UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo a la obtención del título de:
INGENIERO ELECTRÓNICO

TEMA:
IMPLEMENTACIÓN DE UN CONTROLADOR PID CON RASPBERRY MEDIANTE SERVICIO EN LA NUBE APLICADO A UN SISTEMA BALL / BEAM

AUTOR:
DIEGO MIGUEL SOBERÓN PÉREZ

TUTOR:
CARLOS GERMÁN PILLAJO ANGOS

Quito, marzo del 2018

CESIÓN DE DERECHOS DE AUTOR

Yo, Diego Miguel Soberón Pérez, con documento de identificación N° 1714610845, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del trabajo de titulación intitulado: "IMPLEMENTACIÓN DE UN CONTROLADOR PID CON RASPBERRY MEDIANTE SERVICIO EN LA NUBE APLICADO A UN SISTEMA BALL / BEAM", mismo que ha sido desarrollado para optar por el título de INGENIERO ELECTRÓNICO, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago la entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

Diego Miguel Soberón Pérez
C.I.: 1714610845

Quito, marzo del 2018.

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el trabajo de titulación "IMPLEMENTACIÓN DE UN CONTROLADOR PID CON RASPBERRY MEDIANTE SERVICIO EN LA NUBE APLICADO A UN SISTEMA BALL / BEAM", realizado por Diego Miguel Soberón Pérez, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana para ser considerados como trabajo final de titulación.

Quito, marzo del 2018.

Carlos Germán Pillajo Angos
C.I.: 1709255119

DEDICATORIA

A Dios primeramente por permitirme estar en este momento de mi vida, por los logros y momentos difíciles que se presentaron a lo largo de esta carrera y que me enseñaron a valorar y apreciar cada una de las cosas importantes que me han pasado en la vida.

A mi familia, amigos y personas que ahora ya no me acompañan en la tierra, por el apoyo incondicional en todo el trayecto para culminar mi carrera profesional, ayudándome a sobresalir ante los momentos adversos de mi vida estudiantil, por su valiosa guía en mi vida universitaria.

AGRADECIMIENTO

Agradezco a Dios por ser quien ha forjado mi camino y me ha encaminado por el sendero correcto de la vida.

Agradezco a mis padres los cuales han sido un apoyo incondicional en todo momento de mi vida personal y universitaria, son quienes me han ayudado en todos los momentos difíciles de mi vida. Todos mis logros personales se los debo a ellos gracias a su apoyo económico y anímico hacia mi persona.

Agradezco a la Universidad Politécnica Salesiana por haberme aceptado y por abrirme las puertas del conocimiento universitario.

A mi asesor de tesis por su valioso asesoramiento en la realización del proyecto de titulación.

Diego

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE ECUACIONES

ÍNDICE DE TABLAS

RESUMEN

En la actualidad existe una importante tendencia a incorporar servicios en la nube con respecto a áreas de control que tienden a innovar y brindar alternativas de monitoreo, interacción y control, como en este proyecto en el que se implementará un sistema Ball/Bean.

Para el presente trabajo se busca explicar todo lo realizado al implementar un controlador PID a través de un servidor compacto como lo es Raspberry PI 3, con un sensor de distancia, además de un actuador como el ServoMotor que son controlados por un Arduino Mega con una Shield Dragino Yun para la comunicación inalámbrica.

Dentro del Raspberry Pi 3 se tiene un servidor LAMP (Linux, Apache, MYSQL, PHP), donde se programa la adquisición de datos. El cálculo del controlador con la Ecuación 2.2 de la planta. El envió de los datos obtenidos y calculados a una base de datos en un archivo PHP, la encapsulación de datos a través de un método de escritura ligera o intercambio de información como lo es JSON, y finalmente una forma de visualización del comportamiento a través de HTML y JavaScript que obtienen sus datos desde una base de datos MYSQL.

En el Arduino Mega con Shield Dragino Yun se envía los datos del sensor, los cuales se adquieren y escalan con el objetivo de únicamente dar la información requerida. En este caso es la distancia en centímetros(cm), después realiza el proceso de interpretación de los datos calculados que recibe del servidor para dar un ángulo de movimiento al actuador que es un servomotor.

ABSTRACT

Currently there is an important tendency to incorporate services in the cloud with respect to control areas that tend to innovate and provide alternatives for monitoring, interaction and control, as in this project will be implemented to a Ball / Bean system.

For the present work is to explain everything done to implement a PID controller through a compact server such as Raspberry PI 3, with a distance sensor in addition to an actuator such as the Servo Motor that are controlled by an Arduino Mega with a Shield Dragino Yun for wireless communication.

Inside the Raspberry Pi 3 you have a LAMP server (Linux, Apache, MYSQL, PHP), where you program the data acquisition the calculation of the controller with the plant equation, he sent the data obtained and calculated to a base of data in a PHP file, the encapsulation of data through a light writing method or exchange of information such as JSON, and finally a way of visualizing behavior through HTML and JavaScript that obtain their data from a database MYSQL.

In the Arduino Mega with Shield Dragino Yun the sensor data is sent which is acquired and scaled with the aim of only giving the information required in this case is the distance in centimeters (cm), then it performs the process of interpretation of the data calculated that it receives from the server to give an angle of movement to the actuator that is a servomotor.

INTRODUCCIÓN

Usando como punto de partida el proyecto "IMPLEMENTACIÓN DE UN CONTROLADOR PREDICTIVO BASADO EN EVENTOS PARA UN SISTEMA DE CONTROL EN RED INALÁMBRICO" (Pillajo Angos, 2017) donde el controlador es un dispositivo físico que envía los datos en la trama del explorador, el proyecto cambia estos parámetros por un método más veloz y eficiente.

En la actualidad, el campo de la comunicación inalámbrica es extenso por lo que este proyecto busca implementar un proceso de control que use un dispositivo compacto que es el Raspberry PI3 en el cual estará instalado un servidor web con amplia accesibilidad y compatible con los métodos de intercambio de información.

La comunicación inalámbrica se obtiene de un formato ligero de intercambio de datos denominado JSON que significa "(JavaScript Object Notation - Notación de Objetos de JavaScript)" (ecma internacional, 2013) donde se realiza el intercambio de información entre el Raspberrry y la planta que en este caso está constituida por el Arduino Mega y la Shilt Dragino Yun.

La principal ventaja que se puede observar en la comunicación inalámbrica entre la Planta (Arduino Mega con la Shield Draguino Yun) y el servidor (Raspberry Pi 3) es la velocidad en la que se trasmite en una red "LAN (Local Area Network – Red de Área Local) que son las más utilizadas en un intercambio de datos y recursos entre ordenadores de forma inalámbrica" (Prats, 2016), ya que se obtiene una respuesta mejor del proceso sin incluir la información en la trama del explorador y etiquetando la información enviada y recibida que facilita la interpretación.

- **Planteamiento del Problema**

Se desea generar un acceso remoto a una planta Ball / Beam en donde se ejecute un algoritmo PID de lazo cerrado y tenga la posibilidad de emular el hardware y obtener un acceso a los algoritmos de control implementados en la infraestructura de la nube.

