Artículo de Investigación. Revista Ciencia e Ingeniería. Vol. 42, No. 3 pp. 317-326, agosto-noviembre, 2021.
ISSN 1316-7081. ISSN Elect. 2244-8780 Universidad de los Andes (ULA)

Control y supervisión de procesos desde un teléfono inteligente usando el microcontrolador arduino uno

Control and supervision of processes from a smartphone using the arduino uno microcontroller

Plaza, Miguel[1]*; Quintero, Jormany[2]
[1]Departamento de desarrollo en ingeniería de datos, Babblelabs, Campbell, CA 95008, United States
[2]Departamento de Sistemas de Control, Escuela de Ingeniería de Sistemas, Facultad de Ingeniería, Universidad de Los Andes 5101, Mérida, Venezuela
*miguelplaza94r@gmail.com

Resumen

Actualmente el término internet de las cosas está cobrando mayor aceptación en todas las áreas de conocimiento. Esta tecnología permite la comunicación entre computadoras y los sistemas físicos, a través de sensores que envían la información vía internet. Uno de los propósitos de la ingeniería de control es velar por el correcto comportamiento de los procesos autoregulados, en tal sentido, la supervisión de estos procesos ha ido evolucionando con la tecnología. El objetivo de esta investigación consistió en diseñar una aplicación móvil para la supervisión y control de procesos de manera remota desde un smartphone haciendo uso del microcontrolador Arduino UNO. Esta aplicación fue desarrollada en Javascript bajo la filosofía de cliente/servidor en el entorno de ejecución Node.js, Johnny-Five y Framework React Native. El microcontrolador Arduino UNO, por su versatilidad y robusta comunicación, se utilizó como convertidor de señales analógicas y digitales acoplado con un filtro pasabajas. Se diseñó un sistema de control en lazo cerrado clásico que incorpora un PID en discreto para la autorregulación del sistema. La interfaz diseñada para la aplicación móvil, además de ser intuitiva y llamativa, ofrece las opciones para la captura y envío de datos por correo electrónico. Las pruebas y verificaciones del sistema de control integrado se realizaron en un tanque de presurizado TY 35/EV. El uso de la aplicación para el control de procesos en tiempo real mediante un smartphone fue efectiva, permitiendo así la libertad del operador de trasladarse y verificar el comportamiento del sistema desde cualquier lugar con acceso a internet.

Palabras clave: aplicación móvil, internet de las cosas, Arduino UNO, Node.js, control de procesos.

Abstract

The term internet of things is currently gaining acceptance in all areas of knowledge. This technology allows communication between computers and physical systems through sensors that send information via the Internet. One of the purposes of control engineering is to ensure the correct behavior of self-regulated processes, in this sense, the supervision of these processes has evolved with technology. The objective of this research was to design a mobile application for monitoring and controlling processes remotely from a smartphone using the Arduino UNO microcontroller. This application was developed in Javascript under the client/server philosophy in the Node.js runtime environment, Johnny-Five, and React Native Framework. The Arduino UNO microcontroller, due to its versatility and robust communication, was used as an analog and digital signal converter coupled with a low-pass filter. A classical closed-loop control system was designed incorporating a discrete PID for system self-regulation. The interface designed for the mobile application, in addition to being intuitive and eye-catching, offers the options for data capture and sending data via e-mail. Tests and verifications of the integrated control system were performed on a TY 35/EV pressurized tank. The use of the application for real-time process control via smartphone was effective, allowing the operator the freedom to move and verify the behavior of the system from anywhere with internet access.

Keywords: mobile application, internet of things, Arduino UNO, Node.js, process control.

1 Introducción

Hoy en día, se maneja mucho el concepto de Internet de las Cosas IoT (Siglas en inglés de *Internet of Things*), el cual ha ganado la atención de varios investigadores (Zeinab y col., 2017), por convertirse en una tecnología importante que mejora y facilita la calidad de vida. Esta tecnología permite la comunicación entre los objetos, máquinas y cualquier dispositivo con las personas, a través de diferentes tipos de sensores; esto permite acceder al internet de forma remota (o por vía alámbrica) utilizando varias conexiones como el RFID, WIFI, Bluetooth y ZigBee (Zeinab y col., 2017). Uno de los aspectos más influyentes en estas tecnologías es el monitoreo del correcto funcionamiento de cualquier tipo de sistema industrial tales como reactores, calderas, motores entre otros muchos más equipos.

Uno de los dispositivos que se integra perfectamente con la filosofía de IoT por su adaptabilidad con la tecnología inalámbrica y el uso de los diferentes dispositivos móviles como los *smartphones*, es el Arduino UNO. Es un dispositivo microcontrolador de código abierto, de fácil programación, en cualquier instante de tiempo (Louis, 2016). Presenta la capacidad de actuar como una computadora pequeña y fungir de controlador digital dentro de un sistema realimentado por la implementación de múltiples leyes de control.

