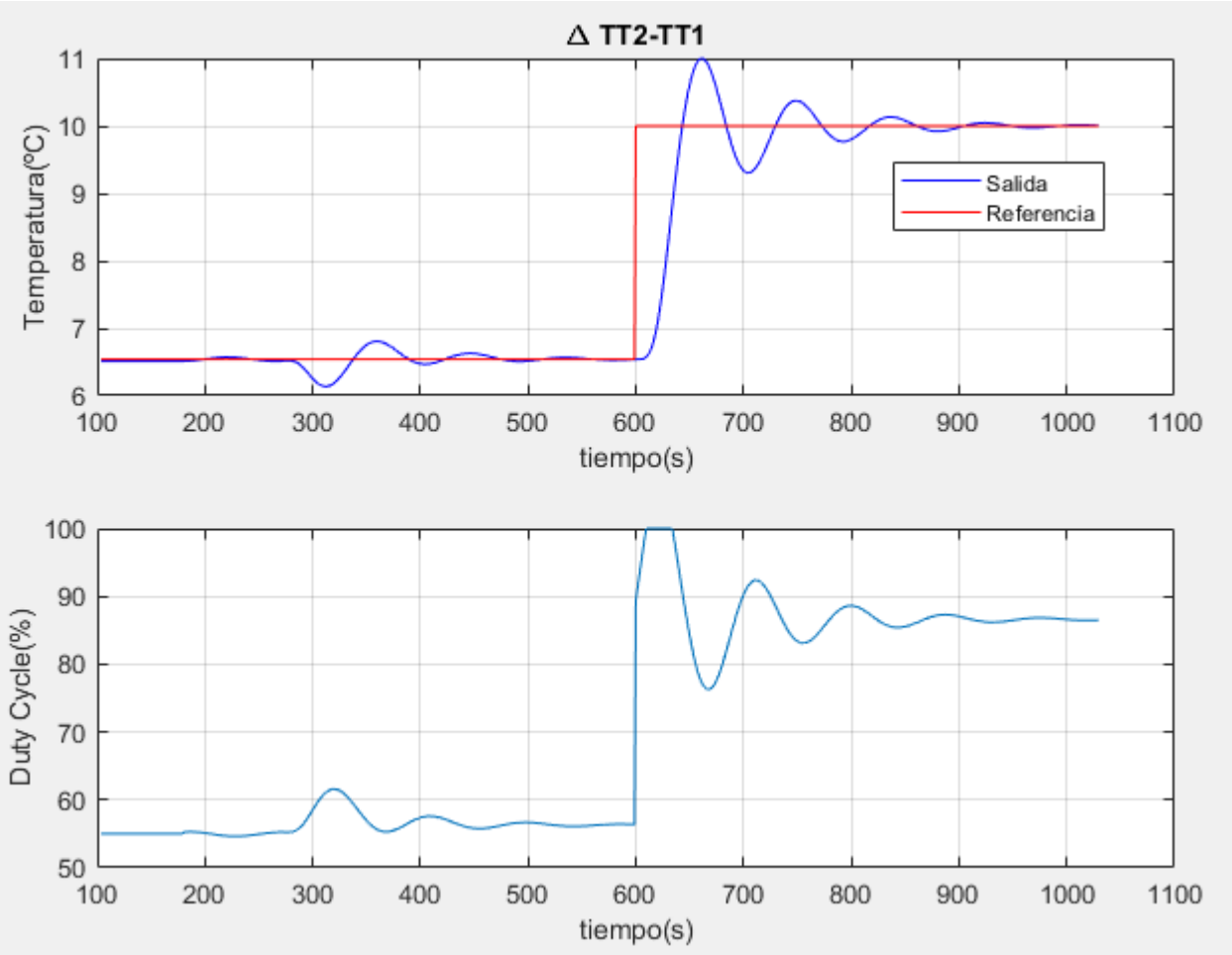


Figura 4.8 – Prueba de controlador de temperatura en EJSS y cambio de manual a automático.

En la figura 4.9 se puede apreciar en el control de temperatura (por ejemplo) qué ocurre cuando se establece una referencia no alcanzable por el controlador cuando éste no tiene la opción de anti-Windup activada. Se observa que el controlador mantiene durante un tiempo la señal de control al máximo, lo que impide que el comportamiento sea adecuado cuando se pide una referencia menor (7ºC).

Por otro lado, cuando la opción de anti-Windup sí está activada ocurre lo que muestra la figura 4.10. Como vemos, cuando se coloca una referencia de 7ªC después de una referencia no alcanzable, la señal de control reacciona inmediatamente, sin quedarse saturada como sí ocurría en el experimento de la figura 4.9.

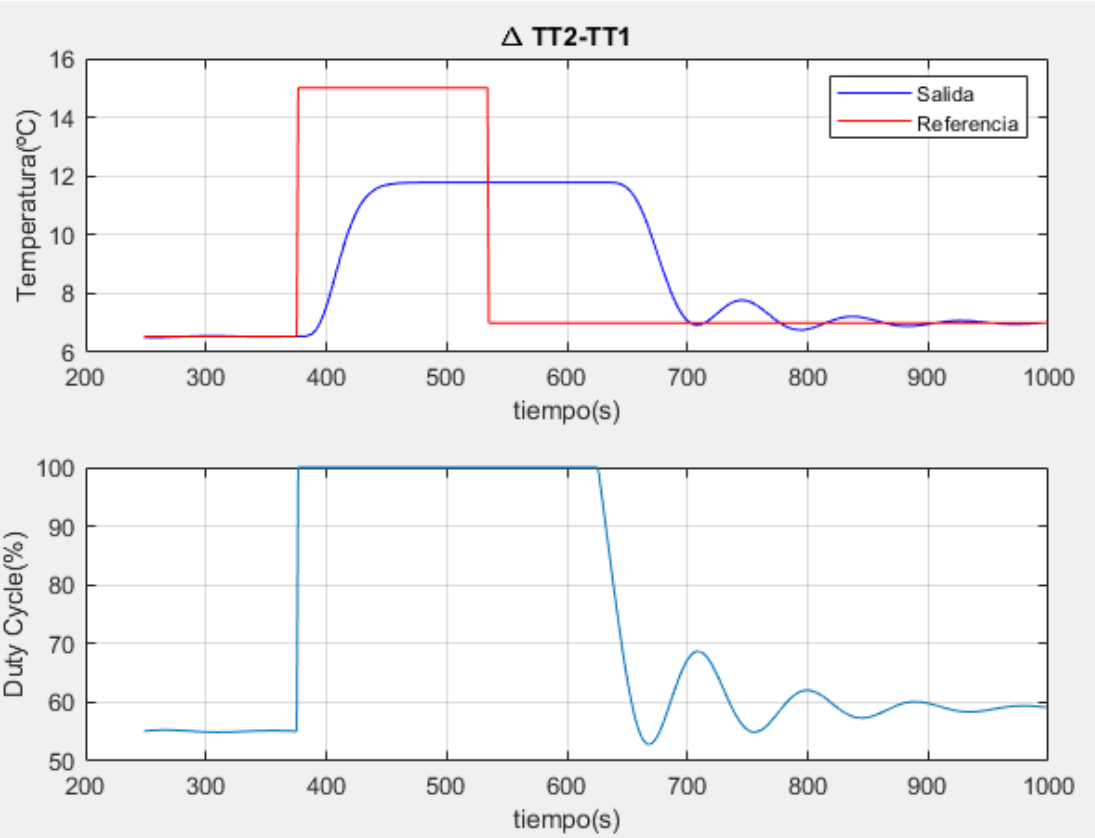


Figura 4.9 – Prueba de controlador de temperatura en EJSS sin anti-WindUp.

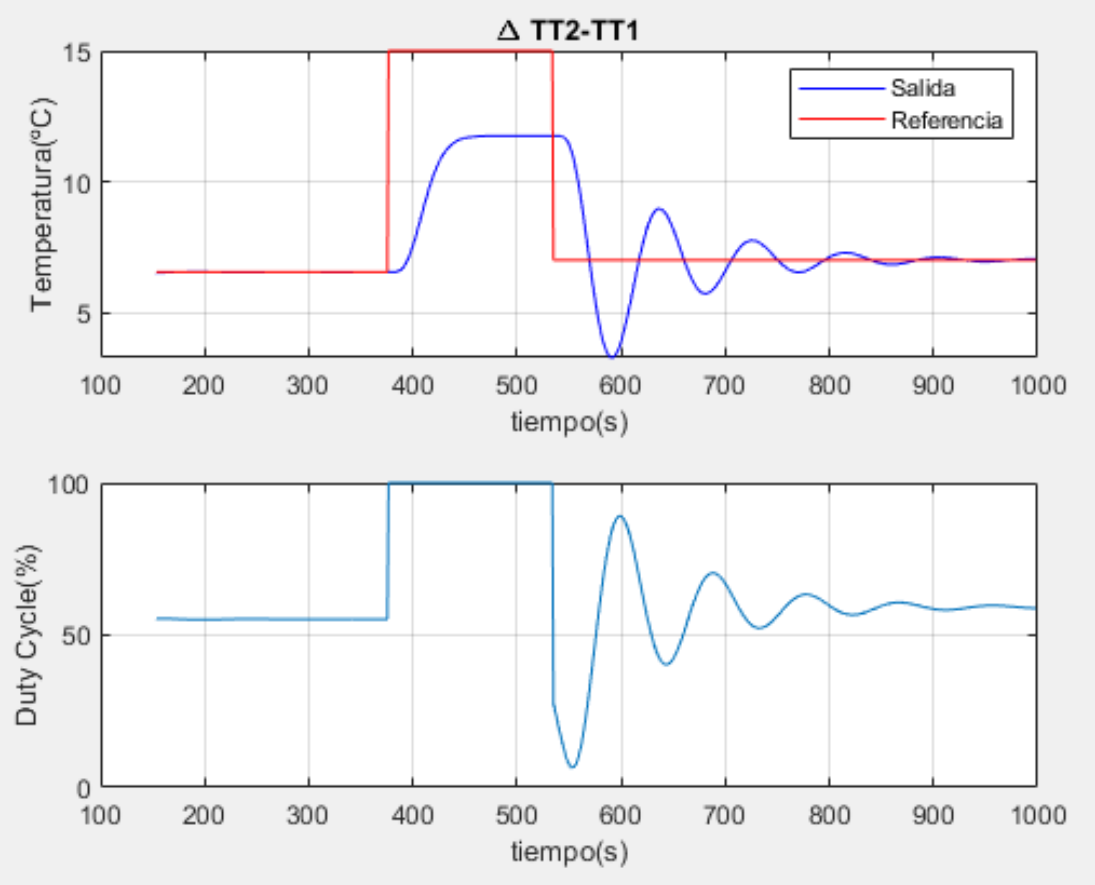


Figura 4.10 – Prueba de controlador de temperatura en EJSS con anti-WindUp.

5 Modo Remoto sobre Modelos en LabView: Conexión de EJSS con LabView

Este capítulo se centra en detallar cómo se ha realizado la conexión entre ambos programas. Es importante comprender qué función cumple cada programa. Esto queda detallado en el esquema que se muestra a continuación:

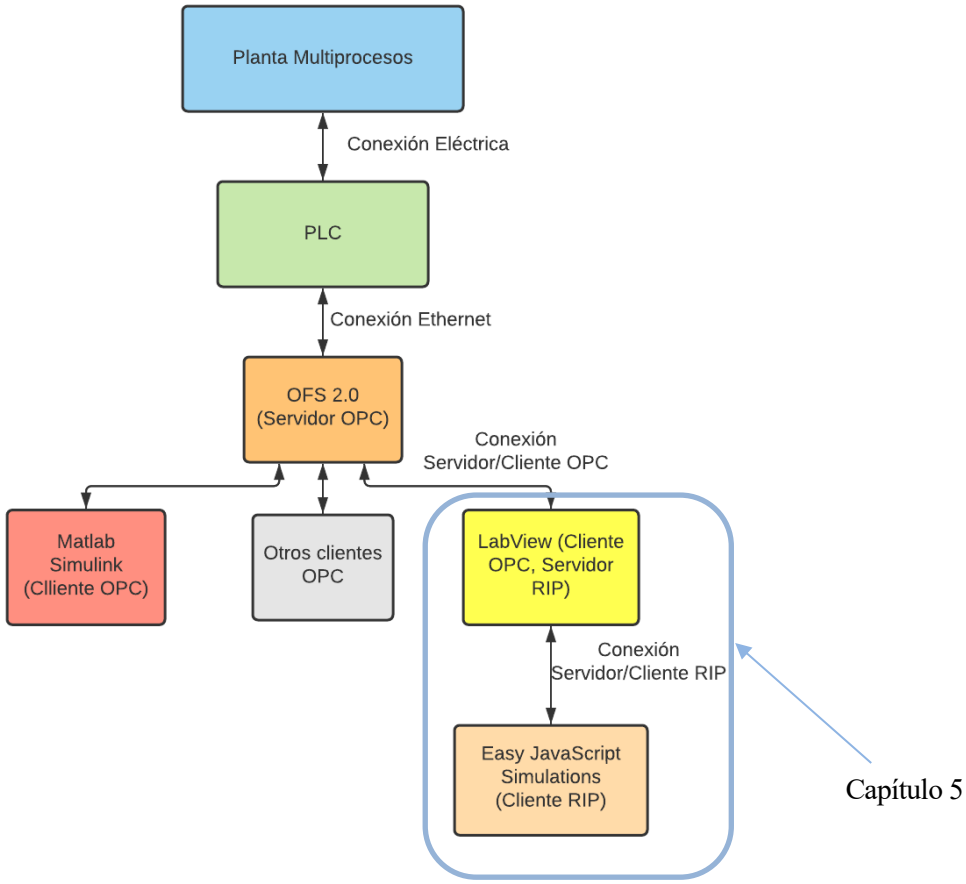


Figura 5.1 – Esquema jerárquico de conexiones, incluyendo EJSS.

Es importante destacar que, en este proyecto, diferencia del anterior, LabView cumple tanto la función de cliente OPC, como de servidor para la conexión mediante el protocolo RIP.

5.1 Remote Interoperability Protocol (RIP).

Remote Interoperability Protocol o RIP es un protocolo de comunicación desarrollado para comunicar EJSS con un servidor que puede ser bien LabView o Python, el cual debe obtener los datos experimentales para enviarlos al cliente EJSS.

Es importante destacar que es necesario disponer de, al menos, la versión de 2015 de LabView.

5.1.1 Servidor EJSS: Configuración de LabView.

Lo primero que se debe realizar para establecer la conexión es descargar el servidor sobre LabView. Éste se puede encontrar en el siguiente enlace: https://github.com/UNEDLabs/rip-labview-server

Una vez descargado, se debe descomprimir el .zip para obtener una carpeta llamada "rip-labview-server-master". Dentro de ella se encuentran los archivos y carpetas que se muestran en la figura 5.2.

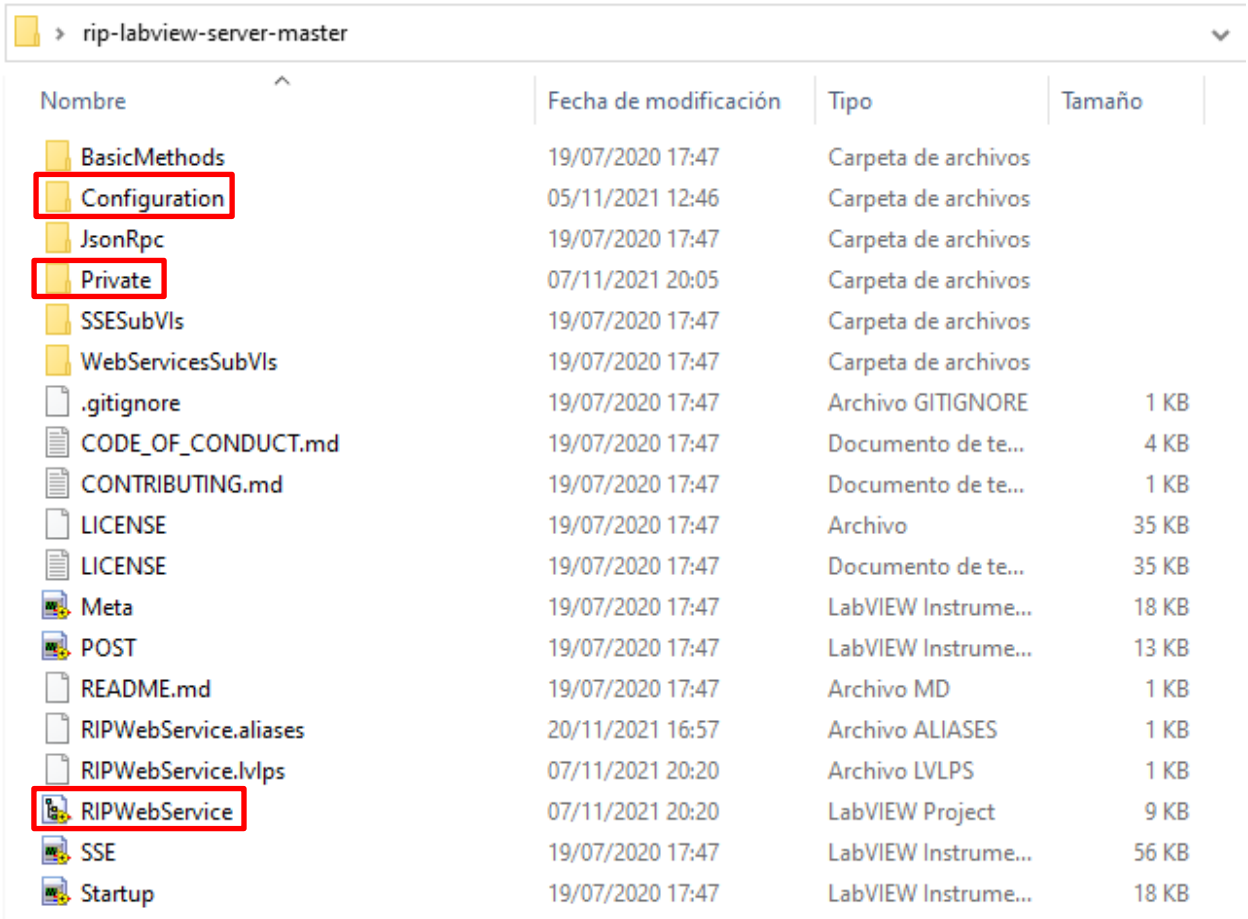


Figura 5.2 – Carpeta rip-labview-server-master.