Para este caso en particular se desea implementar los servicios de la nube como plataforma (PaaS), misma que requiere el pago de una cuota mensual la cual depende del equilibrador de carga y del servicio de copias de seguridad.

La herramienta a usar es SmarterASP.NET que es una plataforma de almacenamiento la cual es compatible con múltiples lenguajes de programación como PHP que puede ser un lenguaje base para programar el controlador en la Raspberry Pi3.

Una vez establecido el procesamiento o virtualizado del controlador Set Point a través de SmarterASP.NET se proveerá un enlace con dirección IP y nombre de dominio que conectará con la planta Ball / Bean. Se pretende ejecutar el sistema PID realimentándolo desde los servicios definidos en la nube.

Actualmente, existe una importante tendencia a incorporar servicios en la nube para explotar los recursos que ésta tecnología brinda hacia las áreas de control y así generar más aplicaciones afines o pensar en otros proyectos innovadores que usen esta tecnología que está siendo orientada a brindar alternativas de monitoreo, interacción y control de sistemas industriales a través del internet.

- **Objetivo General**

Desarrollar un controlador PID de lazo cerrado con servicios en la nube a través del Raspberry.

- **Objetivos Específicos:**

  Investigar la metodología de los servicios en la nube para la implementación de algoritmos matemáticos.

  Diseñar y construir la red para el sistema Ball / Beam para integrar los servicios de la nube en la red.

  Generar pruebas de campo para obtener datos de monitoreo del algoritmo PID.

  Realizar una guía para el funcionamiento e implementación de la plataforma en la nube y el entorno de la planta Ball / Beam.

- **Justificación**

En la actualidad la industria se está orientando a procesos inalámbricos donde no sea necesario tener un operario presente y pueda ser ejecutado desde cualquier parte del mundo tan solo con tener acceso a la nube.

El tema propuesto de estudio permitirá ejecutar un controlador PID virtualizado en la nube y aplicarlo a una planta Ball / Beam teniendo en cuenta los sistemas orientados a la nube donde se utilizarán los recursos que da este tipo de arquitecturas de servicios en tiempo real con énfasis en una comunicación basada en estándares de retardos admisibles para un sistema como es el Ball / Beam.

Se busca implementar un PID de lazo cerrado con acceso a la infraestructura de la nube parar proporcionar un servicio tipo (CaaS). Actualmente, esta operación se realiza a través de un dispositivo físico que permite definir los parámetros iniciales (set point) del PID. La idea principal es virtualizar este controlador para simplificar y centralizar los recursos de procesamiento en la nube sin tener que contar con el dispositivo físico de *set point*, estableciendo los parámetros a través de esta tecnología que para este tipo de controles se encuentra aún en fases iniciales y no ha sido explotada.

Este proyecto está orientado a comprobar que usando los servicios en la nube, el proyecto tendrá un ahorro de costos en hardware que es significativo, sobre todo, si se logran virtualizar los dispositivos físicos; sin embargo, se deben realizar otros estudios para solventar los problemas de efectividad de respuesta que presentan los enlaces debido a los retardos propios de estas tecnologías.

- **Beneficiarios**

Se beneficiará el área de aplicaciones inalámbricas reduciendo de esa manera el cableado entre equipos ya que el controlador desaparece, aplicando el CaaS (Control como Servicio) el cual es aplicado a una planta didáctica la cual también tiene una conexión inalámbrica y el intercambio de información entre la nube o servidor y la planta se realiza mediante JSON, orientando así la automatización a controladores inalámbricos que puedan ejecutar algoritmos de control y orientándose como servicio en la nube.

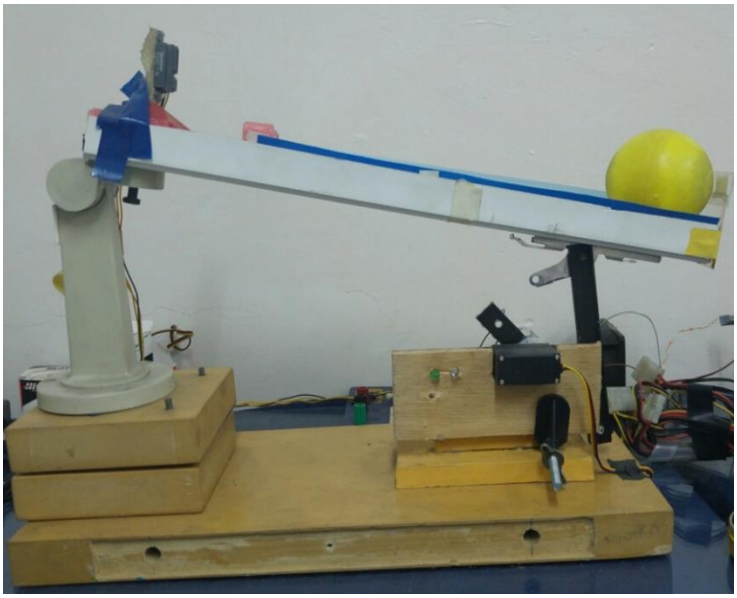
CAPÍTULO 1
MARCO TEÓRICO

En esta sección se detalla las características y conceptos de un intercambio de información entre dispositivos los cuales fueron aplicados a lo largo del proyecto realizado.

1.1 Sistema Ball/Beam

El sistema físico consiste en una barra que pivotee sobre una articulación central, en la cual hay un sensor de distancia que proporciona la posición de la bola en tiempo real. Un servomotor unido a la barra controla el ángulo de giro del mismo sobre las coordenadas x & y como se muestra en la Figura 1.1. (Fernandez, 2013)

Figura 1.1: Sistema Ball/Beam



Planta para estabilizar la bola en el centro, Elaborado por: Diego Soberon

1.1.1 Logística del sistema

La arquitectura del sistema se describe en componentes lógicos en el cual se constituye para realizar el controlador virtual basado en servicios colaborativos del algoritmo que realiza el proceso de recepción, cálculo y envió.

- **Agentes orientados a servicios para la industria colaborativa**

Los sistemas multi-agente orientados a los servicios (SoMAS) son una combinación de las características fundamentales de los servicios orientados y multi-agente en una nueva plataforma para la automatización industrial. Un sistema SoMAS es compuesto principalmente por recursos compartidos en forma de servicios solicitando agentes o dispositivos en las plataformas de automatización industrial. (Marco Mendes) (Maria Ganzha, 2014)

1.1.2 Modelo matemático

El modelo matemático que se va aplicar al proyecto es del PID que es el valor medio de errores, proporcionales que están referenciados con el *Set Point* o punto de referencia con el cual se trabaja, el error integral relacionado con el error anterior o error precedente y el error derivativo el cual esta referenciado con respecto al tiempo que puede predecir un comportamiento futuro del error. (Morilla, 2006)

- **Modelo matemático orientado al mercado computacional en la nube**

La computación en nube es un enfoque emergente para una infraestructura de recursos informáticos en los que grandes sistemas o nubes están conectados entre sí a través de Internet para proporcionar servicios y asignar trabajos a las máquinas virtuales. (K.Mukherjee, 2010) (Shucheng Yu, 2010)

1.1.3 Análisis de la interfaz para el sistema Ball/Beam

En el presente trabajo se describe el desarrollado de la interfaz realizada en PHP, en donde la interfaz principal se desarrolla para servir de interacción entre el usuario y el controlador donde se pondrán los parámetros del PID para estabilizar la bola de la planta; siempre se tiene valores iniciales con los cuales el controlador se estabiliza, pero también se puede modificar el *Set Point* o punto de referencia para demostrar el funcionamiento.