Dada la aplicabilidad que se ha demostrado en el uso de Arduino UNO junto con IoT en diferentes ambientes como sistema de control de equipos de uso doméstico (Piyare, 2013), (Ilyaz y cols., 2016), regulación de temperatura (Singh y cols., 2017), monitoreo del cambio de voltaje de un sistema desde un lugar seguro, algunas aplicaciones de tipo control supervisorio (Darandale y col., 2017), hacen de este microcontrolador una herramienta valiosa para el ingeniero de hoy.

Uno de los retos para cualquier profesional de la ingeniería es buscar la integración de los diferentes tipos de tecnologías existentes y novedosas; es por ello que cada vez se observa más el uso de *smartphones* en todas las áreas de conocimiento. En el campo de la ingeniería de Control de Procesos el uso de estos dispositivos ha ido incrementando, dada las ventajas que ofrece. Entre las ventajas ofrecidas destacan la buena capacidad de memoria y procesamiento para resolver problemas que una computadora pudiese solucionar, bajos costos de adquisición, facilidad de trasladarse con ellos y el poco espacio que ocupan.

En este contexto, dado que en Venezuela no se encontraron estudios publicados que integren el uso de teléfonos móviles con el microcontrolador Arduino UNO en el área de control de procesos, surge la necesidad de comenzar a incorporar estos dispositivos como herramientas para la supervisión y control de sistemas a distancia. Por tal motivo, el objetivo de esta investigación consistió en diseñar una aplicación móvil para la supervisión y control de procesos de manera remota desde un *smartphone* haciendo uso del microcontrolador Arduino UNO, esta aplicación fue desarrollada en uno de los lenguajes más cotizados hoy en día como lo es día como lo es Javascript.

2 Desarrollo de la aplicación móvil para el control y supervisión remota

Para el desarrollo de la aplicación móvil se utilizó el enfoque cliente-servidor (Ionos, 2019). Se diseñaron rutinas alojadas en el servidor llamado Backend, la implementación de la aplicación móvil en Javascript llamado Frontend, para la conexión con el servidor y proceso una rutina que configuró el dispositivo Arduino UNO como tarjeta de adquisición de datos.

2.1 Implementación del Backend

Para el desarrollo del Backend fue necesario cumplir con ciertos requisitos tanto de software como de hardware, los cuales se explican en las siguientes secciones.

2.1.1 Requisitos de Hardware

El Backend del sistema junto con el servidor local se alojaron y ejecutaron en una computadora portatil HP note-book 15. Esta máquina contenía las siguientes especificaciones:

- Procesador AMD A6-7310 APU de 2.16GHz.
- Arquitectura de 64 bits.
- Memoria RAM de 4.00GB.
- Disco Duro Hitashi HTS725050A9A362.
- Adaptador de WIFI Realtek RTL8188EE 802.11 bgn.

Además, el dispositivo enrutador usado para la conexión a la red inalámbrica fue el punto de acceso 4G para la conexión LTE Cat4, marca airtel.

2.1.2 Requisitos de Software

Para la implementación del servidor se instaló como sistema operativo Linux Mint 19.2 y se usaron las herramientas gratuitas disponibles en la web:

- Entorno de ejecución NodeJS versión 8.9.1 para la programación en Javascript.
- Framework Express.js en su versión 4.16.4. para la creación del servidor local.
- Framework Johnny-Five versión 1.1.0 para la conexión con el Arduino UNO.
- Paquete Socket.io 2.2.0. para la comunicación en tiempo real de la aplicación.

2.1.3 Implementación del Servidor Local

En la creación del servidor local fue preciso instalar y usar una instancia del framework Express.js. Primero, se creó un archivo para el punto de entrada de la aplicación, en el cual se declararon las rutas y la inicialización del servidor (Plaza, 2020). Para la definición del servidor local se siguieron los siguientes pasos:

1) Instalación del paquete, a través de la función *require* de Node.js.
2) Declaración de un puerto para las peticiones del servidor. En este caso, se utilizó por omisión el puerto 3000.
3) Definición de las rutas, middlewares y demás características con las que cuenta el framework.
4) Asignación del puerto 3000 para las peticiones del servidor.

2.1.4 Diseño y flujo del servidor

La aplicación móvil está diseñada para realizar un monitoreo de las señales medibles en un sistema de control SISO (*Single In Single Out*) en funcionamiento. Las señales elegidas para la monitorización fueron la salida del sistema, el error y la señal de control del sistema. El servidor contiene implementada la ley de control, en este caso se decidió implementar el controlador en lazo cerrado usando un tipo PID basado en la discretización por velocidad. Adicionalmente se incluyó la posibilidad de usar sus variantes PD, PI, o P ofreciendo una cierta libertad para el usuario al momento de la puesta en marcha.

Para asegurar la toma de datos en tiempo real, se implementó el módulo socket.io de Javascript tanto del lado del servidor como del cliente. Este módulo está compuesto de dos partes:

- Un servidor virtual que se integra con el servidor HTTP de Node.JS: socket.io, que permite la ejecución de funciones en tiempo real en el Backend.
- Una librería cliente que se carga del lado de la aplicación móvil: socket.io-client, que ejecuta las funciones del requeridas por el cliente en tiempo real.