Dentro de dicha carpeta, son de gran importancia los directorios "Private" y "Configuration". También nos interesa el ejecutable "RIPWebService".

5.1.1.1 Directorio Private.

En el directorio Private se deben colocar todos los .vi que queremos que el servidor ponga a disposición del cliente. En nuestro caso, se han colocado los archivos de LabView del Trabajo de Fin de Grado basado en modelos. Ha sido necesario modificarlos para que el programa sea totalmente compatible con el servidor, pues no detecta las variables situadas en las pestañas creadas en LabView.

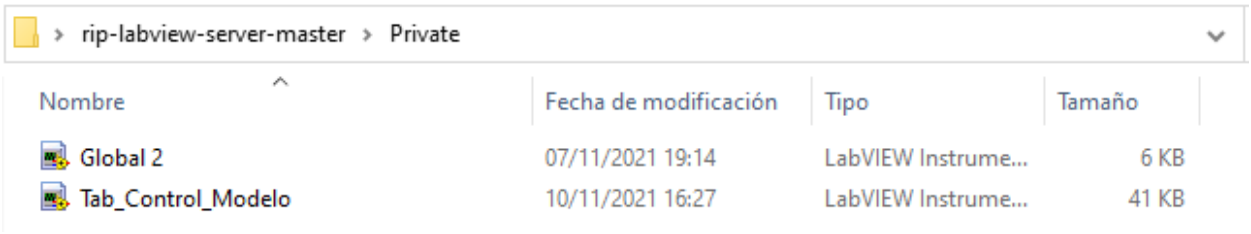


Figura 5.3 – Directorio Private, dentro de rip-labview-server-master.

5.1.1.2 Directorio Configuration.

Dentro de este directorio se puede encontrar un archivo en LabView llamado "Global_Configurations".

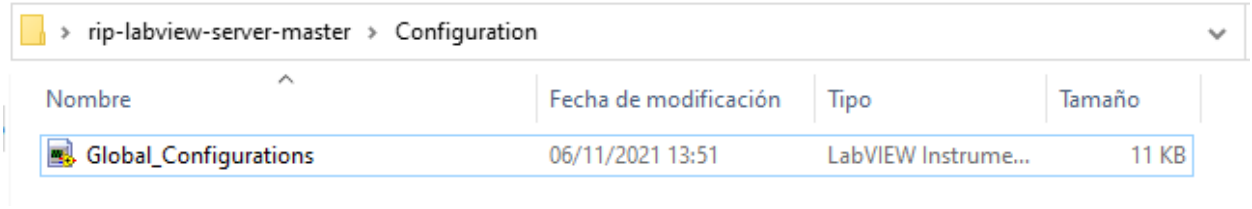


Figura 5.4 – Directorio Configuration, dentro de rip-labview-server-master.

Si lo abrimos, se obsevamos que su función es configurar cada uno de los .vi incluidos en el directorio Private. Para este proyecto, el programa ha sido modificado tal y como se muestra en la figura 5.5.

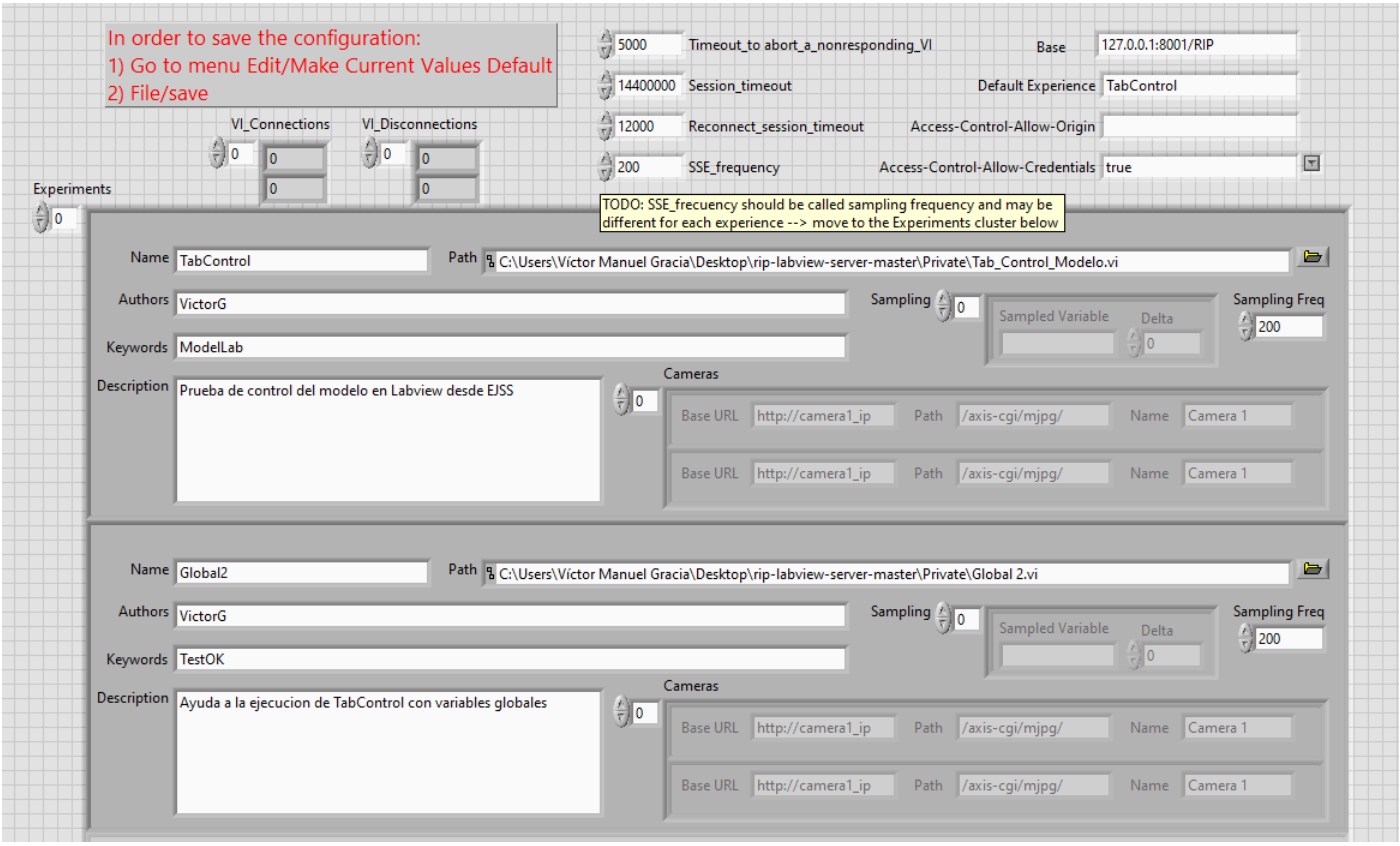


Figura 5.5 – Programa Global_Configurations.vi.

Por último, es necesario ejecutar el servidor. Para ello, abrimos la aplicación RIPWebService.

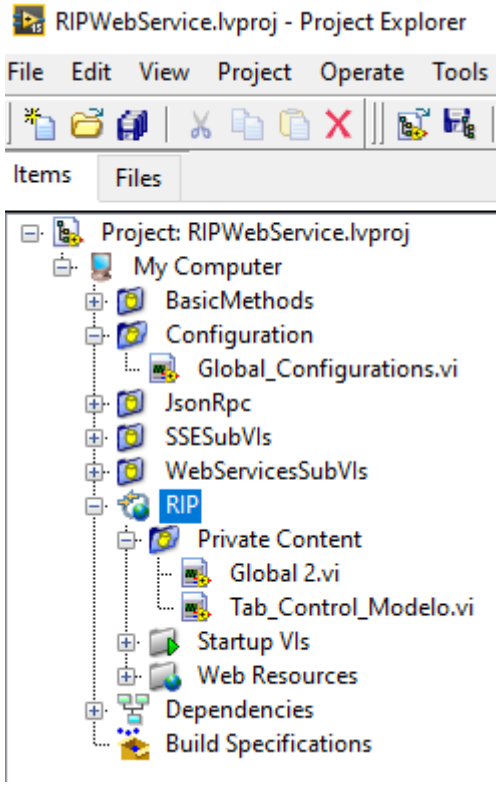


Figura 5.6 – RIPWebService.

En la figura 5.6 se aprecian todas las carpetas que se veían en la figura 5.2, incluidos los directorios Private y Configuration. Para poner en marcha el servidor, basta con hacer click derecho sobre RIP, y hacer click sobre Start.

Una vez hecho esto, nos debe aparecer un mensaje como el de la figura 5.7. Éste nos indica que el servidor se está ejecuntado sobre el puerto 8001, por lo que al conectarnos desde el cliente debemos tenerlo en cuenta.

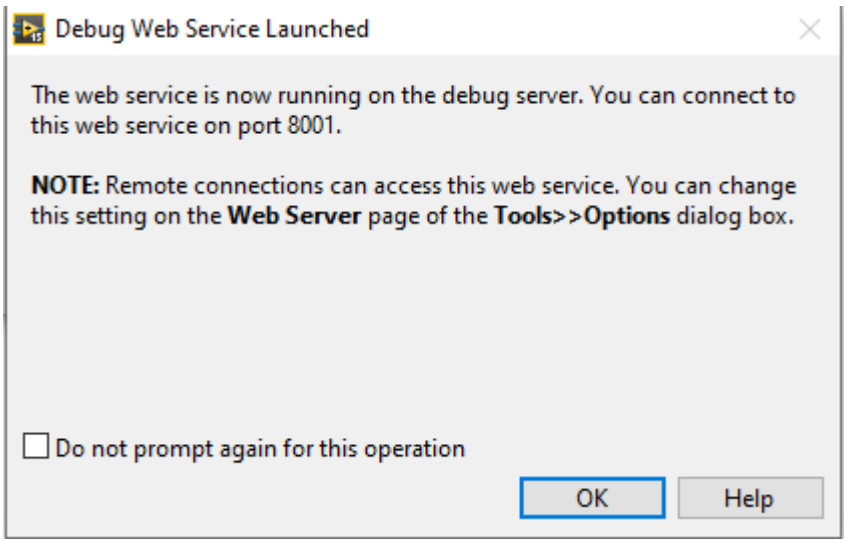


Figura 5.7 – Mensaje de inicio del servidor en LabView.

Una vez realizados estos pasos, el servidor está en funcionamiento y disponible para su conexión con clientes.

5.1.2 Cliente EJSS: Elemento RIP.

Para la conexión del cliente es necesario crear en EJSS lo que se conoce como "Elemento RIP". Para crearlo, nos debemos dirigirnos a la pestaña Modelo, y a la subpestaña Elementos. Dentro de ésta ser debe hacer doble click sobre "Software Links". Esto se muestra en la figura 5.8.

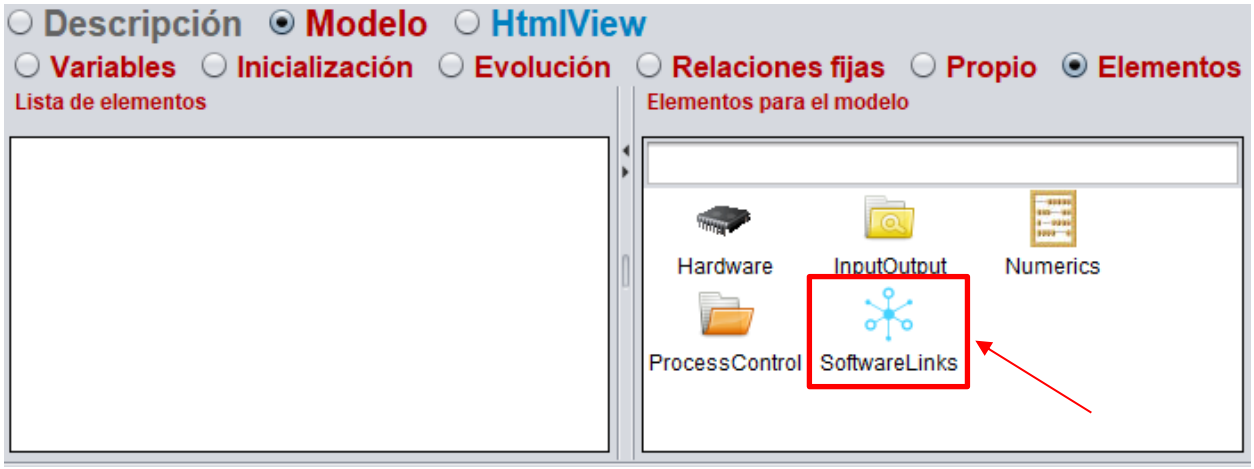


Figura 5.8 – Software Links de EJSS, en Modelo/Elementos.

Como se puede apreciar en la figura 5.9, dentro de Software Links se encuentra el Elemento RIP. Debemos arrastrarlo hacia la lista de elementos que se observa a la izquierda de la figura.

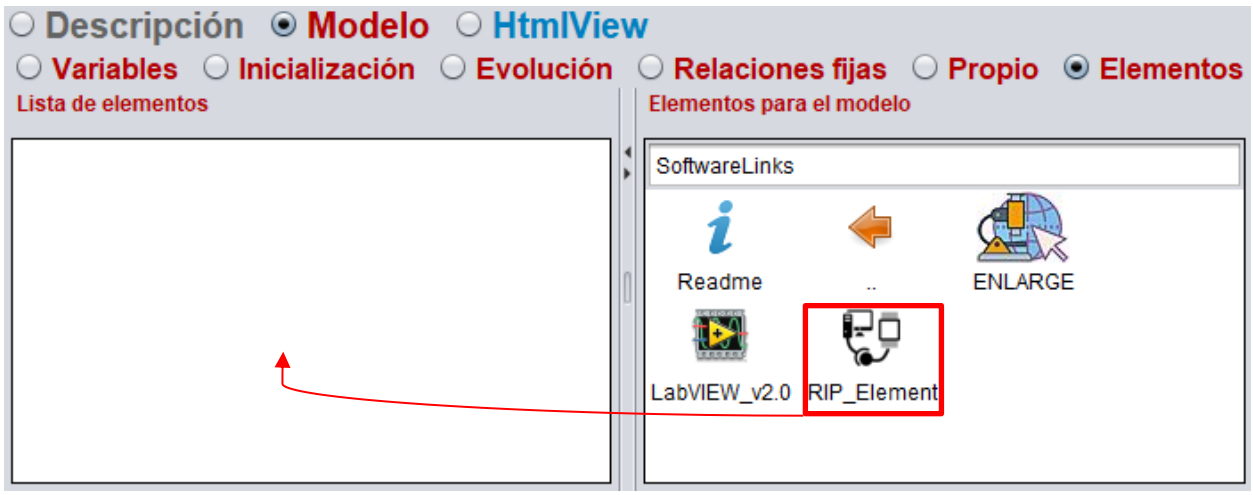


Figura 5.9 – Elemento RIP de EJSS, en Modelo/Elementos/Software Links.

Lo primero que debemos hacer es darle un nombre, como se muestra en la figura 5.10. Es recomendable usar el nombre "rip", por simplicidad.