Una vez que se tiene los valores iniciales se procede a obtener el dato de distancia que es enviado por el sensor y actuador inteligente a través de un sistema de intercambio de información por objetos que envía y recibe en pares, respectivamente, que lleva etiqueta y su valor respectivo.

Para que el proceso se pueda visualizar en tiempo real se necesita obtener la información recibida y retenerla por breves instantes de esa manera los algoritmos tienen tiempo para dibujar el comportamiento.

La primera manera de visualizar en tiempo real es a través de HighChart, este proceso es enviar los datos recibidos y calculados a una base de datos para después de terminado el proceso se llama de la base de datos de los puntos para dibujarlos a través de la librería.

La segunda manera de visualizar, en tiempo real, es a través de Canvas que es el desarrollo de una librería de extensión (js) el cual a través de códigos de dibujo matemático se procede a realizar las curvas de comportamiento y se retiene de esa forma la información en el código PHP a través de JavaScript que realiza procesos en subdivisiones de un cuerpo general que contiene los valores establecidos, y dibuja el comportamiento sin necesidad de actualizar la interfaz de adquisición y envío como se realizaba antes de esta librería donde no se podía modificar los datos en tiempo real y no usar más de una interfaz, previamente se ingresaban los datos a una interfaz independiente para enviarla y realizar el procesos PID y al ser actualizada cada segundo, la interfaz de proceso no se puede apreciar correctamente por el comportamiento del sistema.

1.2 Dispositivos implementados

Se considera dispositivos inteligentes a la Raspberry Pi3 y Arduino con la Shield Dragino Yun que componen los dos dispositivos inteligentes con acceso a redes inalámbricas y que soporten un proceso de intercambio de información como lo es el JSON ya que los dos dispositivos lo soportan, también al ser de wnvio y acceso web lo programado tanto en el Raspberry (Servidor local) y Dragino Yun (Cliente) pueden ser exportados a la web. Lo programado en el servidor puede ser cargado en la Nube para remplazar el servidor local y las funcionalidades no cambiarían ni tendrían que obtener una configuración adicional, únicamente cambiar la dirección ip a la cual se envié y reciba la información.

1.2.1 Servidor Raspberry Pi3

Raspberry es un computador compacto de placa reducida como se muestra en la Figura1.2 de tecnología SMD(Tecnología de Montaje Superficial), también se puede considerar parcialmente como hardware libre ya que no todos los componentes del Raspberry son considerados libres. Están bajo contrato, pues está orientada a la educación informática, actualmente, utilizada en muchos servicios de IoT(Internet of Things o internet de las cosas). El propósito de este dispositivo es más académico y de desarrollo informático y robótico actualmente. (RASPBERRY PI FOUNDATION)

Usa software que si se puede considerar libre es una adaptación de *Debian* un sistema más ligero para poder trabajar con una tarjeta micro sd de 8gb por lo mínimo y se conoce como raspbian en este caso del proyecto es Raspbian Jessi 2017 (RASPBERRY PI FOUNDATION)

Figura 1.2: Raspberry Pi3

Placa fisica del dispositivo, Autor: (RASPBERRY PI FOUNDATION)

1.2.2 Cliente

El cliente es un dispositivo conjunto ya que por sí solo los elementos de este dispositivo no funcionarían ya que el Shield Dragino Yun es un complemento del Arduino que mejora tanto su procesamiento como su memoria interna para soportar la interacción inalámbrica entre el servidor y la planta a la cual la planta viene a ser sensor y actuador inalámbrico.

- **Aduino Mega**

Se trata de una tarjeta controladora como se muestra en la Figura 1.3 basada en los Atmegas que tiene varias aplicaciones actualmente es un dispositivo de fácil manejo y programacion con grandes prestaciones, compatible con software y hadware libre como son los shields que están diseñados para estos dispositivos y que facilitan la comunicación entre dispositivos.

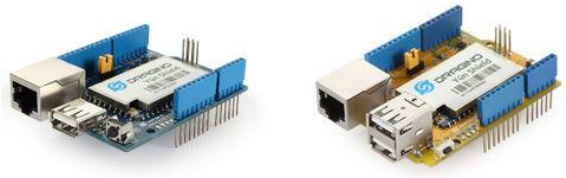
Figura 1.3: Arduino Mega 2560

Descripción físico del dispositivo, Autor: (SpikenzieLabs, 2017)

- **Dragino Yun**

Es un dispositivo que se muestra en la Figura 1.4 desarrollado para resolver los problemas de conexión y almacenamiento que tiene el Arduino por sí solo es compatible con todos las placas Arduino de la gama que existe y su software es compatible con IDE de Arduino ya que tiene un software de código abierto que se ejecuta en el Shield para configurar el mismo, este dispositivo es potente ya que existen aplicaciones web en las cuales, al mismo dispositivo se lo puede usar como servidor o cliente se puede cargar páginas básicas por su gran capacidad de almacenamiento superior al Arduino, su versatilidad ya que se puede conectar a la red en la que se trabaja localmente de forma inalámbrica

Figura 1.4: Shield Dragino Yun



Referencia física de la shield dragino yun, Autor: (Dragino, 2017)

1.3 Servicios para la nube

Los servicios en los cuales se basa para transferir de un servidor local a un servidor externo se toma en cuenta el servicio PASS que está orientado a la automatización ya que busca optimizar la infraestructura y el software.

1.3.1 Plataforma como servicio (PASS)

La PaaS se refiere al conjunto de herramientas y servicios que vuelven rápida y eficiente la codificación y la implementación de aplicaciones, por lo que ayuda a optimizar la interacción entre infraestructura y software de servicios, se obtiene ventajas de copias de seguridad automáticas de las bases de datos junto con los lenguajes específicos de programación o entornos de desarrollo. (Suempresa.cloud, 2016)

- **Arquitectura basada en SOA**

En el ámbito de la automatización industrial, impulsado principalmente por necesidades de agilidad y apoyadas por nuevas tecnologías, la arquitectura (SOA) es el virtualizado de servicios expuestos por sus componentes claves (hardware y software). (Stamatis Karnouskos)

1.4 Herramientas para implementar algoritmos a través de la nube

EL controlador es un sistema Web el cual al ser desarrollado en PHP tiene características Web las cuales al ser exteriorizadas o exportadas a la nube no tiene un cambio brusco del controlador lo único que se realiza para que funcione adecuadamente el controlador es remplazar el archivo cliente,json y crearlo en el dominio así la información es adquirida del dominio y no del servidor local.