Las acciones que toma el servidor del sistema respecto a las peticiones de emisor y receptor asociadas al *socket*, junto con el cálculo y elección de las leyes de control se presentan en la Fig. 1.

2.1.5 Integración del Arduino UNO con el Backend

El microcontrolador Arduino UNO ofrece múltiples ventajas para la adquisición de datos, sin embargo, carece de un módulo de conversión para transformar datos digitales en analógicos. En la alimentación del sistema físico de control se necesita de una señal analógica continua de voltaje que varíe entre *0V* y *5V*. La solución a este problema fue utilizar las salidas PWM que ofrece el microcontrolador.

Para filtrar la señal PWM del Arduino, se utilizó un filtro pasivo pasabajas, como se ilustra en la Fig. 2, con los valores $R=99{,}7\Omega$ y $C=102{,}4\mu F$. Se implementó un seguidor de tensión en la salida del filtro utilizando un amplificador operacional LM324 para compensar la variación de impedancia que puede ocasionar el circuito.

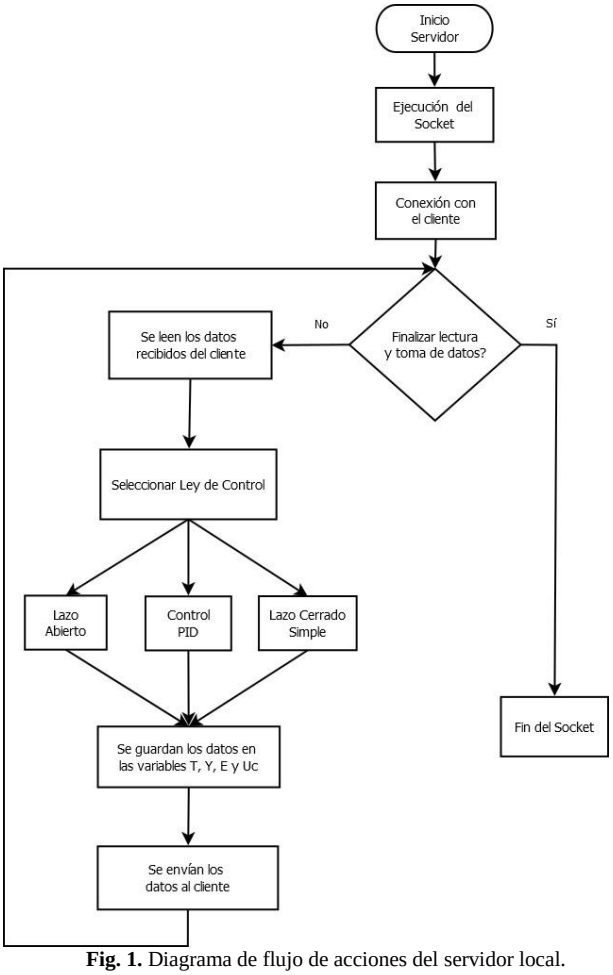

Fig. 1. Diagrama de flujo de acciones del servidor local.

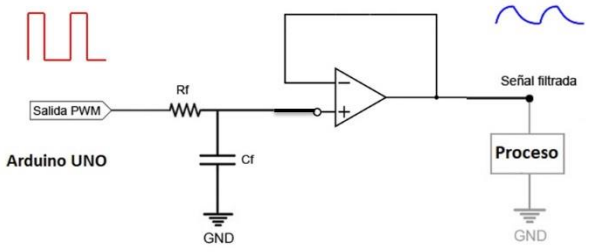

Fig. 2. Esquema del filtro pasa bajas.

Al momento de crear un *script* en Arduino se hace desde un IDE de específico mediante el lenguaje de programación C. A efectos de la aplicación móvil, el Arduino UNO se programó utilizando Javascript y el entorno de ejecución Node.js integrados en el Backend. El software utilizado para este propósito fue el framework Johnny-Five. Esta plataforma está basada en un protocolo muy importante llamado Firmata. Para implementar el protocolo Firmata en Arduino, se instaló y usó el sketch StandardFirmata incluido en el IDE de Arduino y cargarlo a la placa para asegurar la comunicación con el microcontrolador.

2.1.6 Tiempo de muestreo

El sistema implementado funciona en tiempo real. Para asegurar que la toma de datos estuviese sincronizada entre el servidor y el cliente, se procedió a realizar un ajuste. Este ajuste se hizo necesario dado al retraso percibido en cada iteración generado por el tiempo que demora la conexión con el servidor, la demora producida por la comunicación del *socket* con la aplicación móvil, la velocidad de la red inalámbrica local, el tiempo de lectura y escritura de datos en Arduino UNO y el tiempo de cálculo en la ley de control.