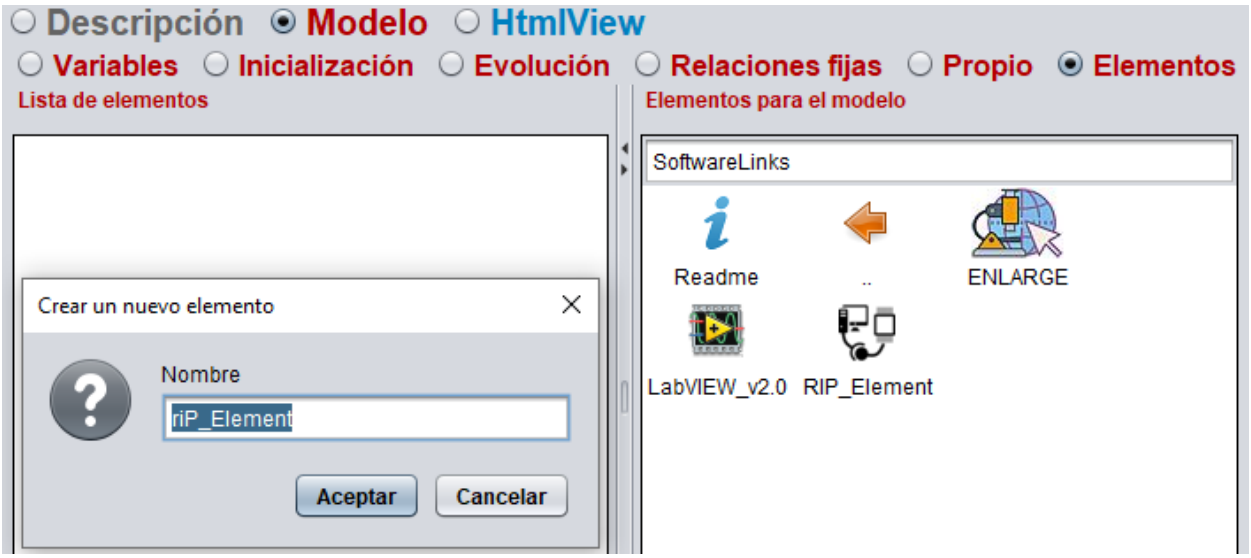


Figura 5.10 – Creación de Elemento RIP en EJSS.

Una vez creado, es necesario configurarlo para establecer la conexión. Hay que tener en cuenta que, para establecerla, el servidor debe estar corriendo previamente. En nuestro caso, el servidor siempre se halla en el puerto 8001.

La configuración se abre haciendo doble click sobre el elemento ya creado. Ésta se muestra en la figura 5.11.

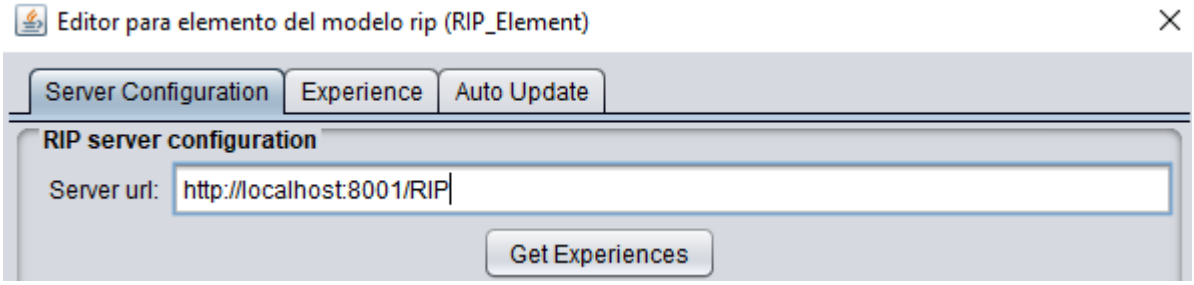


Figura 5.11 – Configuración del elemento RIP: Pestaña Server Configuration.

Tras colocar la url del servidor indicando el puerto 8001, se debe pulsar el botón “Get Experiences”. Si el servidor ha sido iniciado y configurado correctamente, debe aperecer algo similar a lo que se puede apreciar en la figura 5.12.

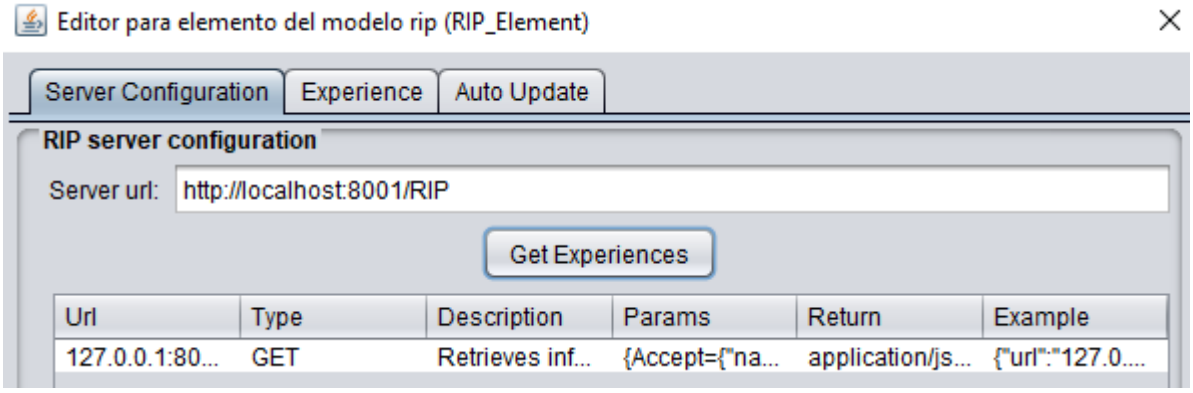


Figura 5.12 – Configuración del elemento RIP: Server Configuration tras pulsar Get Experiences.

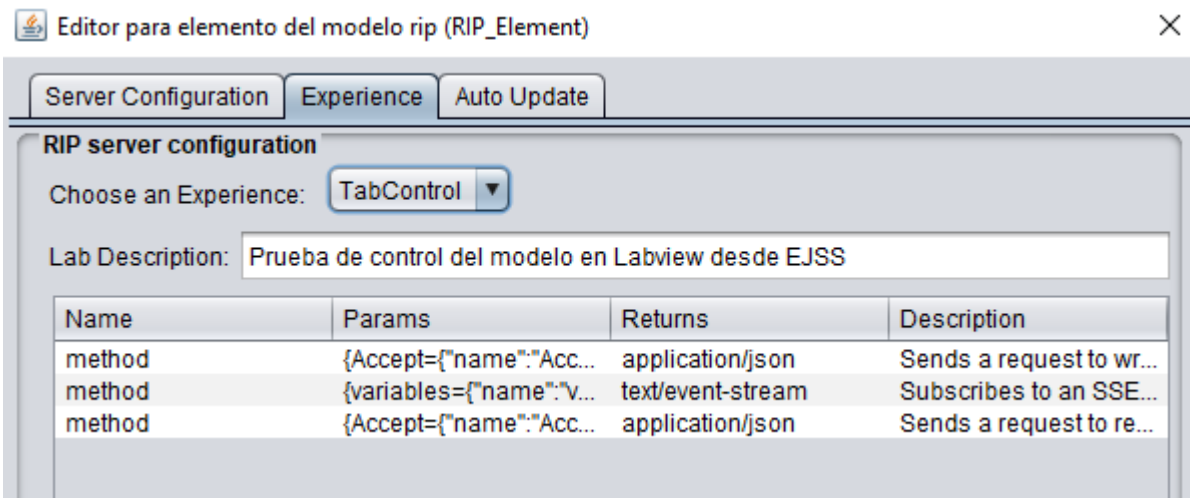


Figura 5.13 – Configuración del elemento RIP: Pestaña Experience.

Como se observa en la figura 5.13, EJSS ha detectado la experiencia "TabControl" (y también "Global2"). Además, se puede ver la descripción que se introdujo en la configuración (figura 5.5).

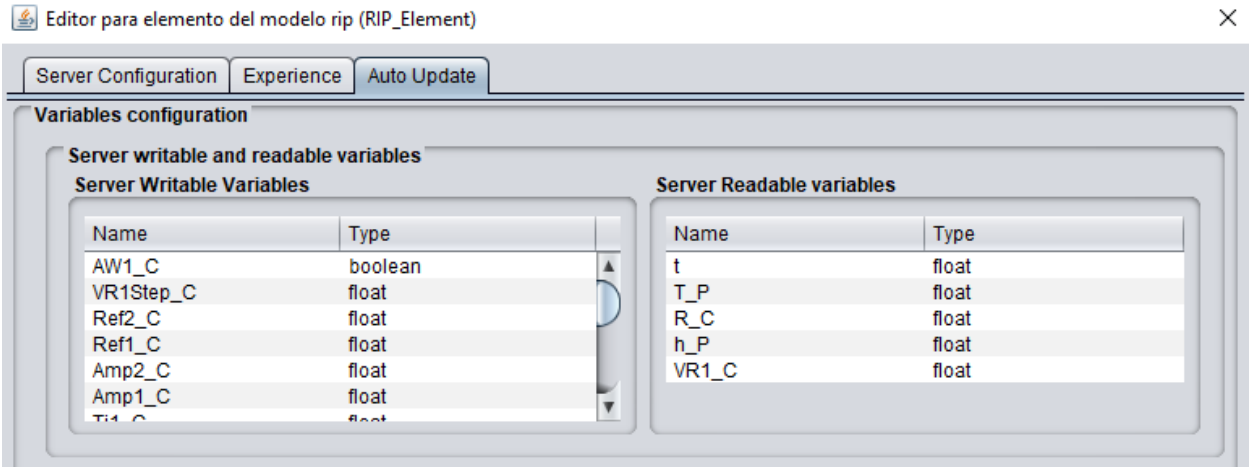


Figura 5.14 – Configuración del elemento RIP: Pestaña Auto Update.

En la figura 5.14 se observan todas las variables del .vi seleccionado en la experiencia. A la izquierda se aprecian las variables de escritura, mientras que a la derecha se observan las de lectura.

Haciendo click derecho sobre cada una de las variables, se muestra la opción de conectarlas con variables de EJSS. En las figuras 5.15 y 5.16 se muestran todas las conexiones realizadas.

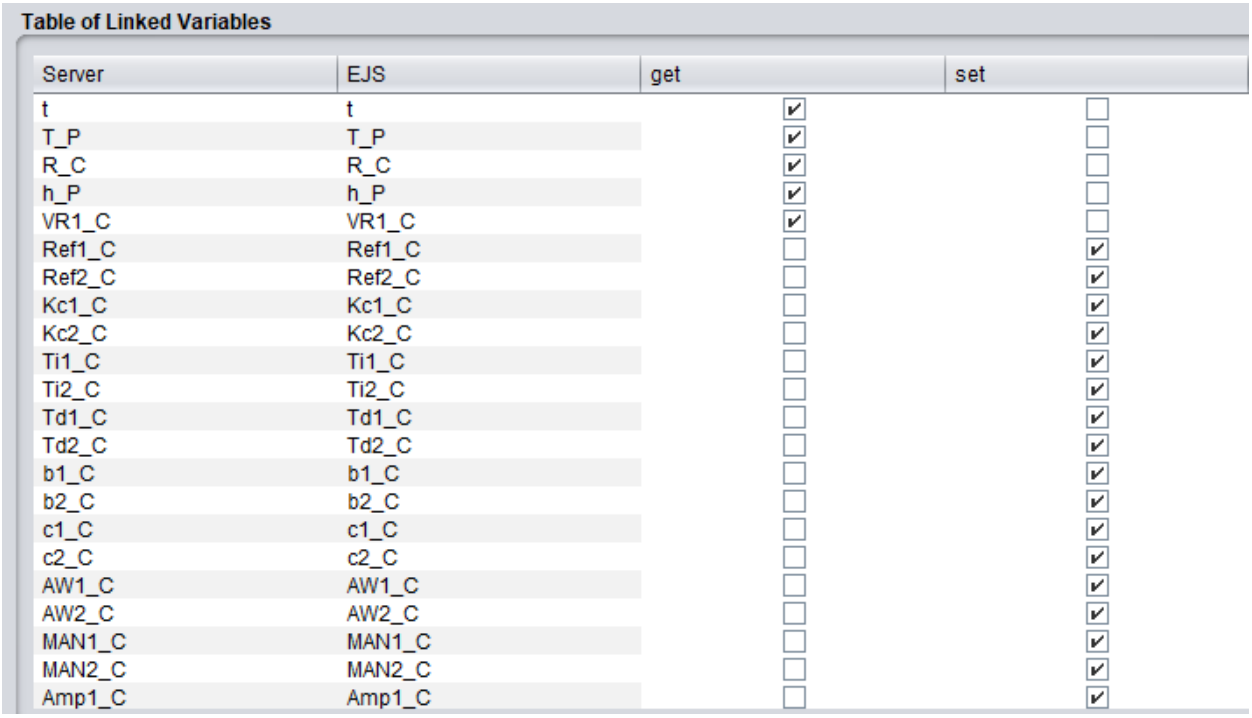
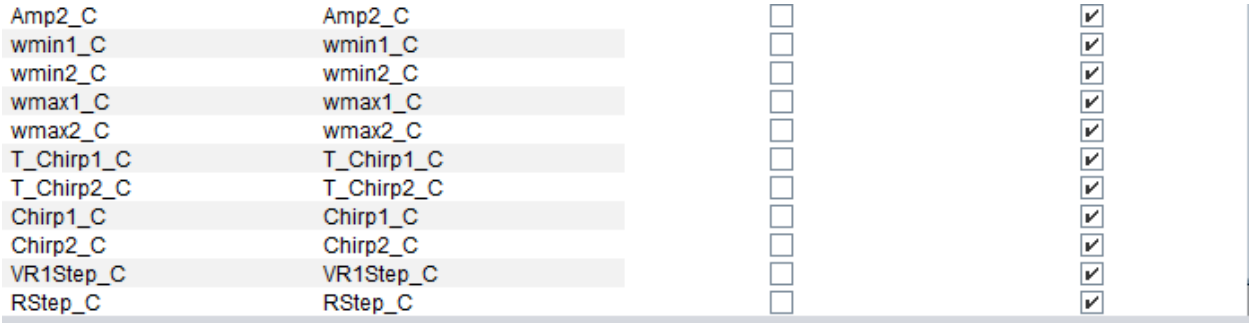


Table of Linked Variables

Server	EJS	get	set
t	t	☑	☐
T_P	T_P	☑	☐
R_C	R_C	☑	☐
h_P	h_P	☑	☐
VR1_C	VR1_C	☑	☐
Ref1_C	Ref1_C	☐	☑
Ref2_C	Ref2_C	☐	☑
Kc1_C	Kc1_C	☐	☑
Kc2_C	Kc2_C	☐	☑
Ti1_C	Ti1_C	☐	☑
Ti2_C	Ti2_C	☐	☑
Td1_C	Td1_C	☐	☑
Td2_C	Td2_C	☐	☑
b1_C	b1_C	☐	☑
b2_C	b2_C	☐	☑
c1_C	c1_C	☐	☑
c2_C	c2_C	☐	☑
AW1_C	AW1_C	☐	☑
AW2_C	AW2_C	☐	☑
MAN1_C	MAN1_C	☐	☑
MAN2_C	MAN2_C	☐	☑
Amp1_C	Amp1_C	☐	☑

Figura 5.15 – Configuración del elemento RIP: Conexión de variables (1).



Amp2_C	Amp2_C	☐	☑
wmin1_C	wmin1_C	☐	☑
wmin2_C	wmin2_C	☐	☑
wmax1_C	wmax1_C	☐	☑
wmax2_C	wmax2_C	☐	☑
T_Chirp1_C	T_Chirp1_C	☐	☑
T_Chirp2_C	T_Chirp2_C	☐	☑
Chirp1_C	Chirp1_C	☐	☑
Chirp2_C	Chirp2_C	☐	☑
VR1Step_C	VR1Step_C	☐	☑
RStep_C	RStep_C	☐	☑

Figura 5.16 – Configuración del elemento RIP: Conexión de variables (2).

5.2 Adaptación de LabView basado en modelos para la conexión con EJSS.

Como se puede comprobar en el apartado 1.5 de este documento, el programa en LabView que implementaba modelos del sistema presenta una iterfaz gráfica que puede ser utilizada por el usuario.

Sin embargo, en este capítulo pretendemos utilizar y controlar los modelos allí implementados desde el navegador web a partir de EJSS. Debido a esto, no son necesarias las pantallas de explotación en LabView, ni una interfaz gráfica utilizable por el usuario puesto que LabView será transparente al usuario final.

Debemos destacar que esta implementación (control de modelos de LabView) ha sido realizada con la mera función de comprobar la conexión de EJSS con Labview. Por otro lado, también facilita la conexión con la planta real al darnos una idea de cómo adaptar el LabView basado en la planta real.

En conclusión, el usuario dispondrá de un archivo en EJSS autónomo y funcional basado en los modelos de parámetros distribuidos, y otro archivo en EJSS que se conectará con el LabView de la planta real, pero no dispondrá del archivo en EJSS para conectarse con el LabView basado en modelos, ya que se trata de una prueba de desarrollo.

En la figura 5.17 se muestra el programa adapatado. Como vemos, el nombre de las variables ha sido modificado para que sea el mismo que el de las variables en EJSS. Se ha realizado de esta forma para evitar confusiones.

Por otro lado, también se puede apreciar que se han eliminado las pestañas, puesto que impiden que sean reconocidas por el cliente en EJSS. Además, se han eliminado las gráficas para evitar una carga computacional innecesaria.

Otra de las diferencias es que se ha adaptado el anti-WindUp para que sea opcional, ya que en el proyecto anterior no lo era. Tambíen se ha acondicionado el programa para que existan dos señales Chirp independientes, pues antes solo había una señal que se compartía con ambas entradas.