1.4.1 PHP (Hypertext Preprocessor)

Es un lenguaje de programación para desarrollo web “PHP (acrónimo recursivo de PHP: Hypertext Preprocessor)” (The PHP GROUP, 2001) es un lenguaje de procesos desarrollado para el lado del servidor ya que tiene características de simplicidad para los programadores y puede ser desarrollado también en HTML. (CodeCademi company, 2017)

Usamos la programación de PHP para desarrollar la desencapsulación de la información que le llega al servidor para posteriormente enviarla a un archivo de texto (archivo.txt) ya que con la configuración inicial del Raspberry Pi3 la lectura y escritura de un archivo de texto es fácil solo se habilitan los permisos de lectura y escritura de (archivo.txt). (The PHP GROUP, 2001)

Con la programación PHP se ejecutan comandos de almacenamiento en la base de datos mientras se procesa la desencapsulación y el cálculo del PID para guardar la información en tiempo real tanto de la distancia que obtiene y la respuesta del cálculo realizado y con eso podemos usar los datos para graficar el comportamiento del sistema. (CodeCademi company, 2017)

1.4.2 Base de datos MySQL

La base de datos es una forma estructurada de organizar y guardar la información con la que se trabaja con un grado de seguridad en la cual impide la copia de los datos de un ordenador a otro o descargas de estos archivos organizados. (The PHP GROUP, 2001)

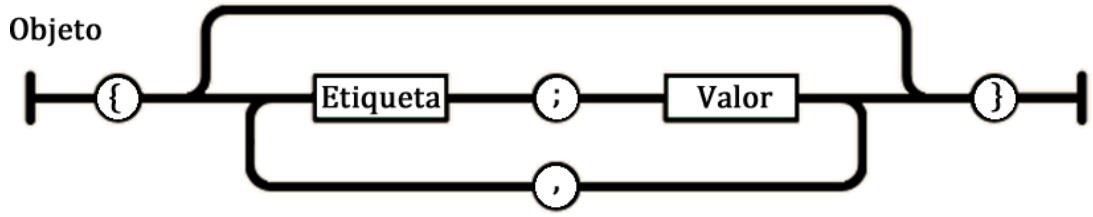
Se trabaja con esta estructura para realizar las gráficas del comportamiento del sistema con el que trabajamos ya que la información que se obtiene en la página PHP es guardada en orden de llegada se organiza para graficar de forma adecuada el comportamiento del sistema usando la configuración TimeStamp que es una marca temporal la cual usa un formato de fecha y hora y lo guarda en el archivo organizado por orden de llegada. (The PHP GROUP, 2001)

1.4.3 JSON (Notacion de objetos de JavaScript)

Es una forma de serializar la información como lo son números, objetos, boleanos, etc. Está basado en el lenguaje de programación JavaScript que es ligero, de fácil interpretación para humanos y aún más para máquinas. Esto sirve para intercambiar información por su facilidad de escritura y lectura. (ecma internacional, 2013)

JSON tiene dos estructuras, estructuras de pares que se muestra en la Figura 1.5 y estructura de lista, en el proyecto usamos la estructura de pares que consiste en nombre y valor, el nombre de la variable distancia y obtener la información, él envió de la respuesta con el ángulo de inclinación del servomotor. (ecma internacional, 2013)

Figura 1.5: Estructura de objeto JSON



Descripción de la trama JSON, Autor: **(ecma internacional, 2013)**

1.5 Librerías para la interfaz de usuario

Se usa una interfaz amigable para el controlador y así destacar en forma real el comportamiento del sensor de distancia que busca estabilizar en el punto de referencia a través del ServoMotor y observar cuando existe la conexión y la desconexión entre los equipos a parte de la señalización luminosa.

1.5.1 Highchart

Es “API (Application Programming Interface - Interfaz de Programación de Aplicaciones): método que permite a un programador utilizar un software para generar diferentes aplicaciones” la arquitectura de la información es comunitaria orientada de lo que es la Web2.0. (Cobos, 2006)

En este caso se uso esta herramienta ya que es interactiva y genera de forma gráfica el comportamiento de los datos que se desencapsulan y se procesan en el interior de la página Web y guardada en la base de datos. (Highcharts, 2017)

1.5.2 Canvas

Se usa para graficar sobre la página web en ejecución o tiempo real que usa JavaScript para realizar los espacios de trabajo que interactúan independientemente de la página base y sirve para funciones de dibujo tales como cuadros, textos, círculos, etc. (Highcharts, 2017)

Para el desarrollo del proyecto se lo usa para graficar la respuesta del sensor y de forma gráfica observar el comportamiento de todo el proceso que se realiza en el controlador que es el servidor Raspberry Pi3 y la Nube.

CAPÍTULO 2
DISEÑO DE IMPLEMENTACIÓN

En la presente sección se destaca el diseño de las páginas, como la comunicación JSON y el diseño de los requerimientos para el controlador y ejecución del sistema Ball/Beam.

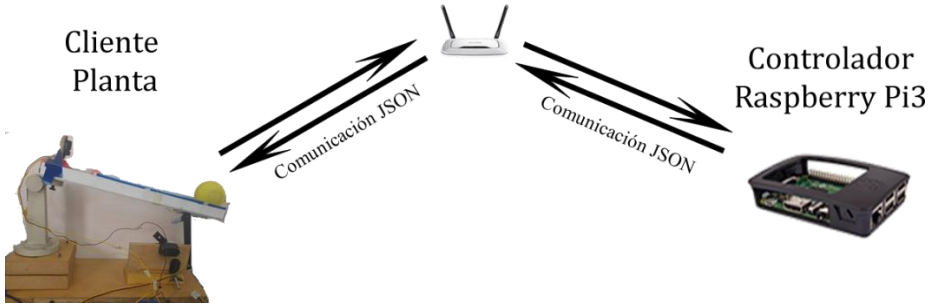
2.1 Diseño conceptual

Se describe el diseño realizado a lo largo del proyecto para cumplir los objetivos planteados, el cual es diseñar un controlador que tenga la capacidad de ejecutarse y obtener la información de forma remota e inalámbrica y va a tener una forma local y forma remota a través de SmarterAPS.NET que es un proveedor de servicios que proporciona un alojamiento temporal en la nube para el proyecto.

2.1.1 Requerimientos de Usuario

El usuario busca tener control y monitoreo de un controlador PID en una planta Ball/Bean como se muestra en la Figura 2.1 la cual está conectada de forma inalámbrica instalado en una red LAN eliminando los elementos guiados, descartando el cableado entre equipos existe una mayor potencia y movilidad de los sensores y actuadores aplicada a la planta construida para demostración didáctica.