Después de numerosas pruebas y toma de datos, se concluyó que el tiempo acumulado en cada iteración es de aproximadamente $T_a=2{,}5ms$. Por lo tanto, al introducir el período de muestreo, la aplicación internamente realiza el ajuste $T_0real=T_0-T_a$. Esto implica que para el tiempo de muestreo que el usuario ingresa (T_0), se disminuye el tiempo T_a en cada iteración; por ejemplo, si el usuario ingresa un tiempo de muestreo de *1s*, el tiempo en el que el Arduino UNO escribirá cada muestra será de $T_0total=T_0real+0{,}0025$. Es importante tomar en cuenta el tiempo de muestreo al momento de analizar el comportamiento del sistema. Cuando el tiempo de muestreo es muy pequeño (menor a *0,5s*) la toma de datos y el sistema de ejecución no asegura un control en tiempo real. Es por esta razón que la aplicación móvil contiene la restricción de tiempo de muestreo mayor a *0,5s*.

2.2 Implementación del Frontend

Se implementó una aplicación funcional híbrida en dos sistemas operativos móviles (Android e iOS) utilizando un mismo código de programación. Para este fin, se usó el lenguaje de programación JavaScript. Para el desarrollo del Frontend fue necesario cumplir con ciertos requisitos de software, definir las variables asociadas al sistema, el flujo de la aplicación y las pantallas o *wireframes* de la aplicación, los cuales se explican en las siguientes secciones.

2.2.1 Variables asociadas al sistema

Luego de dar la bienvenida y pasar por las pantallas de inicio, el usuario elige entre tres opciones de control disponibles: Lazo Abierto, Lazo Cerrado Simple y Control PID, para el sistema que desea controlar y monitorear. Las variables implicadas a cargo del usuario se pueden ver en un esquema de caja negra como en la Fig. 3. Estas variables están debidamente validadas para asegurar el ingreso correcto de los datos.

- **Variables de entrada:** selección (Option), tiempo de muestreo (T0), valor de entrada o referencia (U/R) según sea la opción, ganancia proporcional (Kp), tiempo integral (Ti), tiempo derivativo (Ti), correo electrónico (email).
- **Variables de salida:** señal de salida (Y), señal de error (E), señal de control (Uc), tiempo total de toma de datos (T).

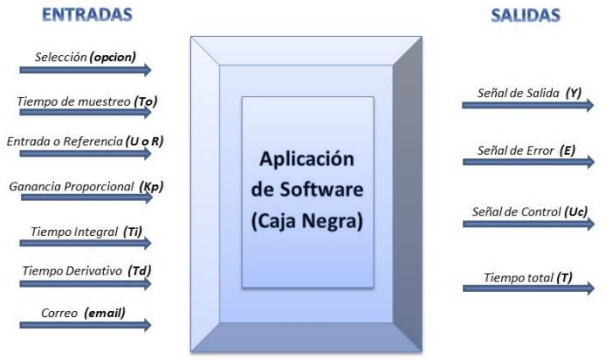

Fig. 3. Diagrama de caja negra del sistema.

Una vez cargadas las variables en la aplicación, estas son enviadas al servidor por vía web para la ejecución del sistema. En la Fig. 4 se observa el diagrama de flujo para la secuencia del funcionamiento del sistema en conjunto Cliente/Servidor, donde la caja con borde discontinuo representa el módulo de ejecución del servidor.

2.2.2 Requisitos para la instalación de la aplicación

La ejecución de la aplicación en los dispositivos móviles necesita la instalación de diferentes herramientas de software, estas fueron:

- Entorno de ejecución NodeJS versión 13.4.0.
- Herramienta Expo SDK versión 36.0.0.
- Framework React Native versión 36.0.0.
- Componente React versión 16.9.0.
- Componente react-native-responsive-linechart version 2.2.0.
- Paquete socket.io-client versión 2.1.1.
- Aplicación móvil Expo Project versión 2.14.0.

Al igual que en el Backend, en el Frontend de la aplicación se deben tener instalado los componentes y funciones asociadas al *socket* (socket.io-client) para la comunicación en tiempo real del sistema.