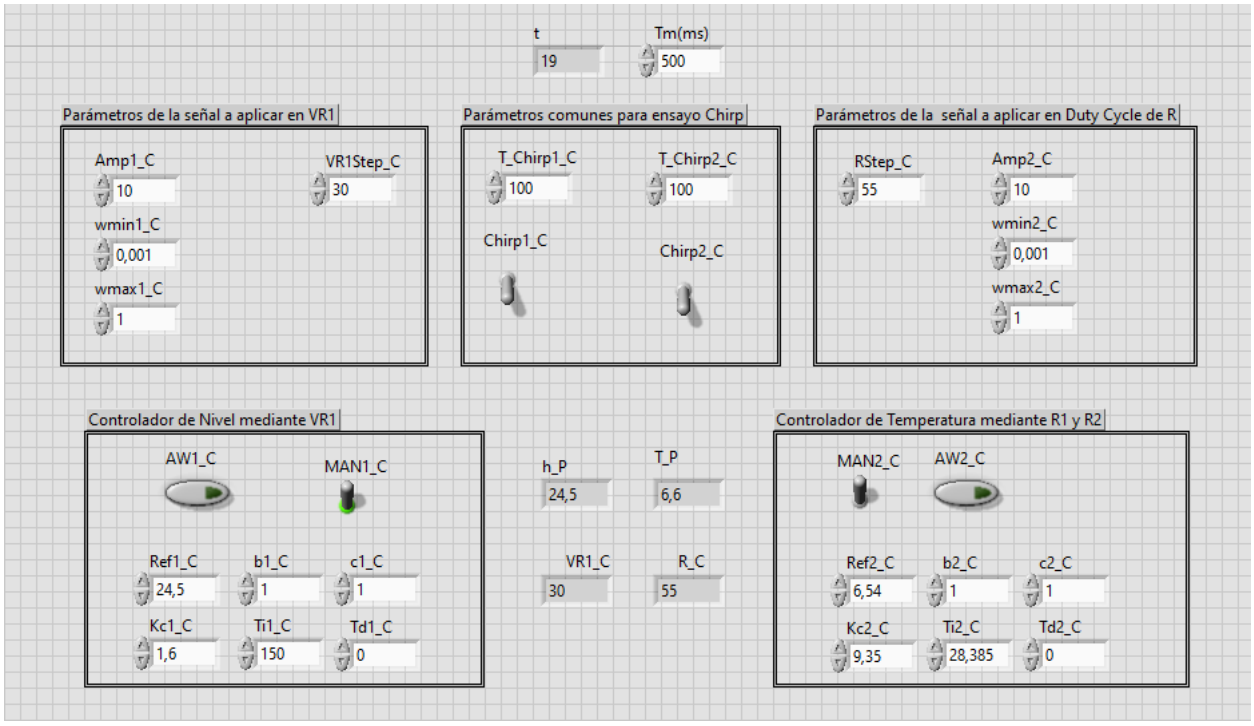


Figura 5.17 – Adaptación del programa en LabView basado en modelos.

5.3 Pruebas sobre EJSS con conexión a modelos en LabView.

En este apartado se realizarán pruebas sobre EJSS conectado a LabView. En las siguientes figuras se verifica el correcto funcionamiento de la conexión.

A partir de las figuras 5.18-5.21 se puede verificar que la funcionalidad de las señales en escalón, Chirp, y de ambos controladores es correcta. Además, se ha comprobado que el paso de manual a automático en ambos controladores es de la forma esperada.

Por otro lado, también se ha comprobado la funcionalidad de la opción de anti-Widup, aunque debido a la duración de las pruebas las cuales son en tiempo real, éstas han sido omitidas.

Por último, es conveniente remarcar que todos los datos que se muestran en las figuras mencionadas han sido generados en LabView, aunque estos se recogen desde EJSS el cual cumple la mera función de interfaz HMI (SCADA).

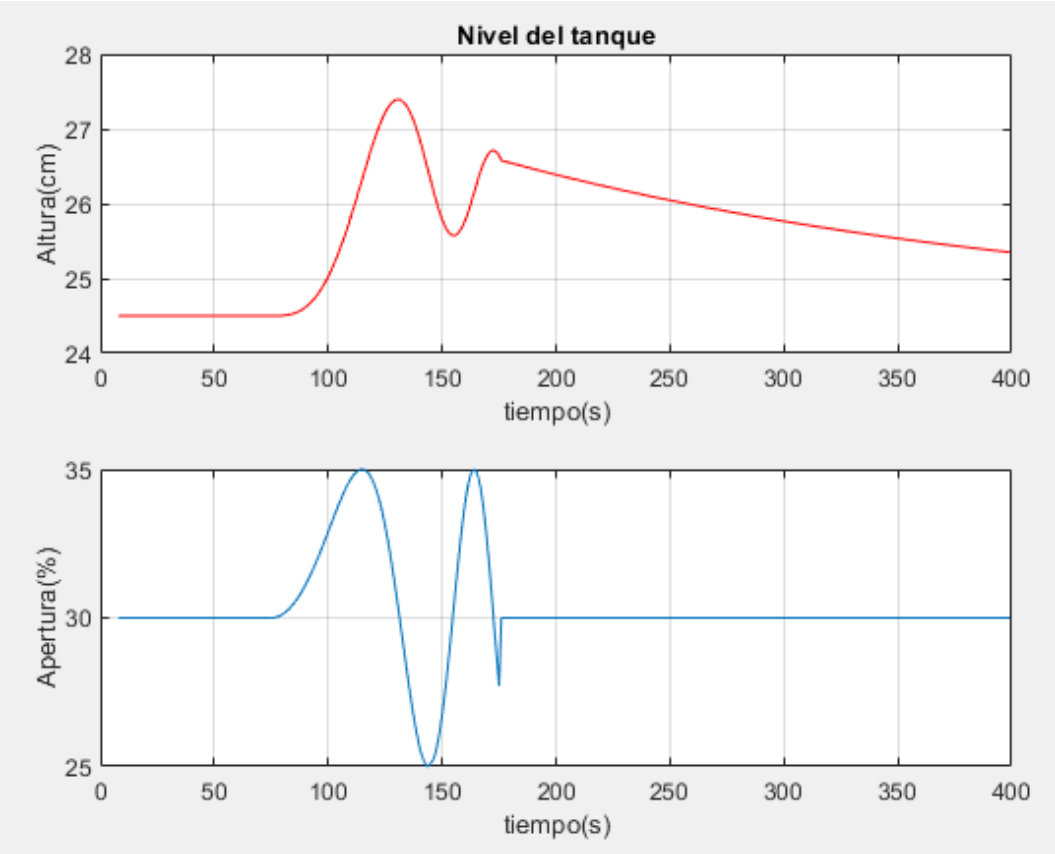


Figura 5.18 – EJSS conectado con LabView. Prueba de señal chirp sobre VR1.

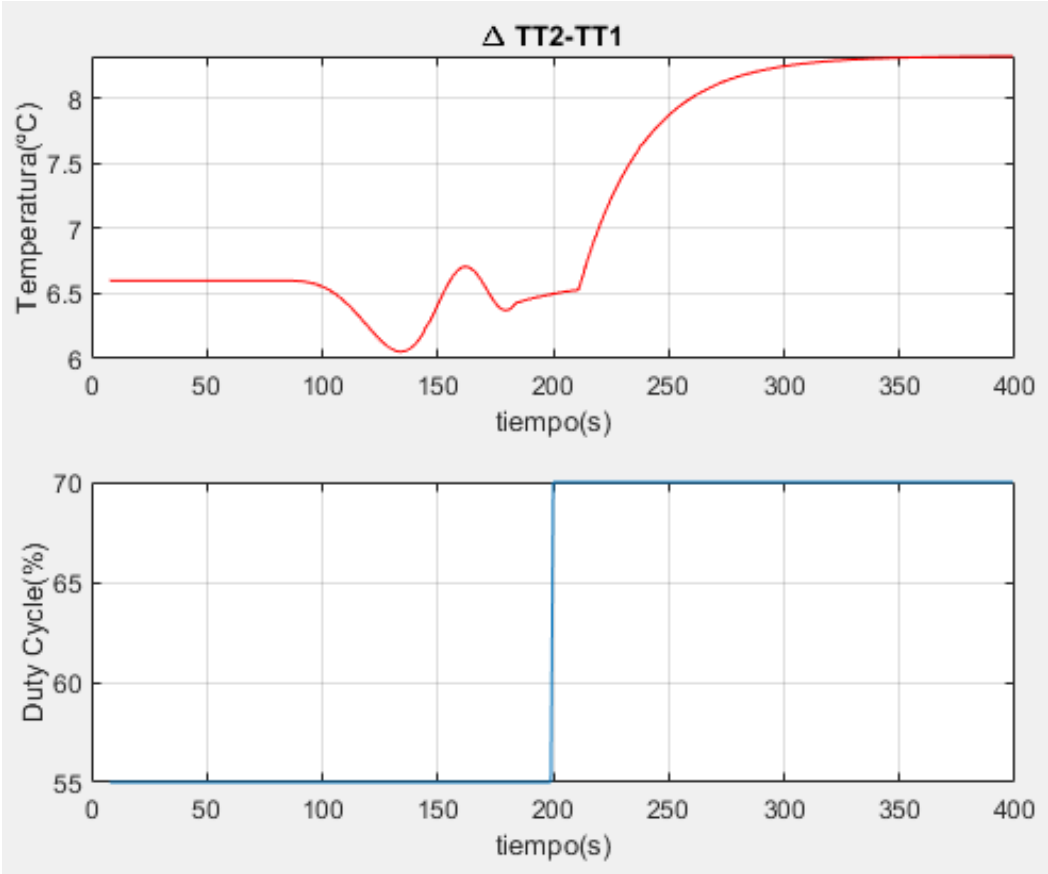


Figura 5.19 – EJSS conectado con LabView. Prueba señal en escalón sobre R.

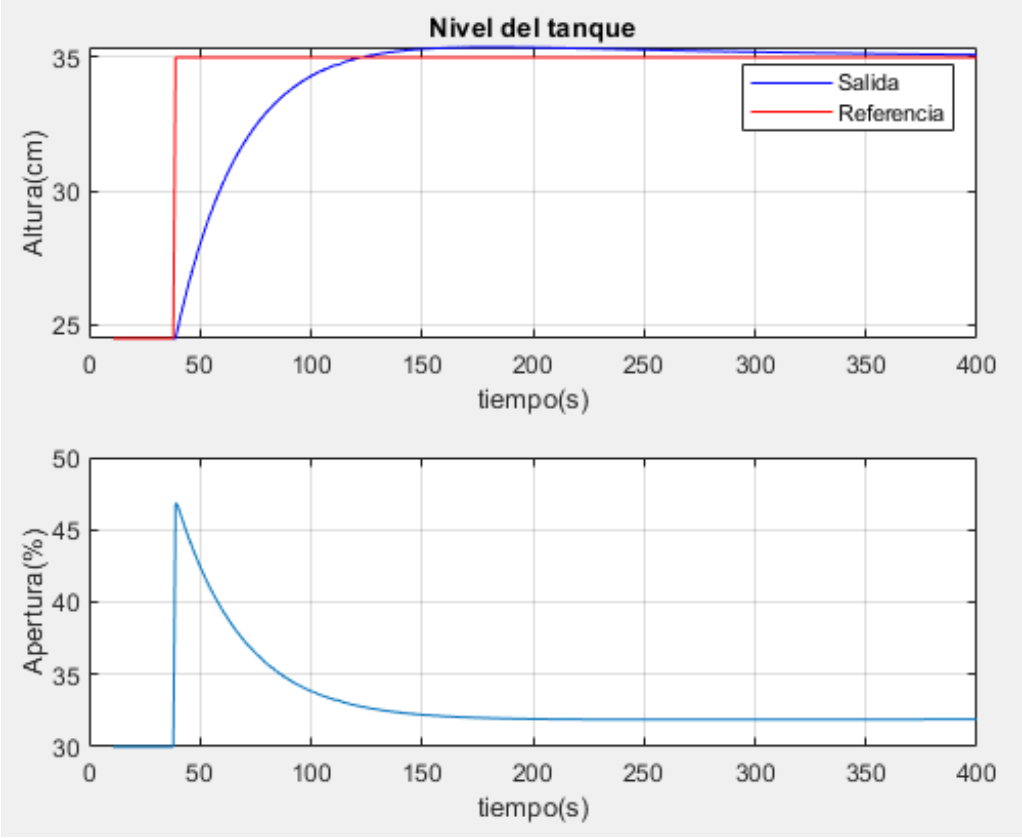


Figura 5.20 – EJSS conectado con LabView. Prueba de control de nivel.

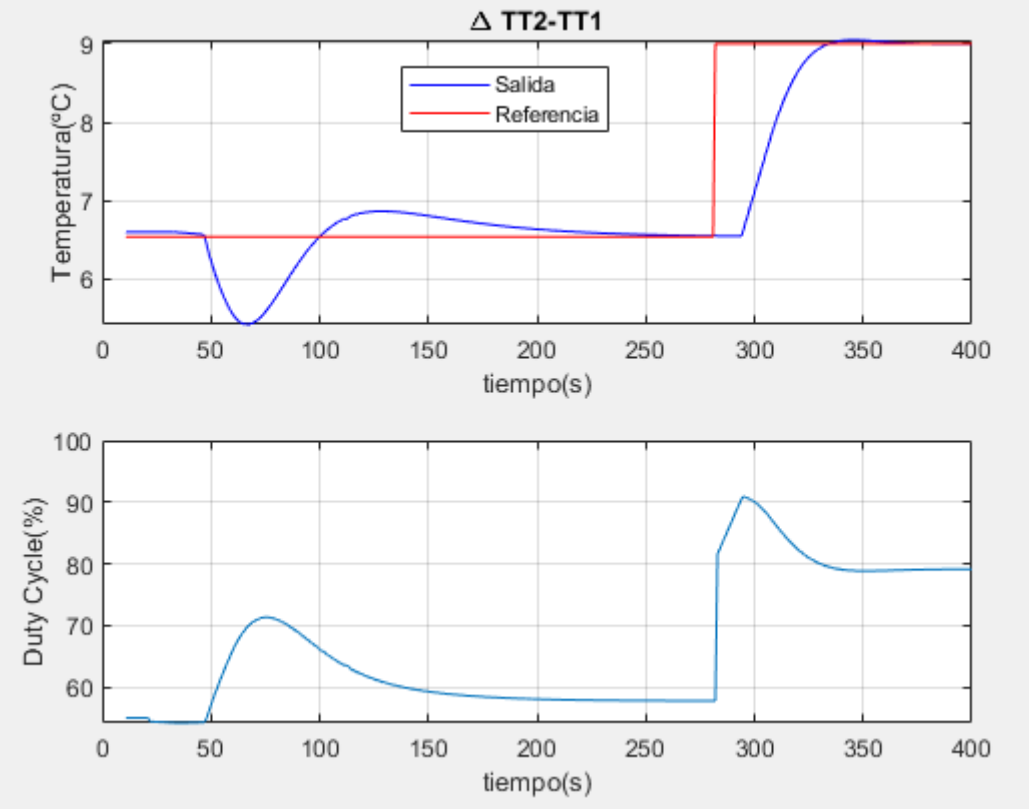


Figura 5.21 – EJSS conectado con LabView. Prueba de control de temperatura.

6 MODO REMOTO SOBRE PLANTA REAL: CONEXIÓN DE EJSS CON LABVIEW DE LA PLANTA REAL

6.1 Adaptación de LabView de la planta para la conexión con EJSS.

Al igual que en el apartado 5.2, el programa en LabView ha sido modificado para ser conectado con EJSS. A diferencia de lo que en dicho apartado se comentaba, éste programa sí que será dispuesto al usuario para que pueda conectar EJSS con la planta del laboratorio.

En primer lugar, se han eliminado las gráficas de LabView y las pestañas para que puedan detectarse todas las variables en EJSS. También se ha incluido el anti-Windup opcional y se han creado dos señales Chirp independientes para las entradas.

La diferencia con el programa del apartado 5.2 es que las entradas y salidas tienen conexión con la planta real, y la existencia de la secuencia de puesta en marcha (indicador "Ready" en la figura 6.1) que se creó para el programa conectado a la planta en el proyecto anterior.