Figura 2.1: Esquema Didáctico del Proyecto



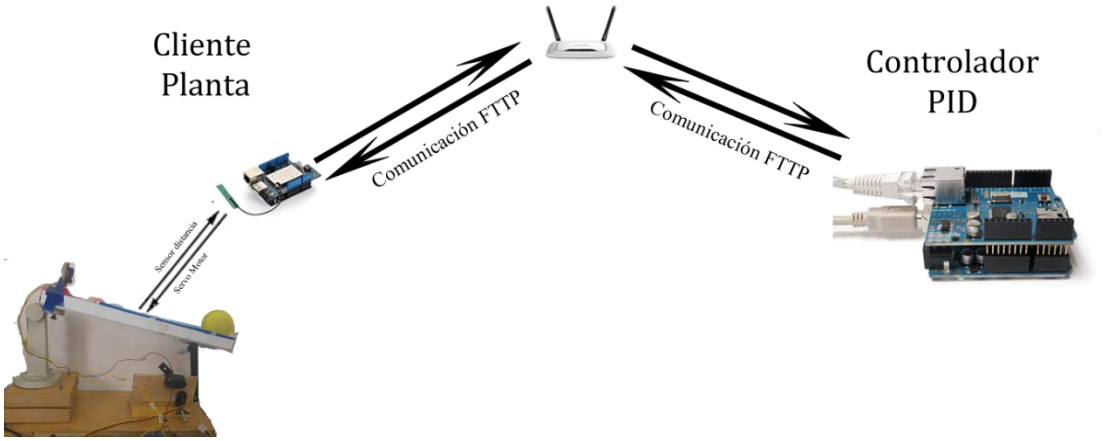
Elementos usados en el proyecto, Elaborado por: Diego Soberon

2.1.2 Descripción del sistema implementado

Se busca la implementación de un controlador PID generando una alternativa al controlador físico realizado en el proyecto “Diseño e implementación de un

Sistema de Control en Red, utilizando sensores y actuadores inteligentes para los mecanismos de control: PID, PWM y Control On/off, monitoreados mediante una aplicación por la nube.” (Bonilla Agualongo, 2016), el cual es el predecesor del trabajo actual, el controlador se compone de un Arduino Mega con una Shield Ethernet que permite enviar la información del algoritmo del control PID mediante la trama FTTP, información que se envía al actuador localizado en la planta constituida por el Arduino Yun en donde la trama de información se descompone y se obtiene el dato para ejecutar el actuador sobre el ángulo de disparo para estabilizar la bola en el centro como se ve en la Figura2.2.

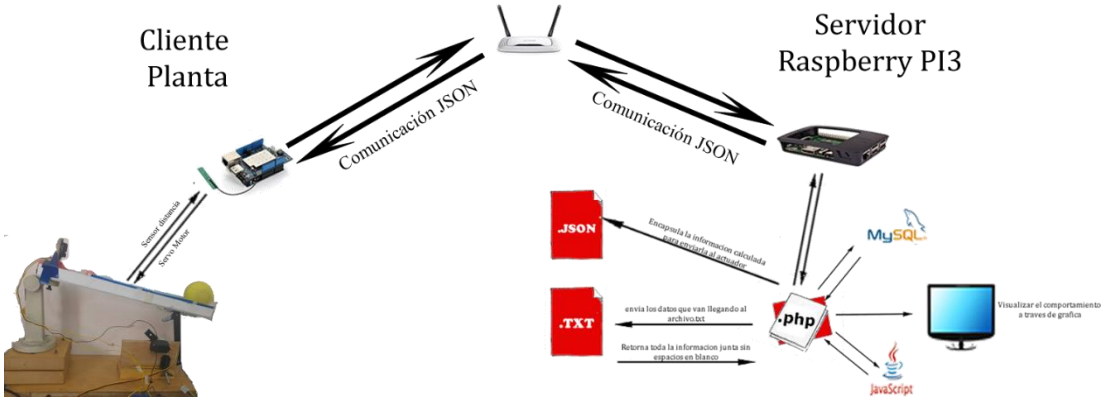
Figura 2.2: Esquema del trabajo "Diseño e Implementación de un Sistema de Control en Red"



Elementos del proyecto predecesor, Elaborado por: Diego Soberon

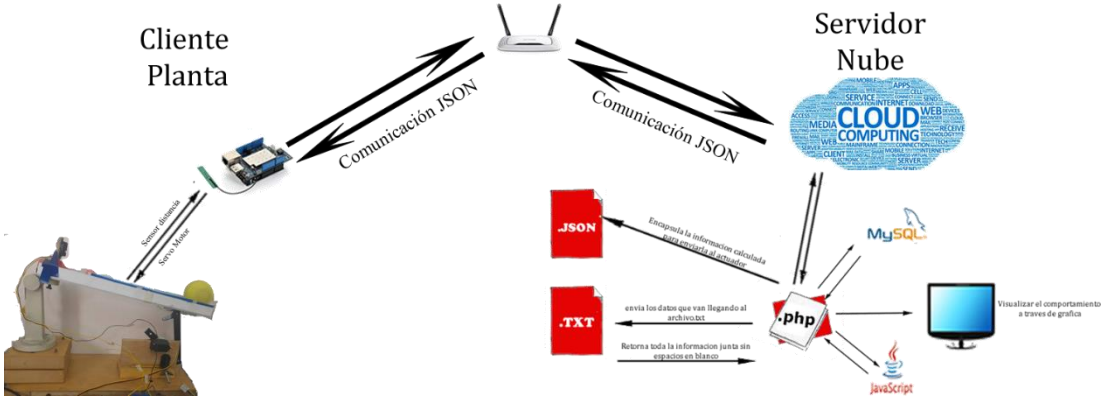
El presente trabajo virtualiza el controlador físico el cual puede ser ejecutado de forma local o remota, de forma local se usa un servidor LAMP (Linux,Apache,MySql,Php) en el dispositivo Raspberry Pi3 como se puede apreciar en la Figura 2.3 y de forma remota se usa un alojamiento virtual como se puede apreciar en la Figura 2.4 que se encuentra con el proveedor de servicios que es “SmarterAPS.NET” con dirección IP dada por el proveedor y la dirección DNS contratada independientemente.

Figura 2.3: Esquema de control Local en Raspberry Pi3



Elementos que intervienen de forma local, Elaborado por: Diego Soberon

Figura 2.4: Esquema de control en la Nube



Elementos que intervienen de forma remota o en la nube, Elaborado por: Diego Soberon

La información que envía el controlador virtualizado se realiza mediante el método de intercambio de información JSON, en cambio la decodificación de la información es realizada por el Arduino Mega y la Shield Dragino Yun que está conectada a el sensor y actuador de la planta.

2.1.3 Análisis de Controlador

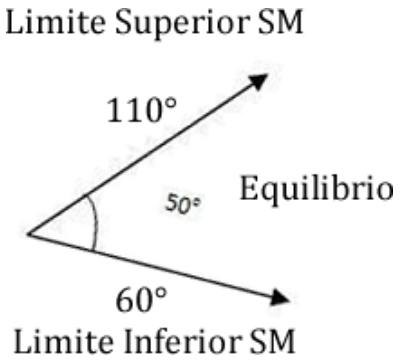
El controlador usado en el proyecto de “Diseño e implementación de un Sistema de Control en Red, utilizando sensores y actuadores inteligentes para los mecanismos de control: PID, PWM y Control On/off, monitoreados mediante una aplicación por la nube.” (Bonilla Agualongo, 2016), es un controlador físico al cual remplazamos por un virtual que ejecutara el algoritmo para equilibrar en el centro la bola.

Se realiza el cambio de controlador usando un servidor local el cual es creado en la Raspberry Pi3 el cual emula el comportamiento de un servidor Linux local servidor LAMP (Linux Apache MySql Php), una vez que se ejecuta el algoritmo PID en forma local, se procede a subir el controlador a un espacio en la nube para observar el comportamiento de forma remota del controlador.