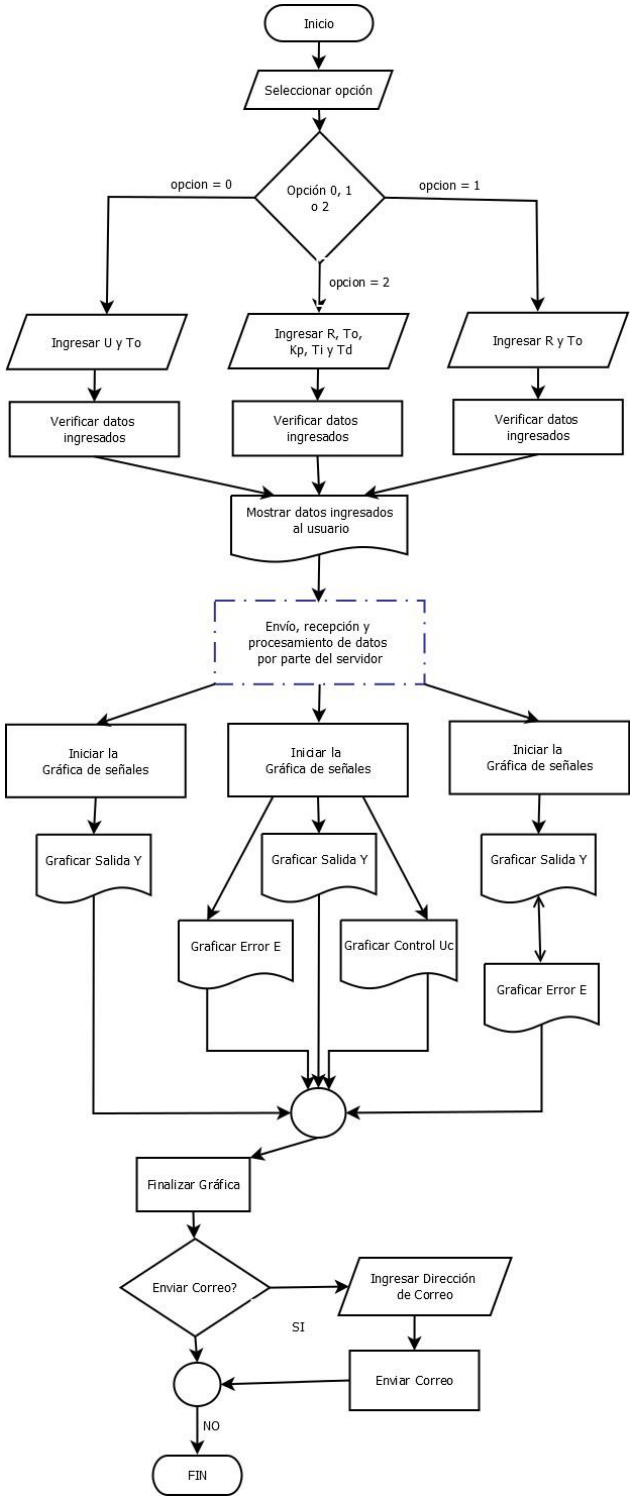

Fig. 4. Diagrama de flujo de las acciones del sistema Cliente/Servidor.

2.2.3 Diseño de las vistas de la aplicación

La ejecución del cliente sigue el esquema de la Fig. 4. Se divide dicho diagrama en distintos módulos para facilitar la comprensión de las pantallas. La vista "pantalla inicial" se visualiza en la Fig. 5a. Esta funciona de bienvenida para el usuario y establece una comunicación estable durante el uso de la aplicación.

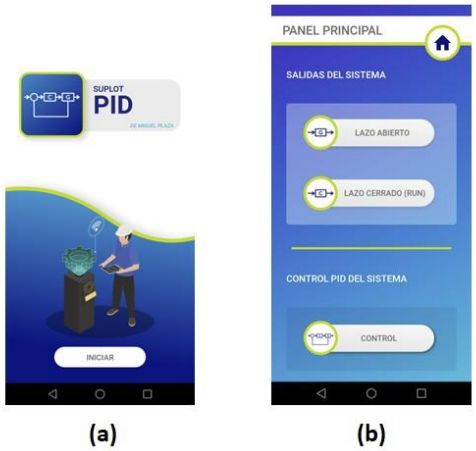

Fig. 5. Diagrama de flujo de las acciones del sistema Cliente/Servidor.

Seguidamente se muestra al usuario las opciones de control para interactuar con el sistema (Fig. 5b). Hecho esto, se visualiza el formulario para el ingreso de las variables en la Fig. 6.

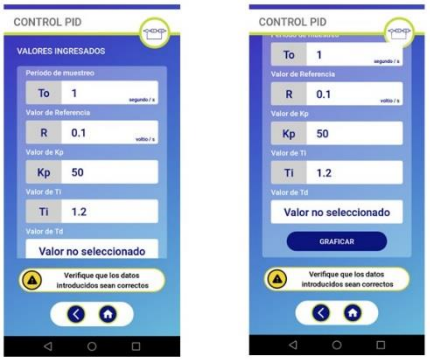

Fig. 6. Datos solicitados para un control del tipo PID.

Una vez puesto en marcha el sistema de control, la aplicación ofrece la posibilidad de mostrar de forma gráfica las variables en los últimos 8 datos leídos para un tiempo de muestreo T_0 como se ilustra en la Fig. 7. Paralelamente, los datos se van almacenando en memoria RAM del *smartphone* y luego enviadas, vía digital, al correo electrónico ingresado por el usuario previamente.

Fig. 7. Gráfica de datos para un control del tipo PID.

3 Sistema de Control

El sistema físico utilizado para la ejecución de la aplicación móvil fue un proceso de presurizado de aire TY 35/EV. La planta física se visualiza en Fig. 8. Dicho proceso contiene:

- Un tanque presurizado, el cual permite el almacenamiento del aire.
- Una bomba eléctrica de circulación que proporciona el flujo de aire a almacenar.
- Un medidor de presión que permite indicar la presión diferencial actual en el tanque presurizado en unidades de Bar. El rango de medida del sensor es de *0Bar* a *2,5Bar*.
- Una servoválvula que facilita la regulación del aire de salida al tanque.