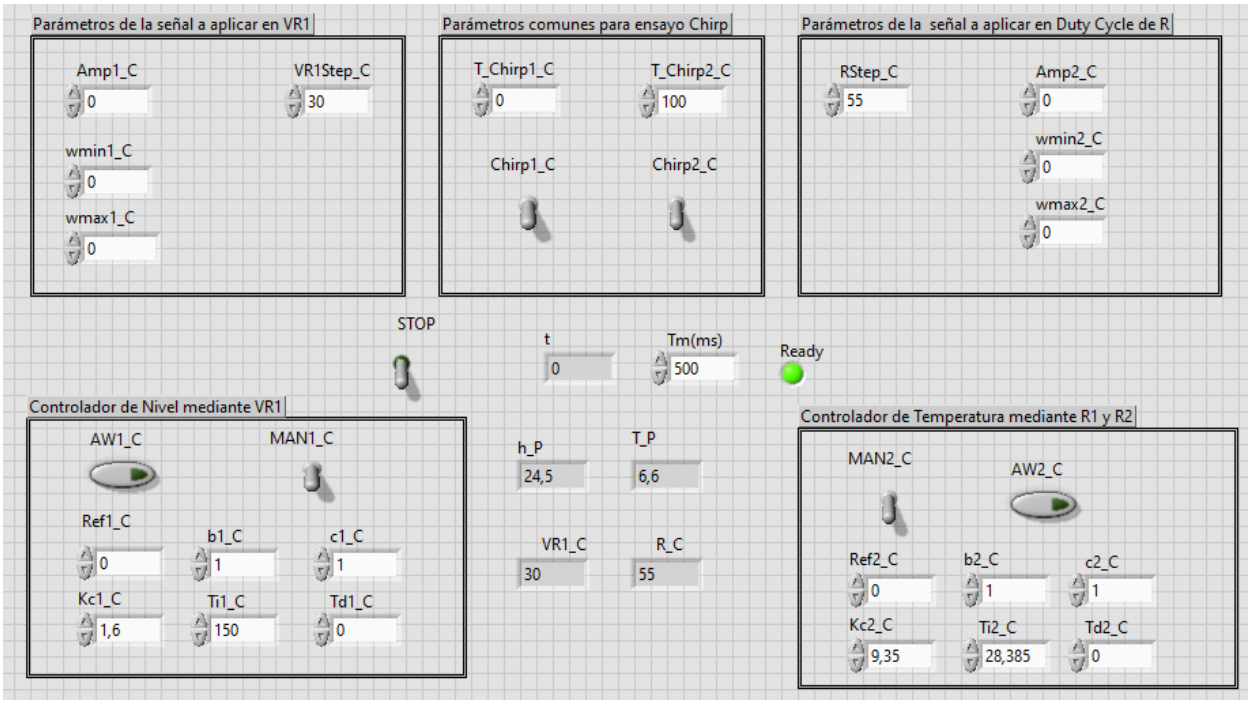


Figura 6.1 – Adaptación del programa en LabView de la planta.

6.2 Pruebas sobre EJSS con conexión a LabView de la planta real.

Debido a que la planta se encuentra en un estado no funcional, no se han podido realizar ensayos para comprobar el funcionamiento del programa. Sin embargo, se ha tenido en cuenta que las pruebas serán realizadas a posteriori, por lo que el trabajo ha sido realizado de modo que todo ha quedado preparado para que pueda probarse de forma sencilla cuando el sistema vuelva a estar operativo.

7 GUÍA DE USO DE LOS PROGRAMAS

Debido a que esta sección es una guía con vistas a ser imprimida, será un documento autocontenido, por lo que cierta información vista en capítulos anteriores será repetida.

En este capítulo se describirá cómo obtener Easy JavaScript Simulations y cómo configurarlo en el sistema operativo Windows.También se hará énfasis en cómo deben ejecutarse los programas según el modo elegido por el usuario, así como se explicará detalladamente la funcionalidad de cada uno de los elementos de las interfaces gráficas disponibles.

Por último, se indicará cómo obtener y representar los resultados de los experimentos realizados en EJSS, y se comentará cómo pueden modificarse los códigos de obtención y representación de datos. De este modo, se da la posibilidad al usuario de representar cualquier variable que desee.

7.1 Descarga y configuración de EJSS.

Para obtener el cliente de EJSS debemos entrar en el repositorio accesible mediante el siguiente enlace: https://gitlab.com/ejsS/tool

Después, hacemos click en descargar en formato .zip, como se muestra en la figura 7.1.

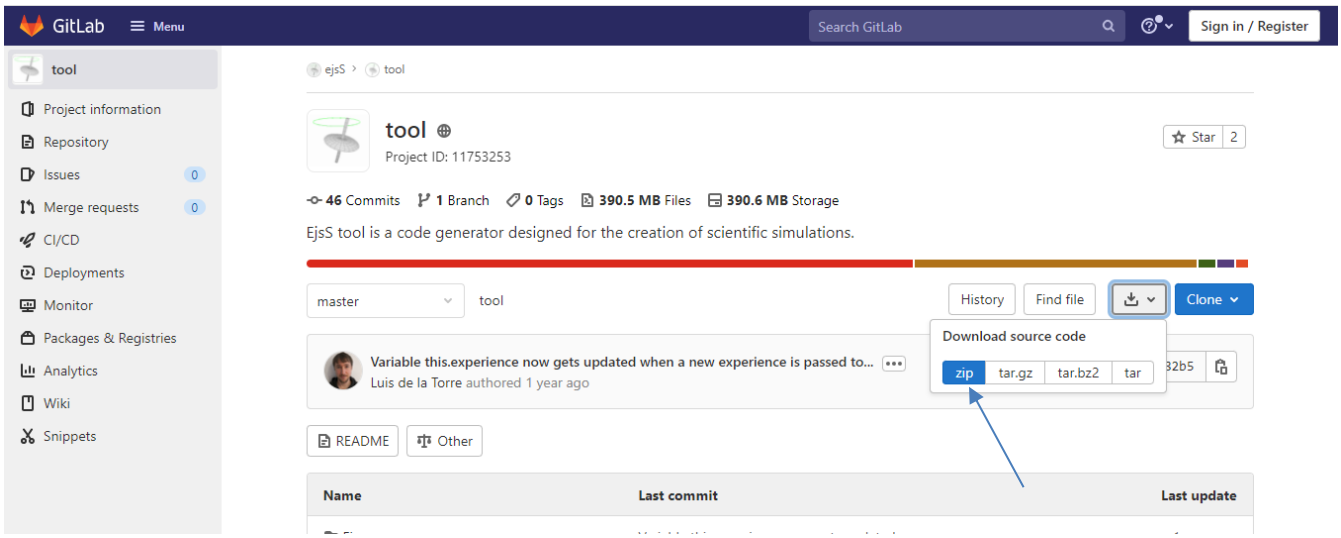


Figura 7.1 – Obtención del cliente de EJSS.

Al hacer click comenzará la descarga del archivo, llamado tool-master.zip. Tras la descarga, debemos descomprimir el .zip para obtener una carpeta con el mismo nombre.

Si entramos en la carpeta, observaremos lo que se puede apreciar en la figura 7.2.

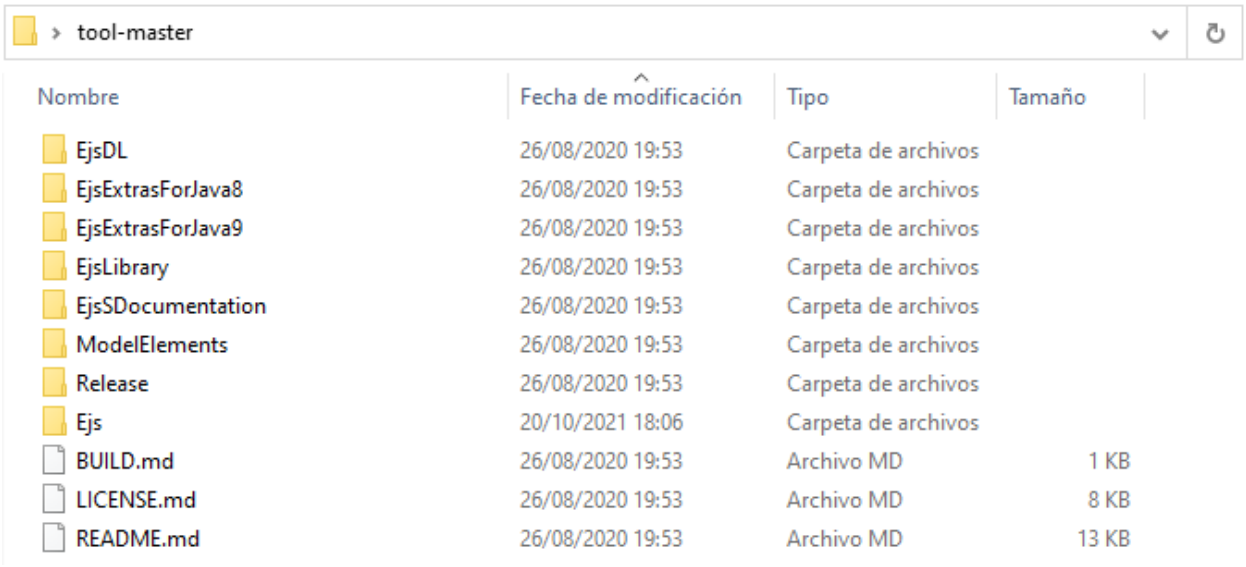


Figura 7.2 – Contenido de la carpeta tool-master.

Debemos dirigirnos ahora Ejs > distribution. Dentro encontraremos un ejecutable llamado EjsConsole. Al tratarse de un .jar, debemos tener instalado Java en nuestro ordenador.

Java puede obtenerse desde el siguiente enlace: https://www.java.com/es/download/ie_manual.jsp

Si lo tenemos correctamente instalado, el icono de la consola de EJSS aparecerá de la siguiente forma:

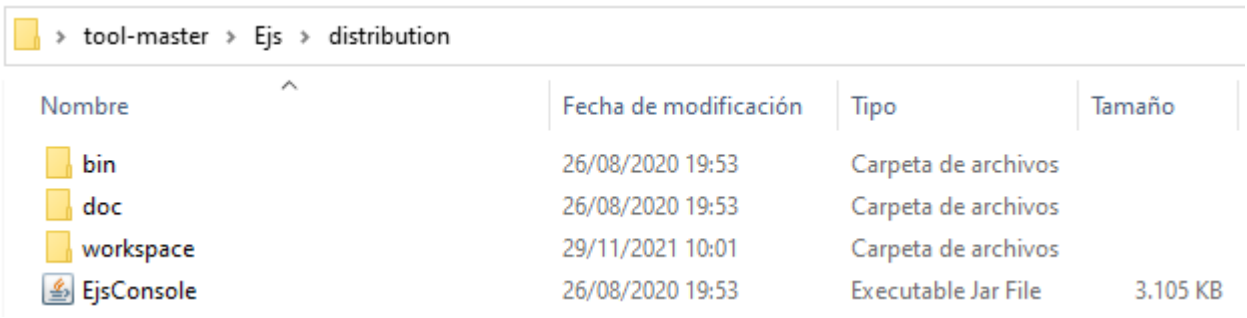


Figura 7.3 – Contenido de la carpeta distribution, en toolmaster > Ejs.

Haciendo doble click sobre EjsConsole debe abrise la consola de EJSS.

Hay que destacar que EJS permite dos lenguajes de programación: Java y JavaScript. En nuestro caso, para que pueda funcionar en modo remoto, además de que sea posible subirlo a la página de la UNED, se ha programado en JavaScript.

Al abrir la consola por primera vez, la opción predeterminada es Java, por lo que debemos cambiarla, tal y como se muestra en la figura 7.4.

Por otro lado, también se aprecia que el espacio de trabajo seleccionado es el que tiene la dirección del workspace de la figura 7.3, lo cual es correcto.

Por último, también se puede modificar el aspecto gráfico si se desea, o el nivel de zoom.

En nuestro caso, no será necesario modificar nada en opciones avanzadas ni en el área de mensajes. Ésta última pestaña puede ser útil para la notificación de errores, y para visualizar los datos de los experimentos en caso de no ejecutar el programa en el navegador (no recomendable).

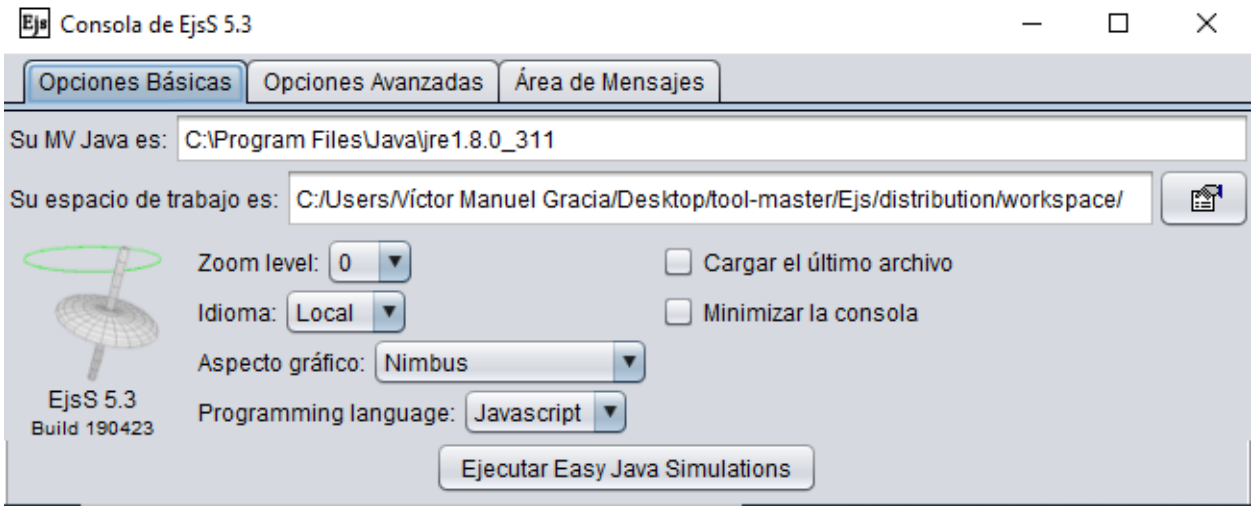


Figura 7.4 – Configuración de la consola de EJSS.

Es recomendable utilizar Google Chrome para la ejecución de EJSS en el navegador por cuestiones de compatibilidad. Esto puede seleccionarse de forma sencilla siguiendo los siguientes pasos:

- Pulsar en opciones de EJSS en la barra de opciones que aparece en el lateral (figura 7.5).

Figura 7.5 – Barra de lateral: Opciones de EJSS.

- Marcar la casilla “Use browser as prefrerred Run method”.
- Hacer click sobre el Tick azul que aparece a la derecha de “Preview command”, y establecer la ruta del navegador (figuras 7.6 y 7.7).

Una vez completados estos pasos, ya tenemos EJSS configurado correctamente en su version 5.3 estable. Existen otras versiones más avanzadas como la 6.0 (solo incluye el lenguaje JavaScript), pero se ha elegido la 5.3 por ser más fiable para realizar la comunicación con LabView.

Por último, comentar que para ejecutar cualquier .ejss es necesario tenerlo en el espacio de trabajo que tenemos en la consola que vemos de la figura 7.4.

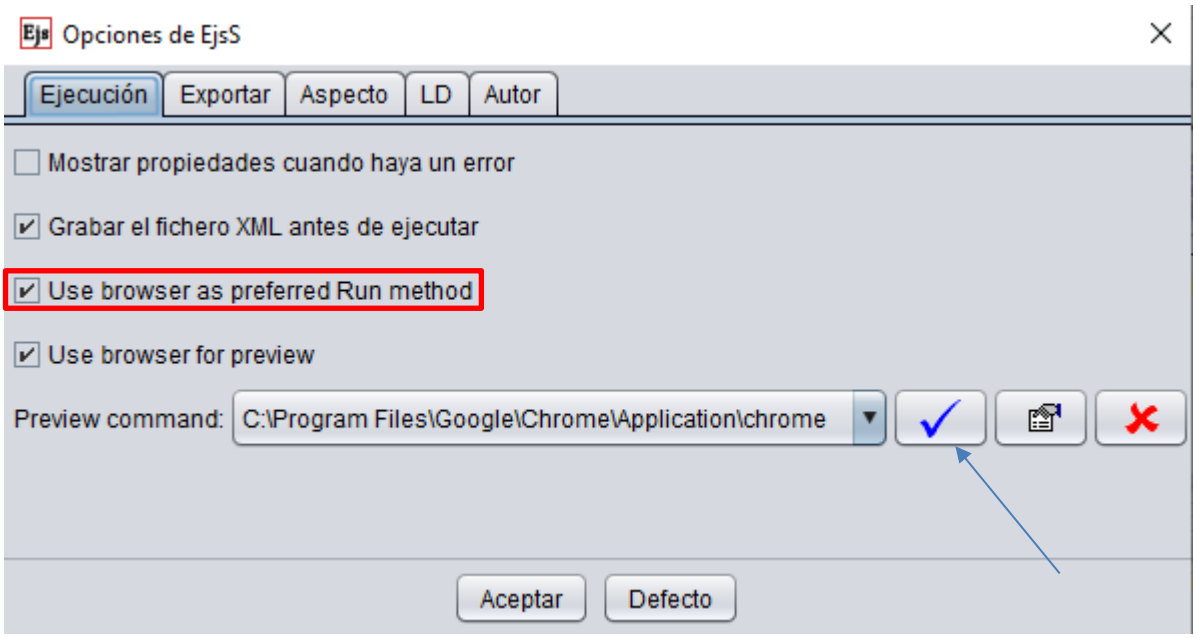


Figura 7.6 – Opciones de EJSS.