2.2 Diseño del algoritmo predictivo

Se desarrolla una subrutina para el algoritmo predictivo en el cual se busca establecer un ángulo promedio del actuador en 70° para conservar la bola en equilibrio, se desea una inclinación no tan pronunciada para cuando pierda información de la distancia, el algoritmo se ejecute hasta reestablecer la conexión del servidor; es decir el ángulo con que actualiza la ejecución no sea tan inclinado.

Figura 2.5: Ángulos de inclinación



Descripción de los ángulos predictivos usados en el algoritmo, Elaborado por: Diego Soberon

Se busca estabilizar de la siguiente manera con la Ecuación 2.1 descrita a continuación que da un ángulo de disparo entre los 110° y 60° como se muestra en la Figura 2.5, usando de referencia el *SetPoint* para equilibrar el proceso:

Ec. (2.1)

$$Angulo = (2.9695 * Setpoint) + 1.7$$

El proceso para obtener la Ecuación 2.1 de tendencia del comportamiento de la planta es el siguiente: Se adquirió una muestra de datos del ángulo de disparo en base al comportamiento del sistema en el punto de equilibrio, con estos datos de

tendencia obtenidos en Arduino se los traspaso a Excel como se muestra en la Tabla 2.1.

Tabla 2.1: Muestra tomada para obtener la ecuación del algoritmo predictivo

# de muestras	Ángulo de disparo
500	69
501	70
502	71
503	70
504	69
505	70
506	71
507	70
508	69
509	70
510	71
511	70
512	69
513	70
514	71
515	70
516	69
517	70
518	71
519	70

Datos obtenidos para aplicar en la gráfica de comportamiento del predictivo, Elaborado por: Diego Soberón

La muestra tomada se gráfica mediante la herramienta de línea de tendencia obteniendo la Ecuación 2.1 que muestra el comportamiento lineal del algoritmo predictivo es decir se liniealizó como se indica en la Figura 2.6.

Figura 2.6: Gráfica de la muestra tomada para realiza ecuación predictivo

Obtencion de formula para aplicar en el algoritmo predictivo de la planta, Elaborado por: Diego Soberon

El ángulo mostrado está dado por números enteros ya que al obtener el dato generado es proporcionado en punto flotante antes de llegar a Arduino y el ángulo de disparo tienen que ser números enteros para que se ejecute en el ServoMotro.

2.3 Arquitectura de la planta

En el caso de la planta tiene varios elementos con los cuales llegan a actuar como sensor y actuador inalámbrico ya que al estar conectados al Arduino de forma inalámbrica tiene un mejor control que de forma independiente entre el sensor y el actuador.

2.3.1 Arduino Mega

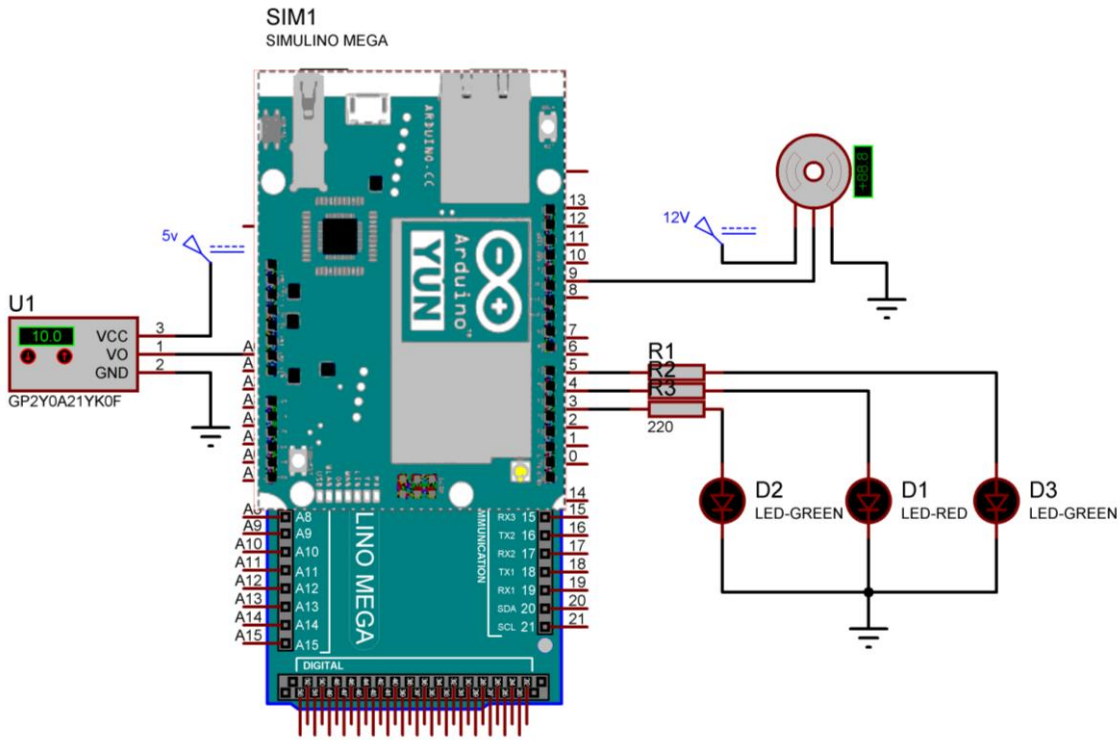
El Arduino Mega es un dispositivo el cual adquiere la información de distancia del sensor Sharp a través de la entrada analógica A0 la cual está incorporada en la tarjeta, y una vez que el controlador calcula el ángulo de inclinación de la barra es ejecutado a través del pin9 que es una salida PWM configurada para el servomotor

2.3.2 Shield Dragino Yun

Este dispositivo permite realizar la comunicación inalámbrica entre el modem y el Arduino Mega para poder enviar la información de distancia que es adquirida del sensor Sharp, también se encarga de recibir la información del ángulo de inclinación de la barra que es enviada del controlador por la red que está representada en la Figura 2.7.

El Arduino y el Dragino Yun se unen en un solo dispositivo que permite la conexión inalámbrica hacia la red, esto permite hacer que el sensor y el actuador sean inalámbricos para obtener los resultados deseados de este proyecto.

Figura 2.7: Estructura de Arduino y Dragino Yun



Esquema electronico aplicado en la planta Ball/Beam, Elaborado por: Diego Soberon

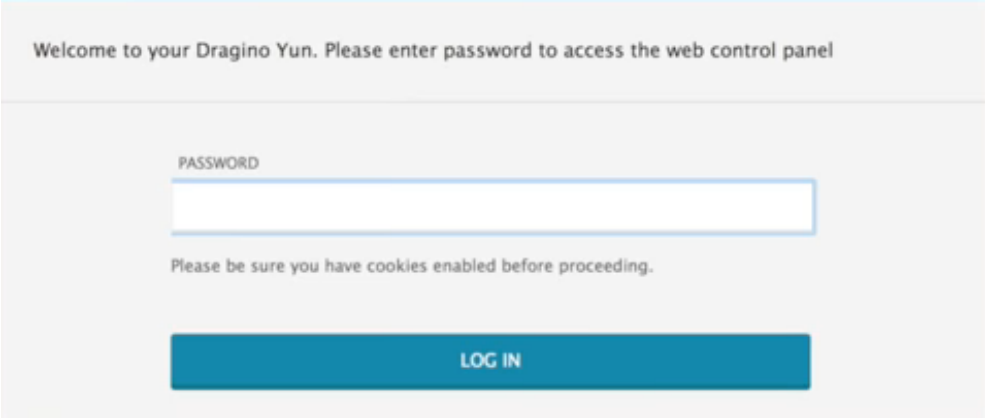
- **Configuración del Dragino Yun**

La configuración general de la Shield Dragino Yun es la siguiente descrita en pasos:

a. Una vez que se conecta el Arduino Mega con la Shield Dragino Yun se alimenta y el dispositivo genera una red inalámbrica.