El proceso (compresor) está diseñado para tener como salida del sistema de control la variable de presión diferencial dentro del tanque. El sistema se debe alimentar con una fuente externa de *5V*, y la variable manipulada es el voltaje de trabajo para la servoválvula del sistema, este varía entre *0V* y *5V*.

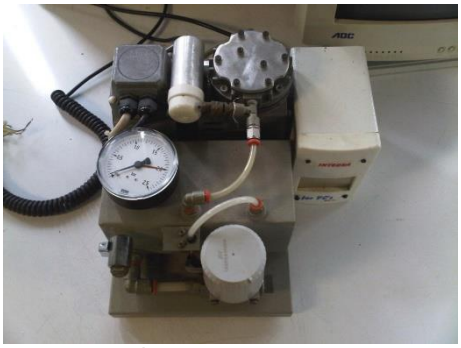

Fig. 8. Planta Física: Compresor de Aire TY 35/EV.

3.1 Modelo matemático del sistema

Dada la dificultad de realizar un modelo matemático paramétrico por las no linealidades involucradas en el sistema seleccionado y la complejidad de obtener los valores exactos de los parámetros, se obtuvo un modelo matemático lineal mediante la identificación de sistemas, haciendo uso del *Toolbox System Identification* del sistema de cómputo MATLAB.

Con una muestra del sistema en lazo abierto para un tiempo de muestreo de T_0*=0,5s*, se utilizó el método de identificación ARX para estimar los parámetros del modelo con un polo y sin ceros. El sistema obtenido se representa mediante Eq. 1. El modelo equivalente en discreto para un T_0*=0,5s* se observa en la Eq. 2. Este modelo representa la dinámica del sistema alrededor del punto de operación *3,824V*. En la Fig. 9 se aprecian la curva de los datos del sistema real recolectados (curva negra), el modelo de primer orden continuo (curva verde) y el modelo de primer orden discreto (curva azul).

$$G(s) = \frac{0{,}0338}{6s + 1} \tag{1}$$

$$G(z) = \frac{0{,}0012}{1 - 0{,}9399z^{-1}} \tag{2}$$

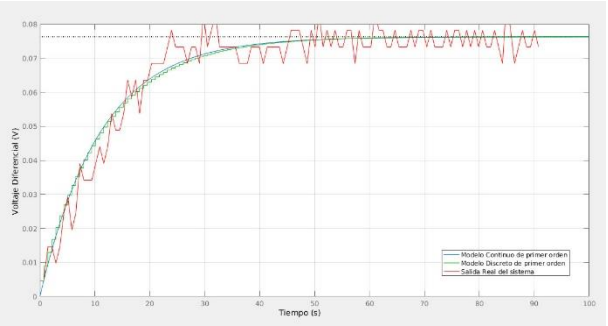

Fig. 9. Respuesta del sistema real y simulaciones.

3.2 Ley de Control

El esquema clásico en diagramas de bloque del sistema de control en lazo cerrado que sigue el proceso se visualiza en la Fig. 10.

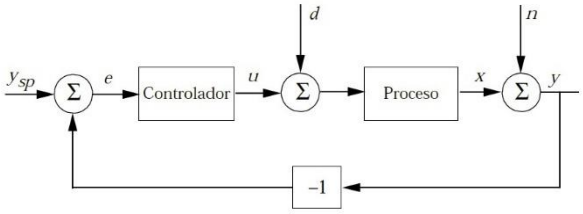

Fig. 10. Diagrama de bloques de un lazo de realimentación con controlador.

La ley de control aplicada en el sistema físico fue la de un control Proporcional-Integral-Derivativo (PID) descrita en la Eq. 3.

$$u(t) = Kp\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right) \tag{3}$$

Se implementó un controlador del tipo PI. La función de transferencia de este controlador viene dada por la Eq. 4.

$$C(s) = Kp + \frac{Kp}{T_i s} = Kp\frac{1 + T_i s}{T_i s} \quad (4)$$

Para este sistema se eligió como especificaciones de diseño un tiempo de asentamiento T_s*=25s* y un porcentaje de sobredisparo *SD=20%*. Los valores del controlador calculados son *Kp=21,865* y T_i*=1,2453s*. En la Fig. 11 se observa la simulación del sistema en lazo cerrado. Se aprecia como el error en el régimen permanente es cero y en el régimen transitorio se cumplen las especificaciones planteadas.

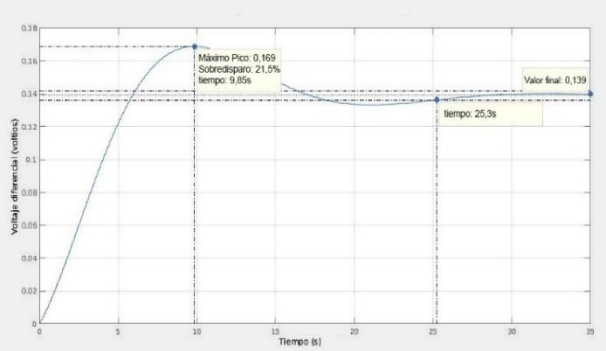

Fig. 11. Gráfica del sistema global ante una entrada de *0,139V*.