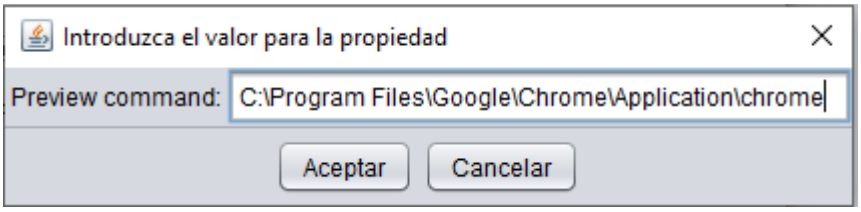


Figura 7.7 – Establecimiento de la ruta del navegador.

7.2 Operación en EJSS: Modo local y modo remoto.

7.2.1 Configuración del modo local.

Para ejecutar el proyecto en modo local, basta con tener la configuración del apartado 7.1, y tener dentro del workspace el archivo .ejss basado en el modelo de parámetros distribuidos del sistema.

Para cargarlo, pulsaremos sobre el icono con forma de carpeta (“Abrir una simulación del espacio de trabajo”) que encontramos en la barra lateral de EJSS (figura 7.8).

Después, seleccionamos el .ejss correspondiente (figura 7.9), y se abrirá. Por último, pulsamos el botón “Ejecutar la simulación” de la barra lateral (figura 7.8).

Tras esto, se abrirá el navegador y la simulación debe comenzar automáticamente.

Figura 7.8 – Barra de lateral: En rojo: Abrir una simulación del espacio de trabajo. En azul: Ejecutar la simulación.

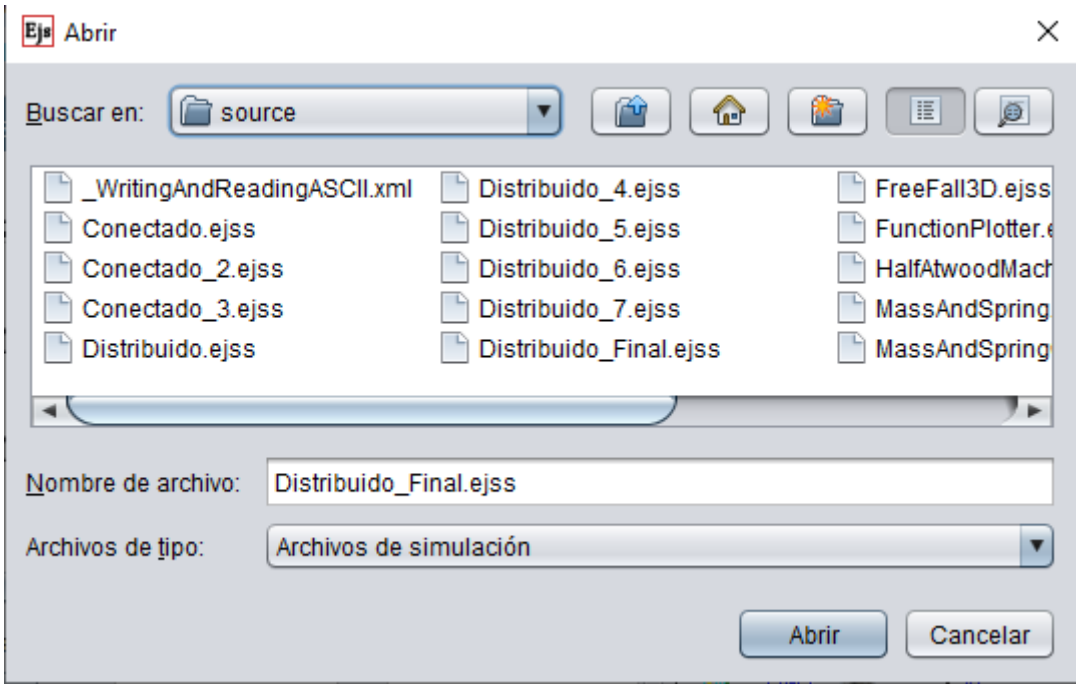


Figura 7.9 – Selección de archivo de simulación a abrir.

7.2.2 Configuración y conexión en modo remoto.

Para la configuración del modo remoto, es necesario tener configurada previamente la conexión entre LabView y la planta del laboratorio (Véase el Tabajo de Fin de Grado "Laboratorio Virtual para el Control de la Planta Multiprocesos basado en LabView").

Una vez hecho esto, debemos tener en cuenta el apartado 5.1. Por un lado, debe estar configurado el servidor RIP, que será el LabView basado en la planta. Por ello, Planta_Real_Tab_Contro.vi, además del .vi correspondiente a las variables globales (Global2.vi) deben estar en el directorio Private. También debe estar configurado el programa Global_Configurations.vi de forma análoga a la que se muestra en la figura 7.10.

En este momento, debemos iniciar el servidor RIP desde la aplicación RIPWebService, haciendo click derecho sobre "RIP" y pulsando Start. Se debe hacer click en aceptar en el mensaje de advertencia posterior (fig. 7.11).

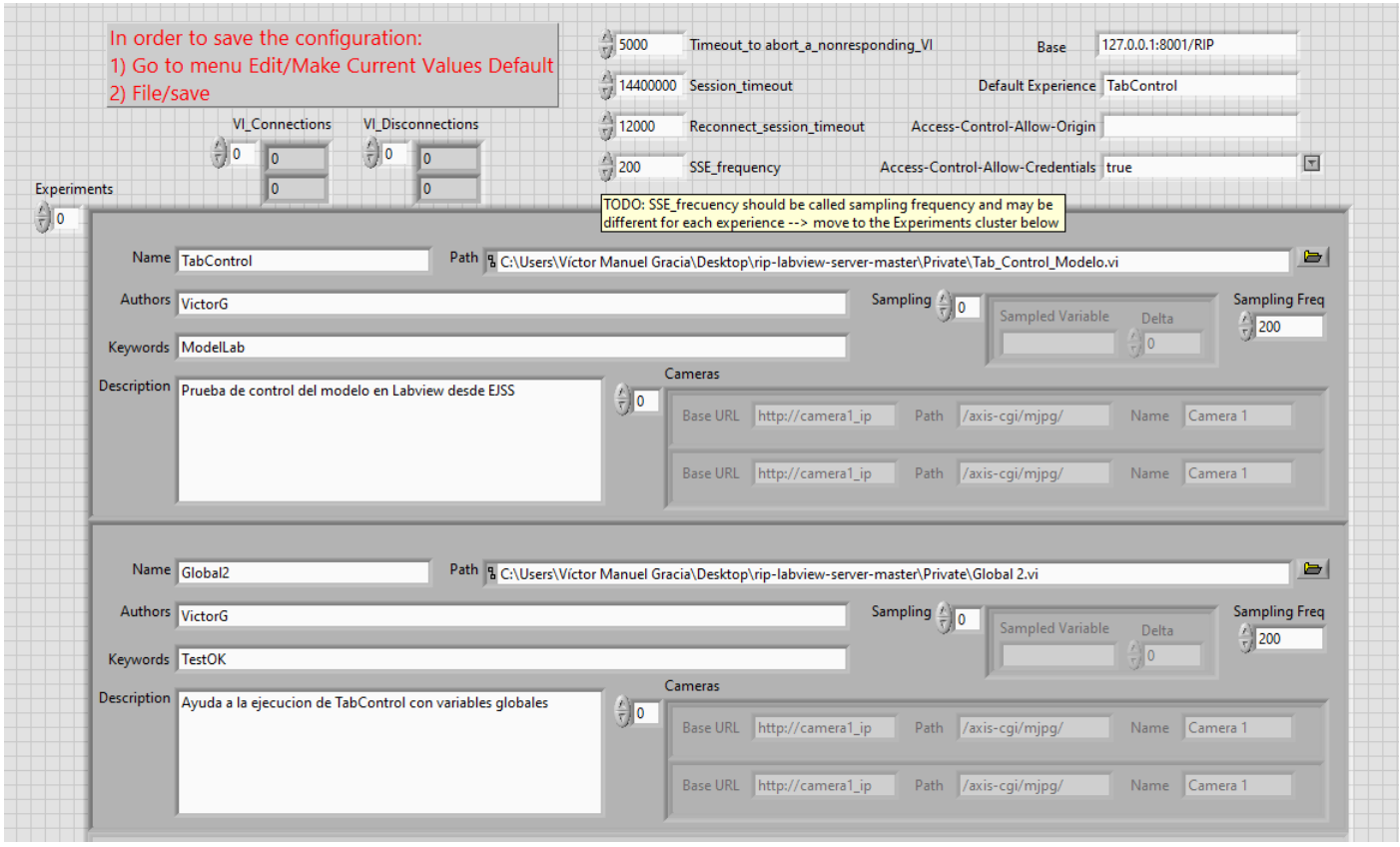


Figura 7.10 – Programa Global_Configurations.vi.

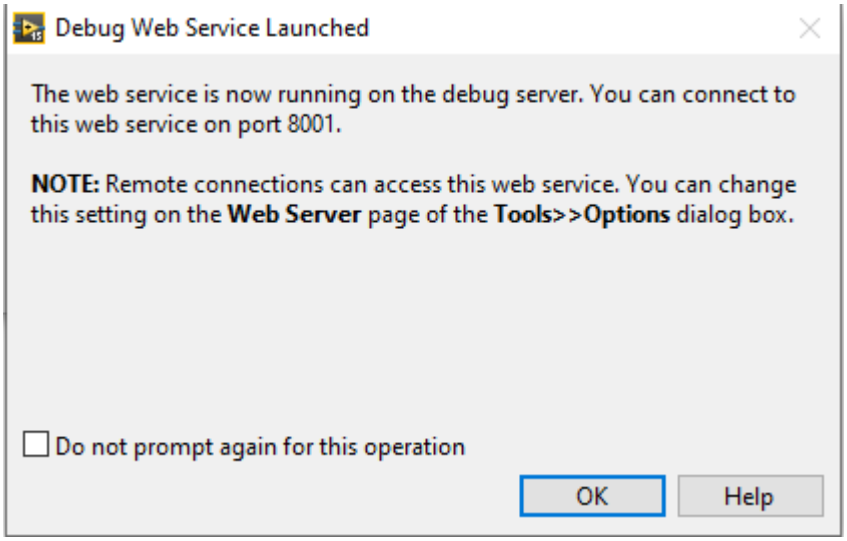


Figura 7.11 – Mensaje de inicio del servidor en LabView.

Por otro lado, se debe abrir el archivo Conectado.ejss para dirigirnos a Modelo > Elementos. En la lista de elementos, debemos hacer doble click sobre el elemento rip creado, el cual se llamará “rip”.

En la url del servidor (figura 7.12) escribimos lo siguiente: http://localhost:8001/RIP.

Después, pulsamos en “Get Experiences”, y nos debe aparecer una nueva fila (figura 7.13). Si no ocurre nada,

es probablemente porque hay algo que no está correctamente configurado, o el servidor no está iniciado.

Si todo está correcto, nos dirigimos a la pestaña "Experiences" y nos aseguramos de tener marcada la opción Planta_Real_Tab_Control.

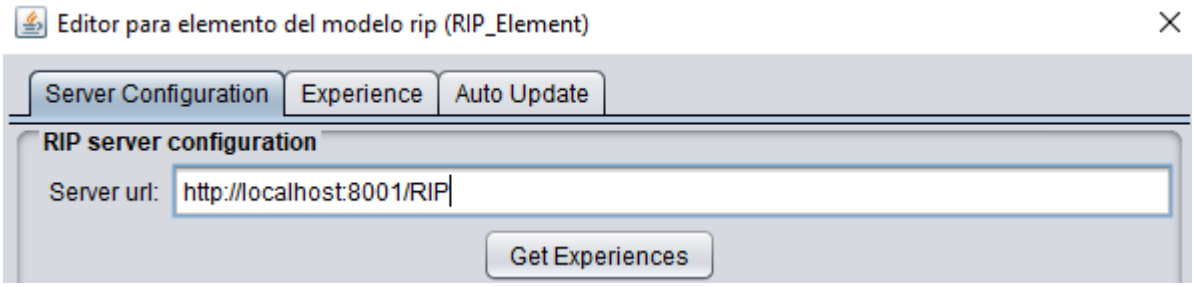


Figura 7.12 – Configuración del elemento RIP: Pestaña Server Configuration.

Por último, en la pestaña "Auto Update" deben estar asociadas las variables de lectura y escritura entre EJSS y LabView (figura 7.14 y 7.15) (es conveniente que las variables de ambos programas se llamen igual) que nos interesen para que el funcionamiento de la planta sea correcto.

Con todo lo comentado, bastará con iniciar la simulación para controlar la planta desde EJSS.

Nota: En caso de no estar ejecutando en el mismo ordenador el EJSS y el LabView, es necesario modificar la dirección url en el elemento RIP. Se debe colocar la dirección IP del ordenador que ejecuta el LabView de la planta en lugar de localhost (http://direcciónIP:8001/RIP).

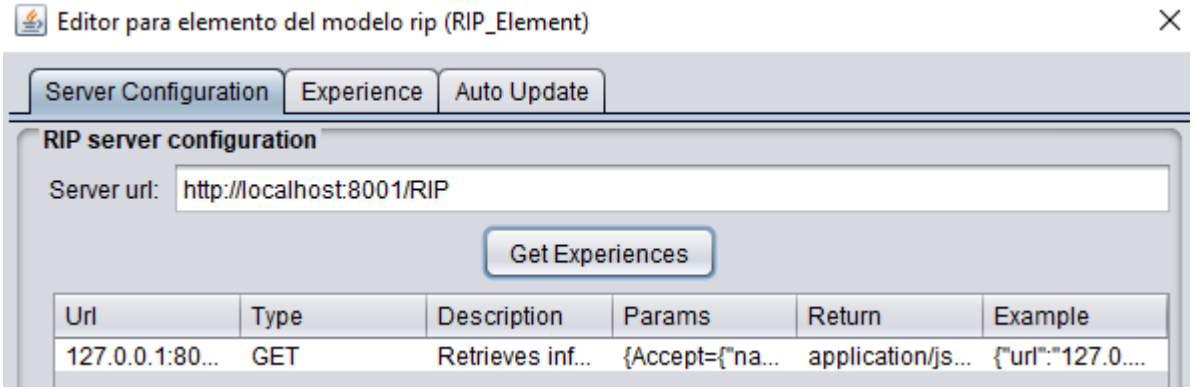


Figura 7.13 – Configuración del elemento RIP: Server Configuration tras pulsar Get Experiences.

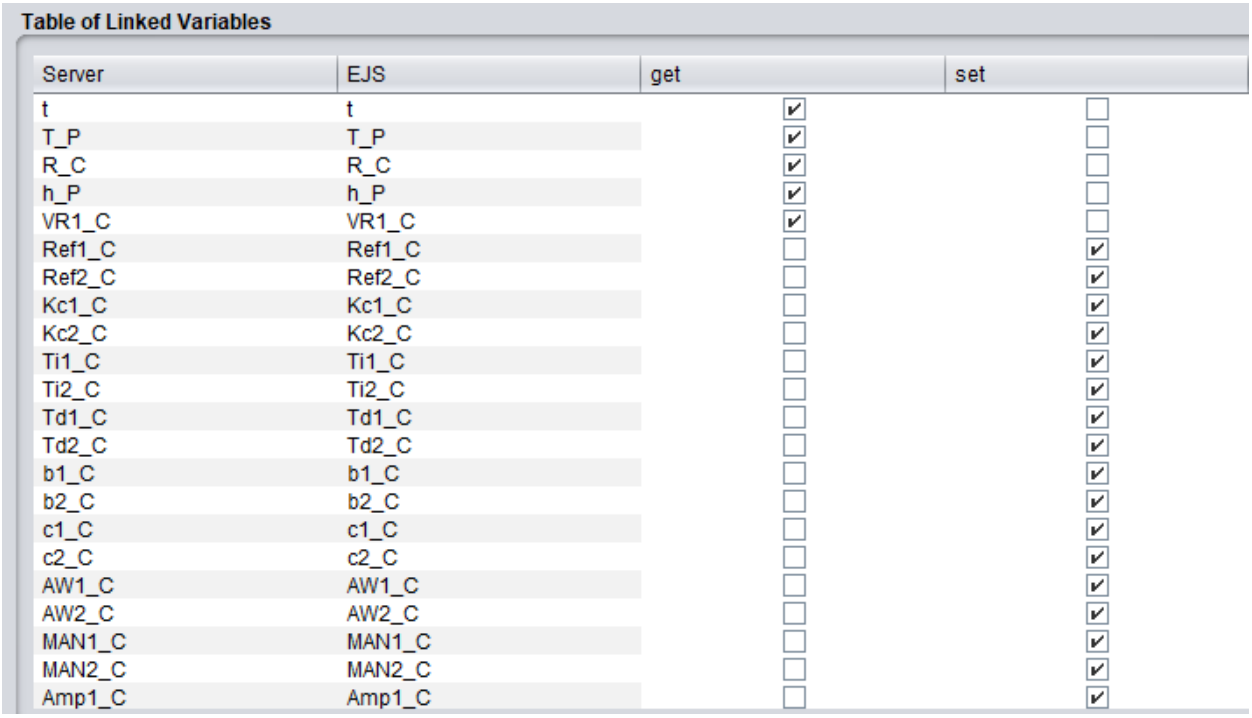
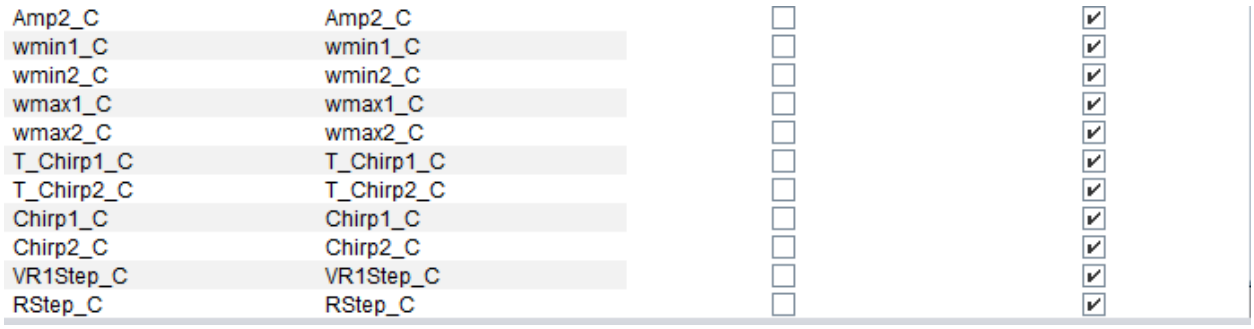


Table of Linked Variables

Server	EJS	get	set
t	t	☑	☐
T_P	T_P	☑	☐
R_C	R_C	☑	☐
h_P	h_P	☑	☐
VR1_C	VR1_C	☑	☐
Ref1_C	Ref1_C	☐	☑
Ref2_C	Ref2_C	☐	☑
Kc1_C	Kc1_C	☐	☑
Kc2_C	Kc2_C	☐	☑
Ti1_C	Ti1_C	☐	☑
Ti2_C	Ti2_C	☐	☑
Td1_C	Td1_C	☐	☑
Td2_C	Td2_C	☐	☑
b1_C	b1_C	☐	☑
b2_C	b2_C	☐	☑
c1_C	c1_C	☐	☑
c2_C	c2_C	☐	☑
AW1_C	AW1_C	☐	☑
AW2_C	AW2_C	☐	☑
MAN1_C	MAN1_C	☐	☑
MAN2_C	MAN2_C	☐	☑
Amp1_C	Amp1_C	☐	☑

Figura 7.14 – Configuración del elemento RIP: Conexión de variables (1).