b. Se busca la red inalámbrica que genera el dispositivo en el explorador de redes, para conectarnos entre la computadora y el Arduino Mega con Dragino Yun.
c. Después de establecer la conexión entre la computadora y el Arduino se procede a abrir cualquier explorador y en la barra de direcciones se pone "192.168.240.1" siempre y cuando la computadora este configurada dentro de la misma red
d. Se presenta la interfaz en la Figura 2.8 en la cual pide contraseña que por defecto del equipo es "dragino"

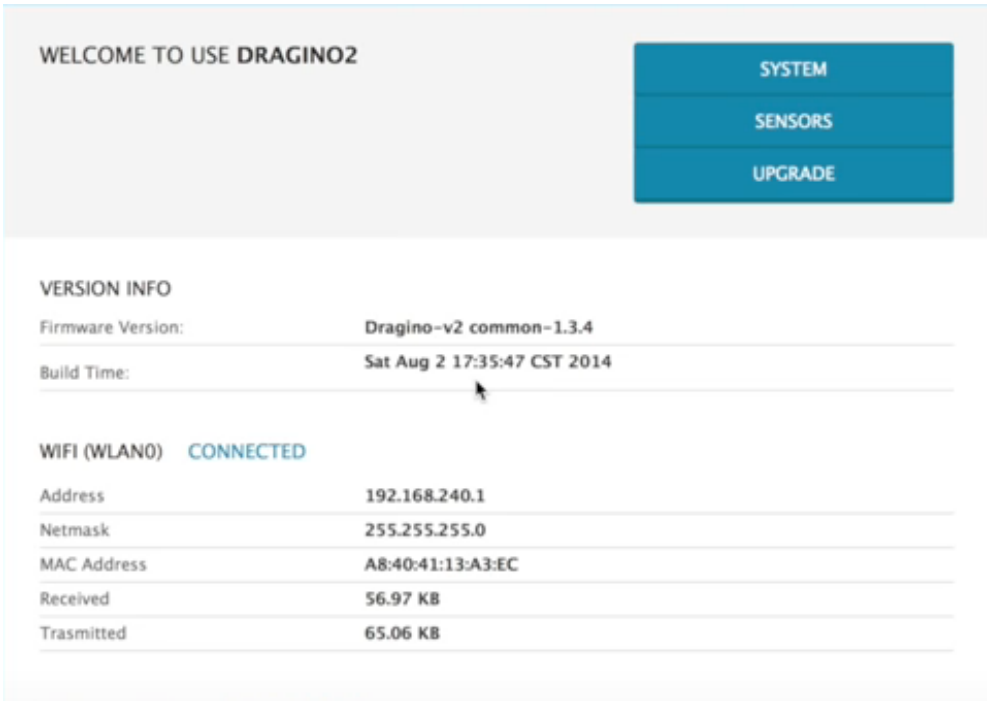
Figura 2.8: Interfaz de Log



Interfaz de inicio para configurar Dragino Yun a la red, Elaborado por: Diego Soberon

e. Procede a la interfaz principal donde escogemos la opción de System que llevan a la siguiente interfaz que se ve en la Figura 2.9

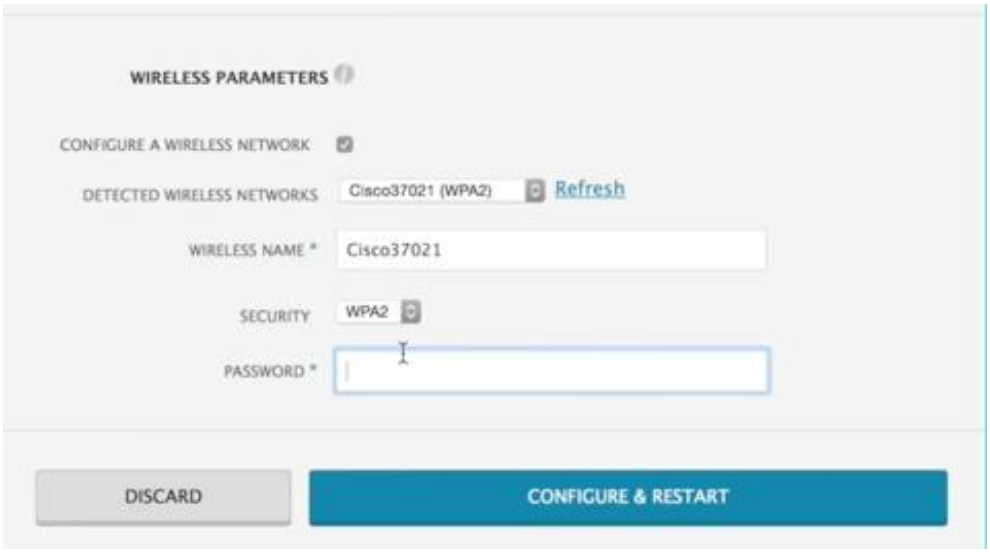
Figura 2.9: Interfaz Principal



Interfaz de descripcion del dispositivo, Elaborado por: Diego Soberon

f. Hay que desplazarse hasta la opción de Wireless Parameters donde se busca el nombre de la red local a la cual se conectará, y en la parte de abajo ponemos el tipo de seguridad que tiene la red y la clave de la misma que se representa en la Figura 2.10.

Figura 2.10: Interfaz de conexión a red



Interfaz paa agregar la red local al Dragino Yun, Elaborado por: Diego Soberon

Ahora para configurar el software de Dragino Yun para implementar una librería JSON que se descargó y se la adiciona al programa Arduino donde la librería se llama “ArduinoJson-master.Zip” se realiza de la siguiente manera la adición de esta librería al programa ArduinoIDE.

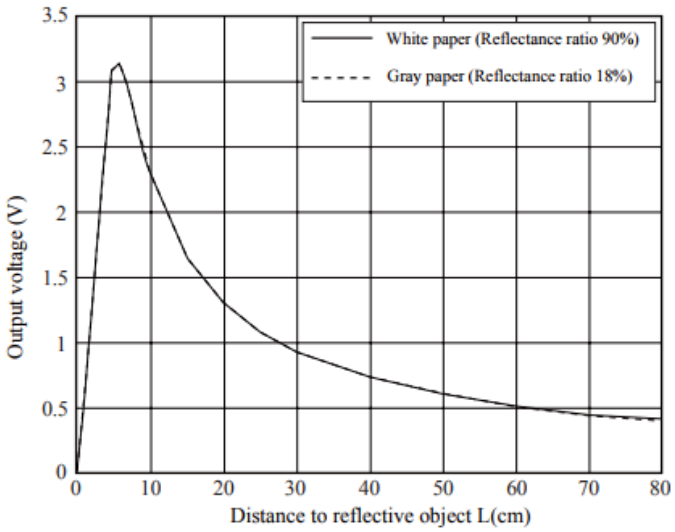
En la pantalla del programa ArduinoIDE se va a en la barra superior a:

- Programa
    - Incluir librería
        - Añadir librería zip
            - Se abre un explorador de archivos y se buscar el archivo “ArduinoJson-master.Zip” donde se encuentre guardado y damos aceptar.