3.3 Discretización del Controlador PI

Se discretizó el modelo del controlador PI continuo obtenido. Utilizando el método de la integración numérica rectangular por adelanto. El modelo del controlador obtenido se describe en Eq. 5.

$$C(z) = \frac{21{,}865 - 13{,}086z^{-1}}{1 - z^{-1}} \quad (5)$$

4 Pruebas y Resultados

4.1 Prueba del sistema integrado

Sustituyendo los valores de *Kp* y T_i anteriores y utilizando un tiempo de muestreo T_0*=0,5s*, se obtuvo la implementación del controlador como se observa en la Eq.6.

$$\begin{aligned} u(k) &= u(k-1) + 21{,}865e(k) - 13{,}086e(k-1) \\ q_0 &= 21{,}865 \\ q_1 &= -13{,}086 \\ q_2 &= 0 \end{aligned} \quad (6)$$

Para la puesta en marcha de la aplicación móvil, se seleccionó la opción de 'Control PID'. Se realizó un control PI con los siguientes parámetros para la toma de datos y el control: T_0*=0,5s*, *Ref=0,139V*, *Kp=21,865*, T_i*=1,2453* y *Td=0*. Al verificar los datos ingresados en la aplicación, se procedió a ejecutar el sistema diseñado. El Arduino UNO recopila la señal de salida del sistema físico y en pantalla se muestra en tiempo real los últimos 8 datos recibidos por el dispositivo móvil.

4.2 Resultados

Con los datos recibidos por correo electrónico luego de la prueba, se simularon estos con ayuda de un *script* en MATLAB. Se graficaron las variables de salida (*y*), Error (*e*) y Señal de Control (*uc*) para un tiempo total *T=60s*. En la Fig. 12 se aprecia el comportamiento de la variable controlada (*y*), en ella se observa que el sistema se estabilizó alrededor del valor de referencia *0,139V*. Sin embargo, respecto a la simulación hecha, la salida real no se comporta del todo igual a la simulada.

Tal y como se observa, el sistema real se estabiliza un poco antes de los *30s*, mientras que en la simulación se estabiliza a los *25,3s*. Esto se debe a la saturación de la entrada del sistema, donde el control *uc* tomó como valor máximo unos *5V*; adicionalmente se debe tomar en cuenta la velocidad en que el tanque se llena de aire cuando la servoválvula se encuentra completamente abierta, entre otras.

Otra característica importante que presenta la curva real, es que posee un sobredisparo de aproximadamente un *10%* cumpliendo como parte de los requisitos de diseño, mientras que la simulación posee un sobredisparo de aproximadamente un *21%*.

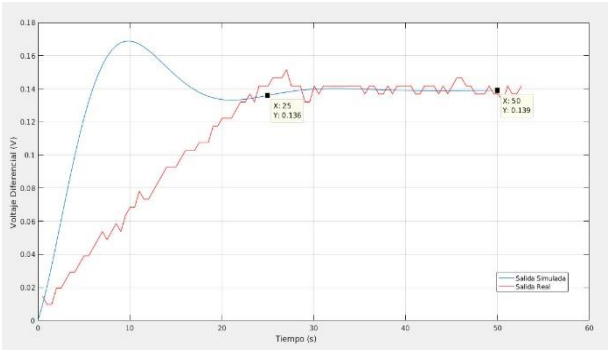

Fig. 12. Comparación de la salida del sistema global simulada y real.

Para el caso del estado transitorio, el error comienza a tener una magnitud elevada al inicio de la toma de datos, pero comienza a disminuir a medida que la salida del sistema se acerca más al valor de referencia. En la Fig. 13 se observa la señal de control y como es su evolución en el tiempo. Esta señal oscila alrededor del valor de *4,1V* para el estado estacionario tanto para los datos reales como para la simulación.

En el estado transitorio se observa que el control llega hasta el valor máximo de saturación de *5V* para la señal real, es decir, la servoválvula se abre al máximo para que el tanque de presión se llene lo más pronto posible de

aire comprimido, por otro lado, se observa que la gráfica del control simulado aumenta drásticamente hasta llegar a aproximadamente unos *7,24V*. Esta discrepancia con los valores simulados es debida a las limitaciones físicas en cuanto a inercias y comportamientos de la válvula que no fueron tomados en cuenta para la simulación teórica. Sin embargo, a pesar de las diferencias en cuanto al estado transitorio, se puede observar cómo el controlador diseñado se desempeña correctamente cumpliendo con los requerimientos planteados.