Amp2_C	Amp2_C	☐	☑
wmin1_C	wmin1_C	☐	☑
wmin2_C	wmin2_C	☐	☑
wmax1_C	wmax1_C	☐	☑
wmax2_C	wmax2_C	☐	☑
T_Chirp1_C	T_Chirp1_C	☐	☑
T_Chirp2_C	T_Chirp2_C	☐	☑
Chirp1_C	Chirp1_C	☐	☑
Chirp2_C	Chirp2_C	☐	☑
VR1Step_C	VR1Step_C	☐	☑
RStep_C	RStep_C	☐	☑

Figura 7.15 – Configuración del elemento RIP: Conexión de variables (2).

7.3 Uso de la Interfaz Gráfica.

Como se comentó, la interfaz gráfica está organizada en dos paneles: Visualización y Configuración. Cada panel, a su vez, se organiza en pestañas. Esto se puede apreciar en la figura 7.16.

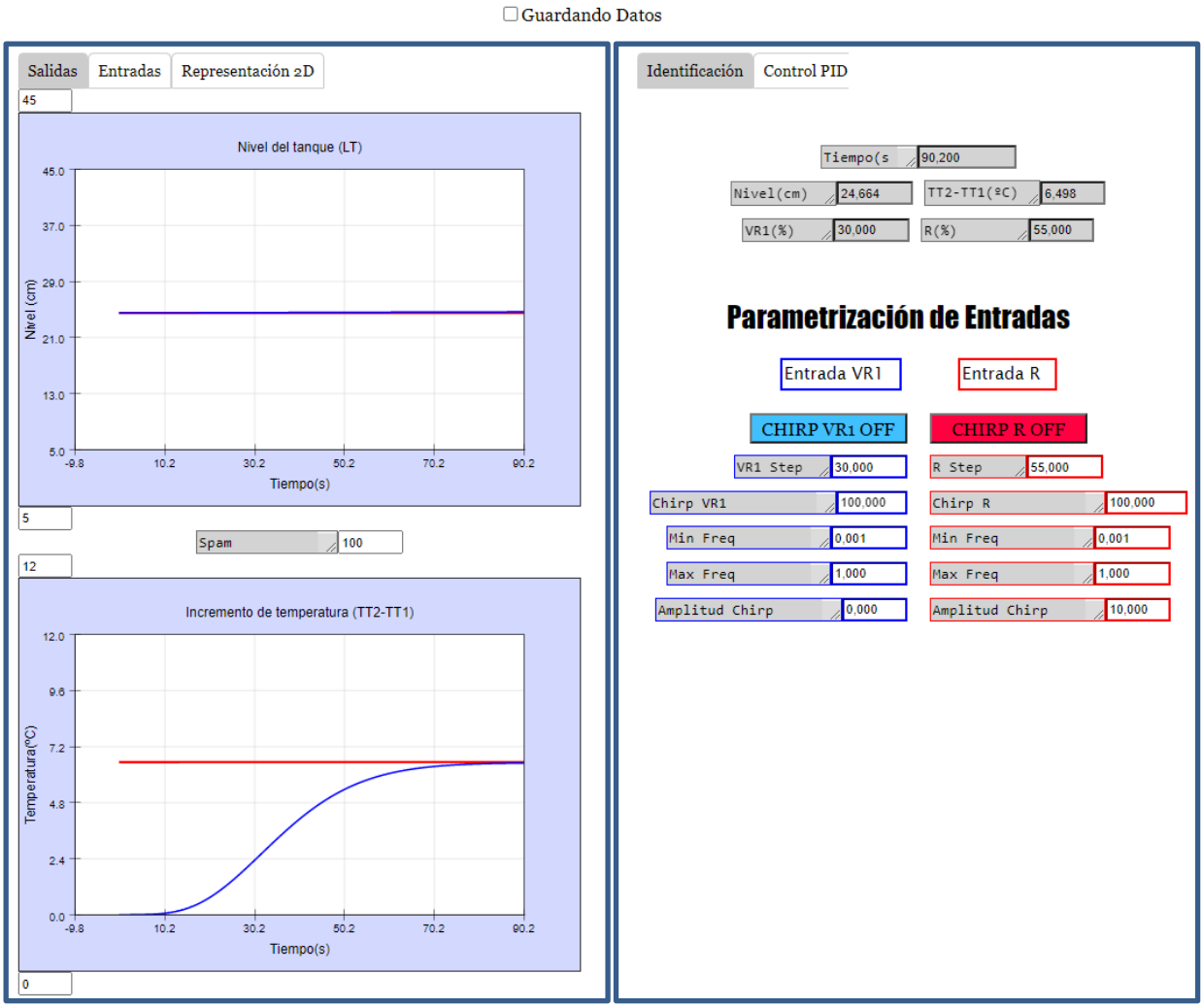


Figura 7.16 – A la izquierda: Panel de Visualización. A la derecha: Panel de Configuración.

7.3.1 Panel de Visualización.

Comenzando por el panel de visualización, se puede observar que hay tres pestañas distintas:

- Salidas.
- Entradas.
- Representación 2D.

7.3.1.1 Pestaña de Salidas.

En esta pestaña vemos dos gráficas, una sobre la otra. En la figura 7.17 se indica la funcionalidad.

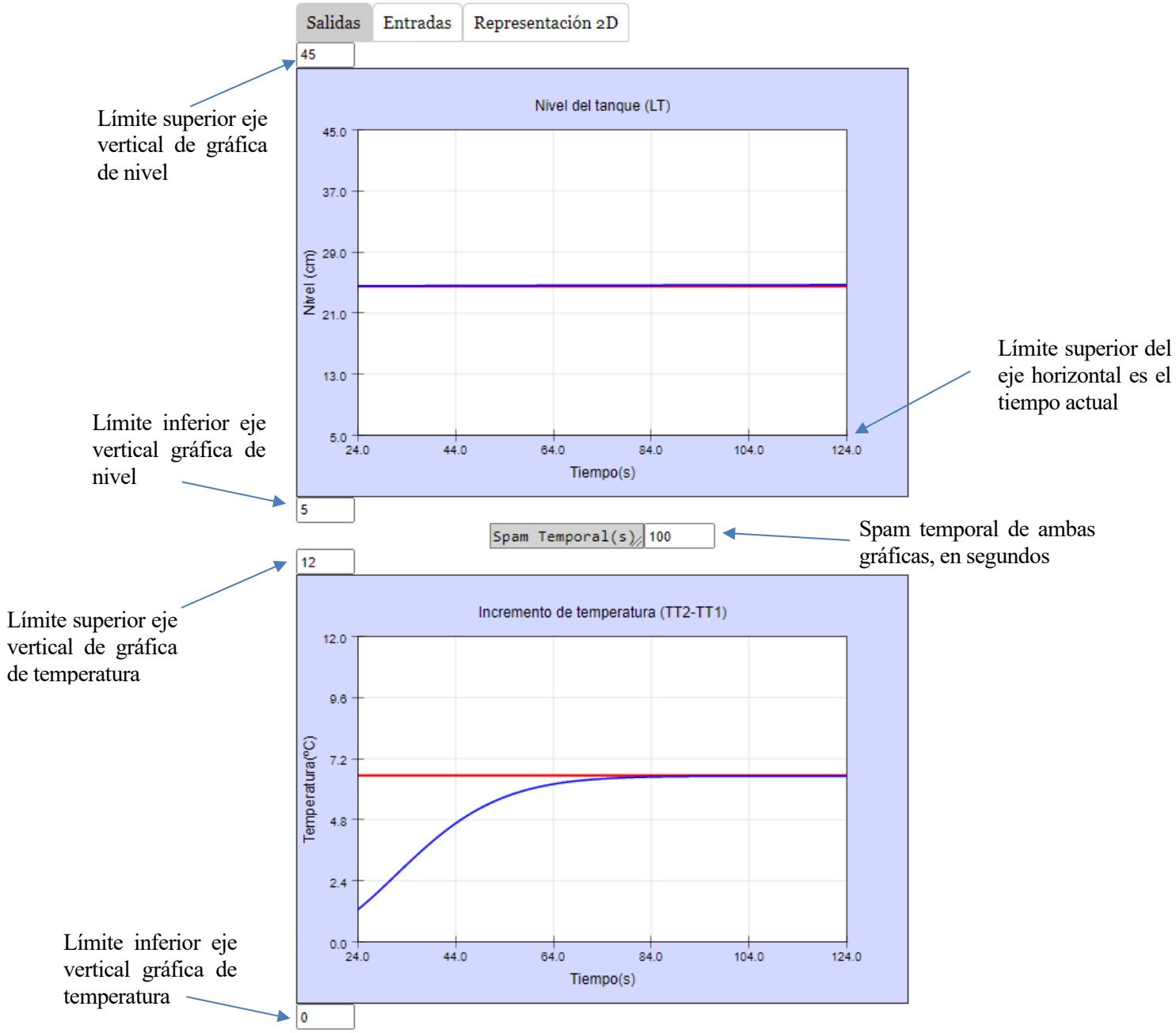


Figura 7.17 – Pestaña de Salidas: Funcionalidad.

Nota: En las gráficas de salida también se muestra el valor de la referencia. Ignorarla en modo manual o identificación.

7.3.1.2 Pestaña de Entradas.

En la pestaña de entradas también se aprecian dos gráficas, pero con los valores de las entradas.

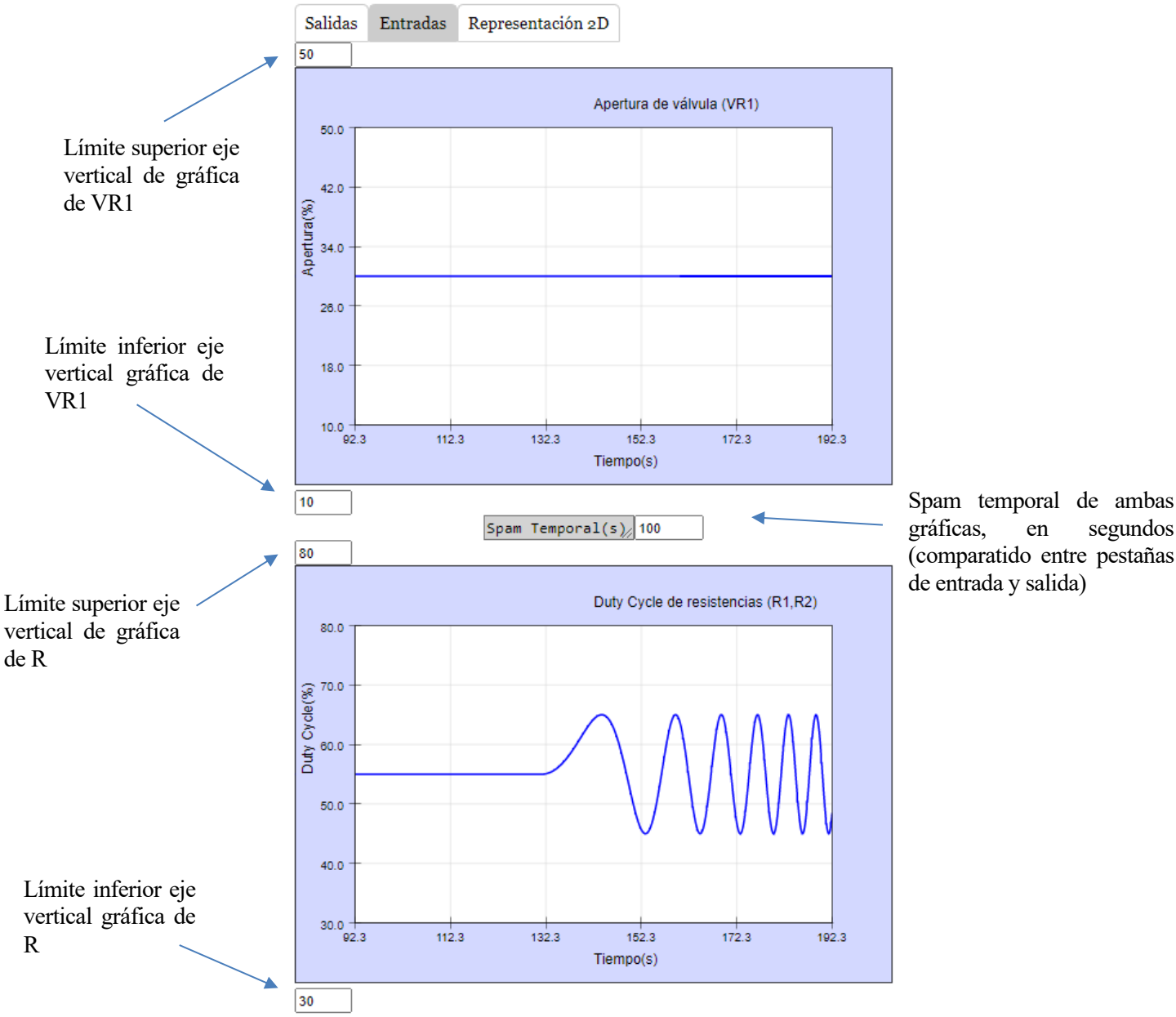


Figura 7.18 – Pestaña de Entradas: Funcionalidad.

7.3.1.3 Pestaña de Representación 2D.

La pestaña con el esquema 2D tiene una función meramente representativa. Muestra, mediante una animación, el nivel del tanque y la temperatura del agua que viaja por la tubería.

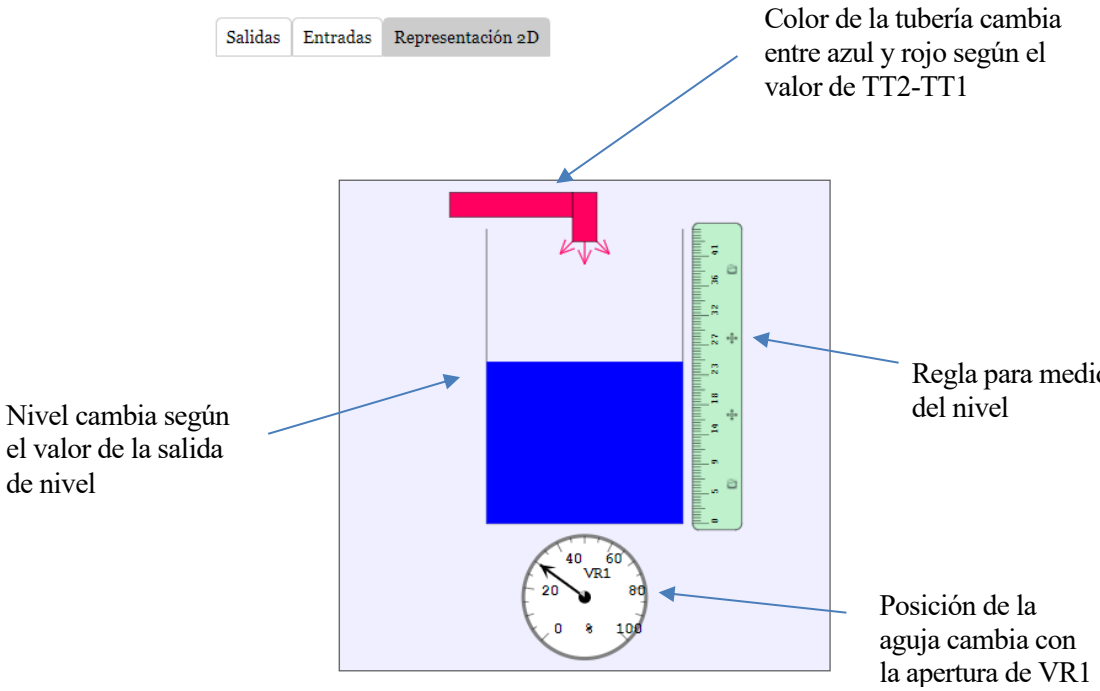


Figura 7.19 – Pestaña de Representación 2D: Funcionalidad.