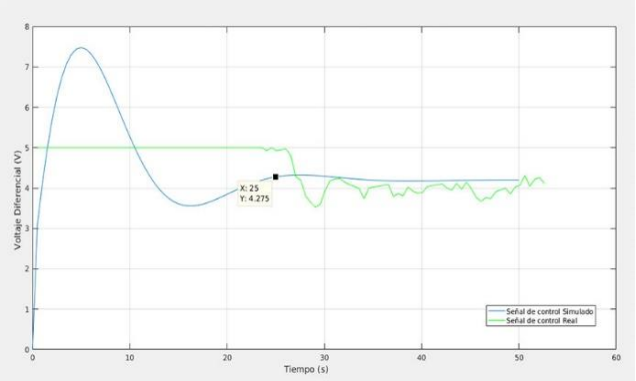

Fig. 13. Comparación de la señal de control PI del sistema global simulada y real.

5 Conclusiones

En esta investigación se desarrolló una aplicación móvil de tipo supervisor para la adquisición y control de procesos, los cuales pueden ser manipulados de forma remota en tiempo real desde un *smartphone*; lo cual permite al usuario realizar sus actividades de monitorización desde cualquier lugar con acceso a internet. La integración de nuevas tecnologías permitió la comunicación efectiva entre el usuario y el proceso físico con ayuda de la placa Arduino UNO.

El uso del componente *socket* permite una conexión inalámbrica eficiente entre servidor y cliente de manera instantánea, permitiendo así la lectura y visualización en la pantalla de la aplicación móvil de todas las señales asociadas al sistema. El proceso usado para las pruebas, cumple los requerimientos mínimos en los cuales el sistema de control se ejecuta de forma correcta y certera, permitiendo así al componente de software ejecutar y enviar las señales en tiempo real. Esto es fácilmente comprobable por las gráficas elaboradas con los datos arrojados por el sistema.

Para esta aplicación móvil no se creó una base de datos de almacenamiento de toda la información recopilada. Es recomendable tener un registro de todos los datos recibidos y enviados por la aplicación móvil, ya que permitiría tener un historial del comportamiento e información útil que pudiera ayudar en la toma de decisiones a un nivel superior. En futuras actualizaciones se aconseja agregar el valor y la escala de las unidades de la variable controlada, así como extender el sistema de control a múltiples entradas y múltiples salidas. Se desea posteriormente la incorporación de diferentes estrategias de control, con el fin de enriquecer el sistema y asignar una ley de control más eficiente según las necesidades y limitaciones físicas que pueda presentar el proceso de estudio.

Conflicto de intereses

Los autores declaran que no hay conflicto de intereses.

Referencias

Darandale DC & Gunjal BL, 2013, Development of Web-Based SCADA like Application using Arduino Platform, International Journal of Emerging Technology and Advanced Engineering, Vol. 3, No. 8, pp. 172-176.

Ilyaz B, ChiktayM& Salahuddin D, 2016, Home automation using arduino WiFi module ESP8266. Ionos. Qué es un servidor? Un concepto, dos definiciones. Se encuentra en https://www.ionos.es/digitalguide/servidores/knowhow/que-es-un-servidor-un-concepto-dos-definiciones/. Fecha de consulta: 01 de octubre de 2019.

Lous L, 2016, Working Principle of Arduino and using it as a Tool for Study and Research, International Journal of Control, Automation, Communication and Systems (IJCACS), Vol. 1, No. 2, pp. 21-29.

Piyare R, 2013, Internet of things: ubiquitous home control and monitoring system using android based smartphone, International Journal of Internet of Things, Vol. 2, No. 1, pp. 5-11.

Plaza M, 2020, Diseño de una aplicación móvil para la adquisición y el manejo de datos de manera inalámbrica usando el microcontrolador Arduino UNO (Tesis de pregrado), Departamento de Sistemas de Control, EISULA, Universidad de Los Andes, Mérida, Venezuela.

Santiago C, 2019, Desarrollo de una Aplicación Web para la Gestión de Condominios (Tesis de pregrado), Departamento de Sistemas Computacionales, EISULA, Universidad de Los Andes, Mérida, Venezuela.

Singh K, Moloy D & Pritam R, 2017, Automatic fan speed control using Arduino, International Journal of Novel Research and Development, Vol. 2, No. 4, pp. 75-77.

Zeinab K & Elmustafa S, 2017, Internet of Things applications, challenges and related future technologies, World Scientific News, Vol. 2, No. 67, pp. 126-148.

Recibido: *18 de febrero de 2021*

Aceptado: *05 de junio de 2021*

Plaza, Miguel: *Ing. de Sistemas opción Control y Automatización. Desarrollador Junior de Backend y Frontend en Aplicaciones Móviles y Plataformas Web, Desarrollador FullStack en BabbleLabs, CA, United States*

Quintero, Jormany: *Msc. Modelado y Simulación de Sistemas. Ing. de Sistemas opción Control y Automatización. Profesor en el Departamento de Sistemas de Control de la Escuela de Ingeniería de Sistemas, Facultad de Ingeniería, Universidad de Los Andes, Mérida, Venezuela. Correo electrónico: jormany@ula.ve*