7.3.2 Panel de Configuración.

En el panel de configuración podremos parametrizar tanto las señales de entrada, como los controladores PID, así como elegir el modo de funcionamiento del sistema. Está formado por las pestañas de Identificación y Control PID.

7.3.2.1 Pestaña de Identificación.

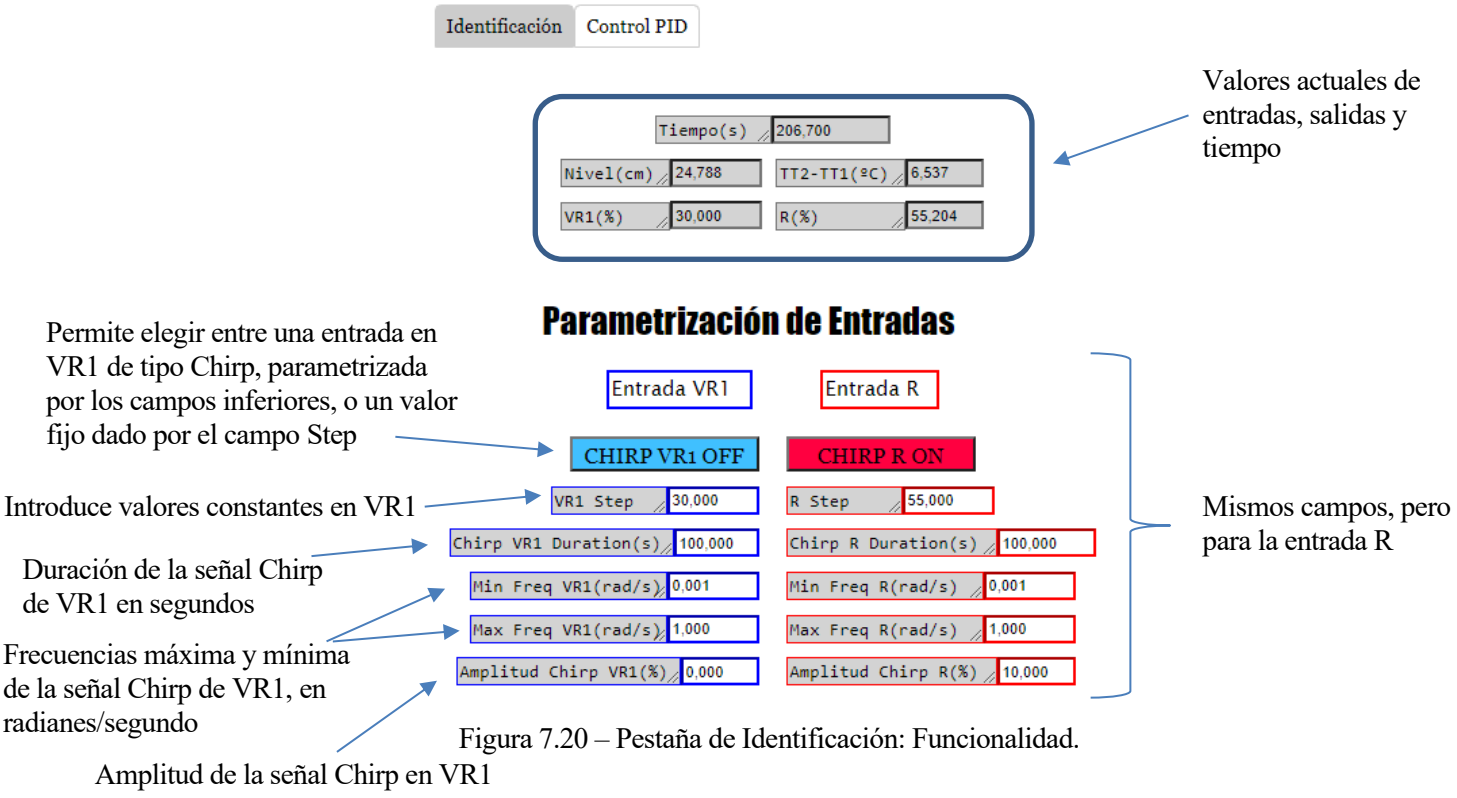


Figura 7.20 – Pestaña de Identificación: Funcionalidad.

7.3.2.2 Pestaña de Control PID.

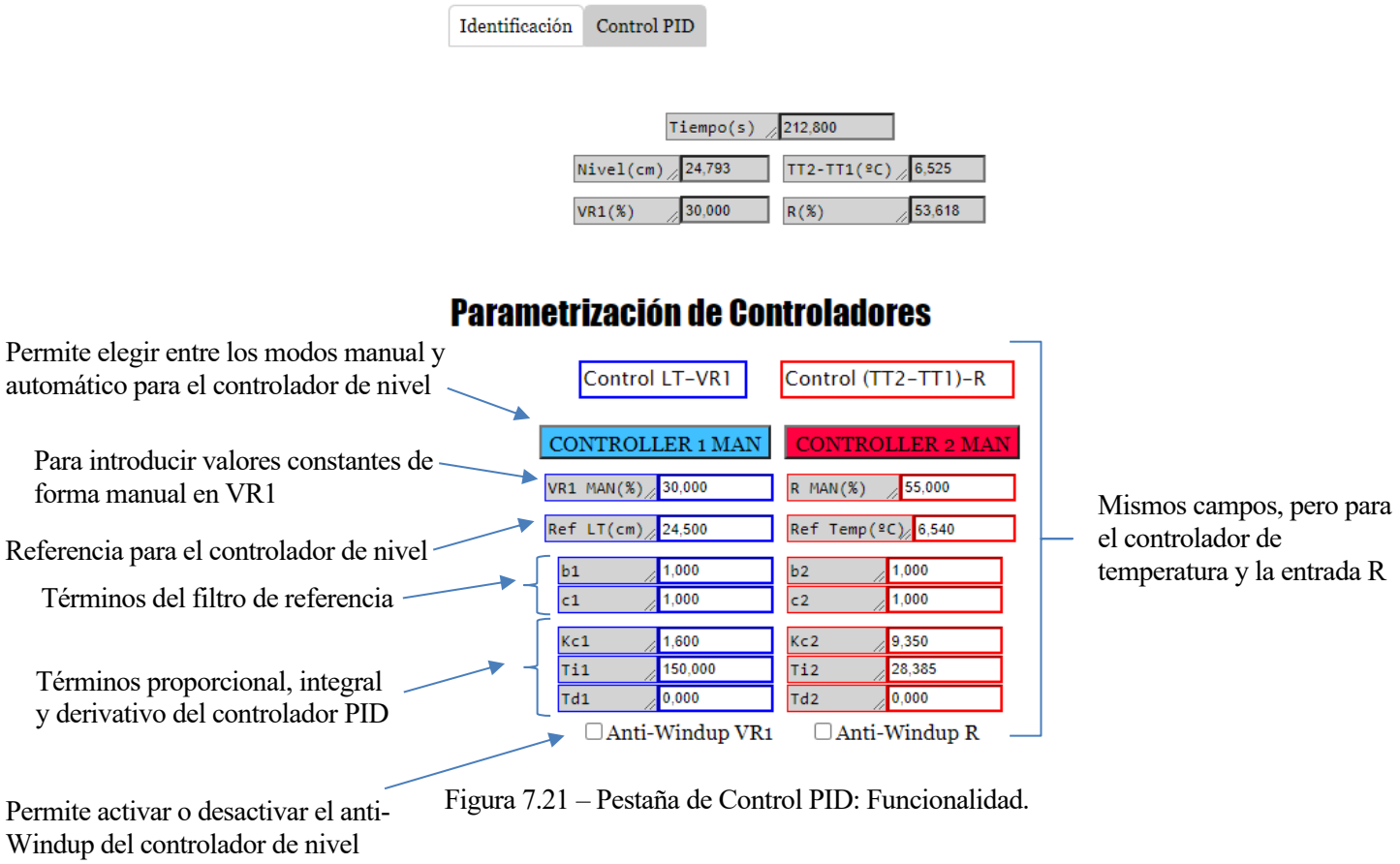


Figura 7.21 – Pestaña de Control PID: Funcionalidad.

7.4 Obtención y representación de los datos experimentales.

Como se puede observar en la figura 7.16, en la parte superior de la interfaz gráfica, fuera de los paneles de Identificación y Control PID, hay una opción que puede marcarse o no, llamada "Guardando datos".

Esta opción se encarga de imprimir los datos más relevantes de EJSS en la consola (del navegador si ejecutamos EJSS en el navegador, o de EJSS en caso negativo). Para abrir la consola del navegador debemos pulsar la tecla F12, y nos vamos a la zona donde ésta imprime los mensajes.

Dependiendo de si se quiere imprimir las referencias (caso de experimento de control PID) o si no (caso de experimento de identificación), se debe elegir una línea en la página de impresión de la figura 7.22 y comentar la otra.

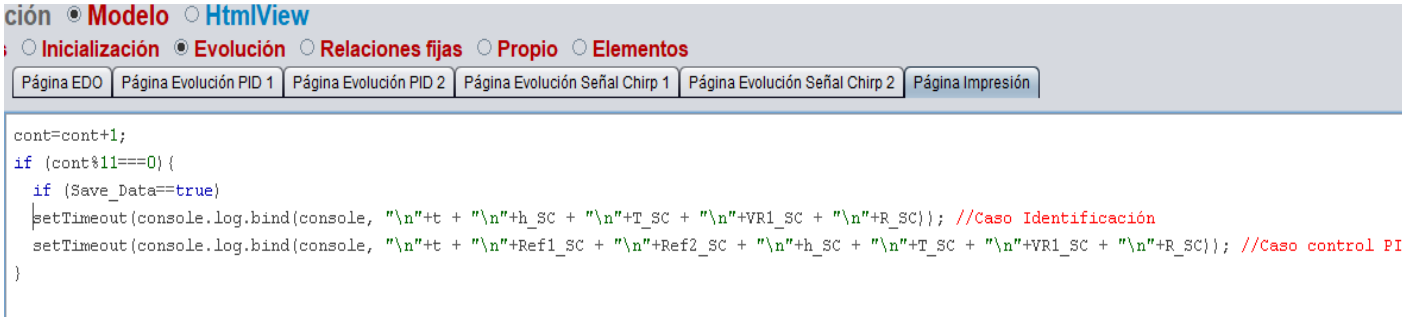


Figura 7.22 – Impresión de datos (caso local) en consola mediante EJSS.

La página de impresión varía ligeramente para el caso remoto, ya que se introduce una variable de tiempo anterior para que no se impriman los valores varias veces en el mismo instante de tiempo. Esto se muestra en la figura 7.23.

Página Evolución

```
if (t%1===0 && t_ant!=t){
  if (Save_Data==true)
  setTimeout(console.log.bind(console, "\n"+t + "\n"+h_SC + "\n"+T_SC + "\n"+VR1_SC + "\n"+R_SC));
  setTimeout(console.log.bind(console, "\n"+t + "\n"+Ref1_SC + "\n"+Ref2_SC + "\n"+h_SC + "\n"+T_SC + "\n"+VR1_SC + "\n"+R_SC));
}
t_ant=t;
```

Figura 7.23 – Impresión de datos (caso remoto) en consola mediante EJSS.

Los datos se imprimen en forma de columna. Si se hace click derecho sobre los datos en la consola, aparece la opción "Guardar como" (figura 7.24). Esto nos generará un archivo de texto .log que se puede cargar en Matlab.

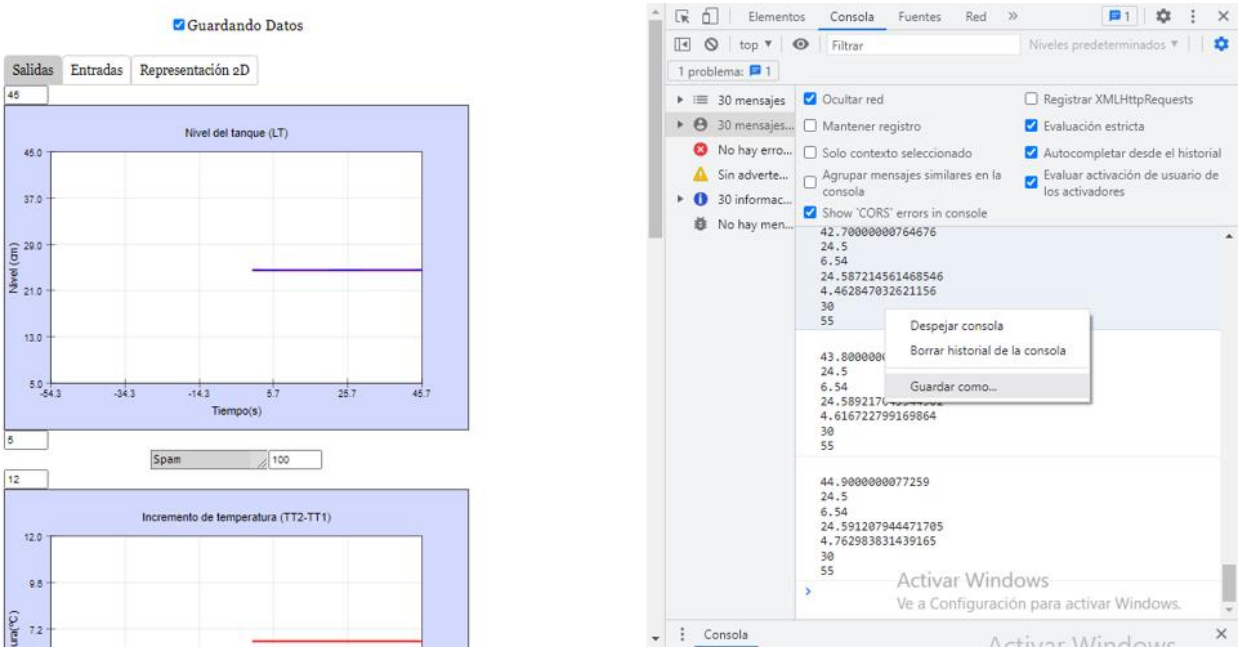


Figura 7.24 – Consola del navegador, guardar datos como archivo .log.

Podemos ponerle cualquier nombre. En nuestro caso se le ha llamado data, pero se quiere poner otro nombre, se debe modificar el archivo .m para que cargue el archivo con el nombre elegido.

Tanto el archivo de datos .log como el .m para la carga y representación de datos deben estar en el mismo directorio. Si ejecutamos el archivo "read_data.m" nos pedirá que marquemos 1 si el ensayo es de identificación, y 2 si el ensayo es de control. Esto se muestra en la figura 7.25.

```
>> read_data
Introduce 1 si Identificación. Introduce 2 si Control:
```

Figura 7.25 – Mensaje de elección de experimento en Matlab.

Una vez se seleccione un número, el programa procesará los datos y los representará en dos gráficas, una para

cada par entrada-salida.

REFERENCIAS

[1] Gracia Villegas, V. M. (2020, 26 marzo). *Laboratorio virtual para el control de la planta multiprocesos basado en LabView*. IdUS - Depósito de Investigación Universidad de Sevilla. Recuperado 2 de diciembre de 2021, de https://idus.us.es/handle/11441/94586

[2] Esquembre, F., & García Clemente, F. J. (2009). *Easy Java Simulations Wiki.* - EJSS Wiki. https://www.um.es/fem/EjsWiki

[3] Christian, W., & Esquembre, F. (2015, 4 Noviembre). *Easy Java/JvaScript Simulations Manual.*

[4] UNEDLabs. (2019). GitHub - UNEDLabs/rip-spec: *Specification of the Remote Interoperability Protocol.* GitHub. http://github.com/UNEDLabs/rip-spec

[5] de la Torre, L., Chacón, J., & Chaos, D. (2019, 30 Octubre). *Remote Interoperability Protocol for online laboratories.* Madrid, España.

[6] G.U.N.T. Gerätebau GmbH, Manual de Experimentos del Sistema Didáctico Modular para la Automatización de Procesos RT 450, 2005.

[7] National Instruments (2008). *Control Design Toolkit User Manual.*

[8] Julian-Laime, Edgar & Almidón Elescano, Ángel. (2018). Manual de programación LabVIEW 9.0. 10.5281/zenodo.2557815.