2.3.3 Sensor de distancia Sharp GP2Y0A21YK0F

Este sensor ayuda a medir la distancia a la cual se encuentra ubicada la bola que se busca balancear al centro, como envía una señal de voltaje para determinar la distancia a la que se encuentra el objeto a estabilizar, aunque sea un sensor no lineal como se muestra en la Figura 2.11, es muy preciso y se puede estabilizar la respuesta que genera el sensor con una ecuación exponencial para obtener una respuesta lineal que es la más recomendable y eficiente.

Figura 2.11: Comportamiento del Sensor Sharp GP2Y0A21YK0F



Comportamiento normal del sensor de distancia Sharp, Autor: (Pololu, 2017)

Se usa una ecuación para linealizar la distancia, pero en la cual solo usamos una distancia de 10 cm que es la distancia en donde el sensor empieza a tener un voltaje que tiende a ser lineal, y 30 cm que es la longitud total de la barra.

2.3.4 Servo Motor HS-311

El actuador de la planta es un Servomotor que se muestra en la Figura 2.12 el cual recibe una señal PWM(Pulse Width Modulation) que le da un ángulo de inclinación el cual es procesado en la placa Arduino una vez recibido la respuesta del controlador se usa ángulos de inclinación entre 110° y 60° son las condiciones de programación a las cuales está sometido el actuador para estabilizar el proceso la ventaja de trabajar con el servomotor y el controlador Arduino, se establece en grados la respuesta del actuador para no realizar ningún tratamiento, mapeo o escala de señal en la ejecución, precisa más el movimiento de la barra para que la bola se situé en el centro de la barra.

Figura 2.12:ServoMotor HS-311

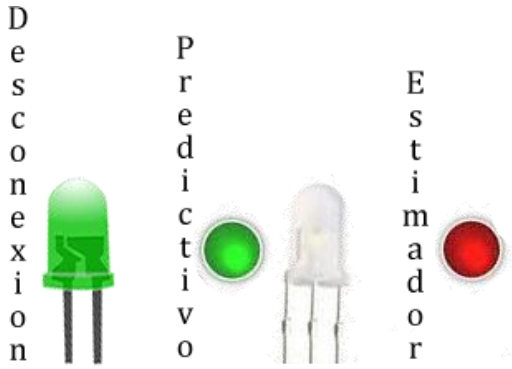
Gráfico del motor a usarse para dar el ángulo de inclinación en la planta, Autor: **(Hitecrcd, 2017)**

2.3.5 Indicadores

Se incluye indicadores de estado para identificar en que acción se encuentra la planta ya que al ser un servicio en la nube o un servicio local se tiene latencias y hasta desconexiones momentáneas en las que deben entrar un código de estabilización de procesos que pueda sostener el proceso hasta que retorne la comunicación perdida a eso se llama predictivo los cuales son representado por

leds de colores para referenciar a estos modos funcionamiento en el proceso como se representa en la Figura2.13.

Figura 2.13: Indicadores visuales de proceso



Descripcion de los indicadores led inplementados en la planta Ball/Bean, Elaborado por: Diego Soberon

Se usa un led normal de color verde que representa la desconexión que es interpretada por el Arduino que no tiene ninguna dirección IP a la cual enviar o recibir datos.

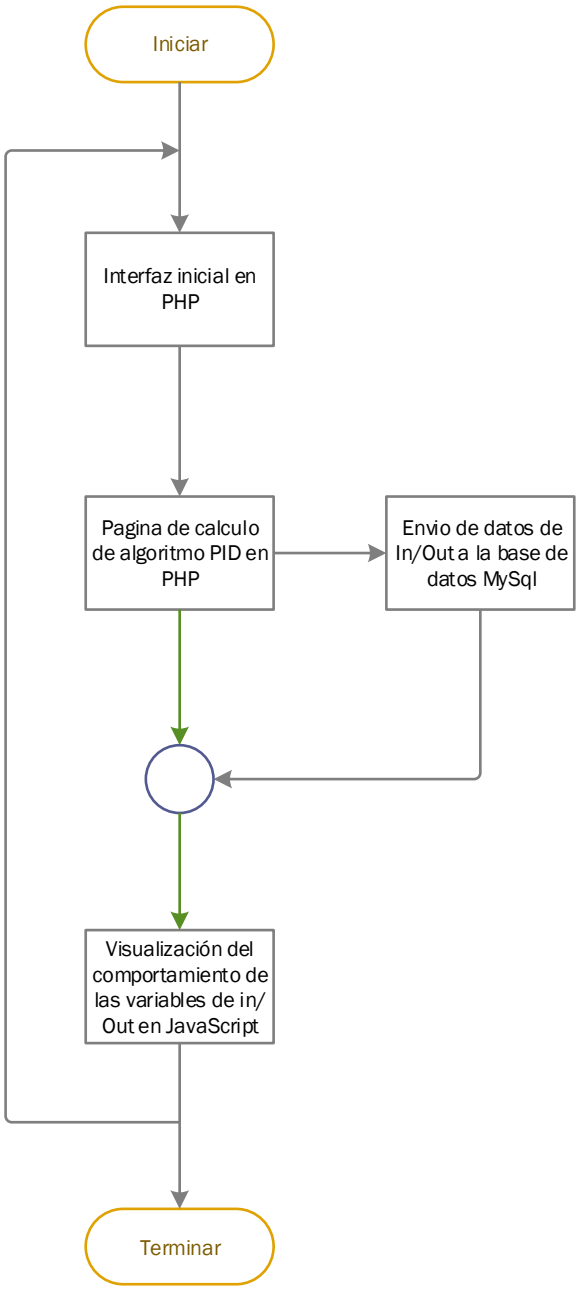
También se usa un led bicolor el cual al encender el led de color verde representa el predictivo que es mayo a 70º es considerado ángulo positivo por el diseñador, cuando se enciende el led de color rojo representa el estimador que es menor a 70º es considerado ángulo negativo por el diseñador.

2.4 Esquema de diseño del sistema ball/beam

El controlador esta implementado sobre un servidor Linux el cual trabaja con servicios libres y de fácil configuración ya que no necesita tantos permisos como lo usa el Windows, pero al ser libre es más vulnerable a ataques del exterior.

Se usa un interpretador de código ligero por defecto que es el “Geany Programmer´s Editor” soporta varios lenguajes de programación como PHP, JavaScript con los cuales se desarrolla el controlador que usaremos en el proyecto que se describe en el siguiente flujograma.

Figura 2.14: Flujograma de Controlador



Flujograma de comportamiento general del controlador virtual, Elaborado por: Diego Soberon

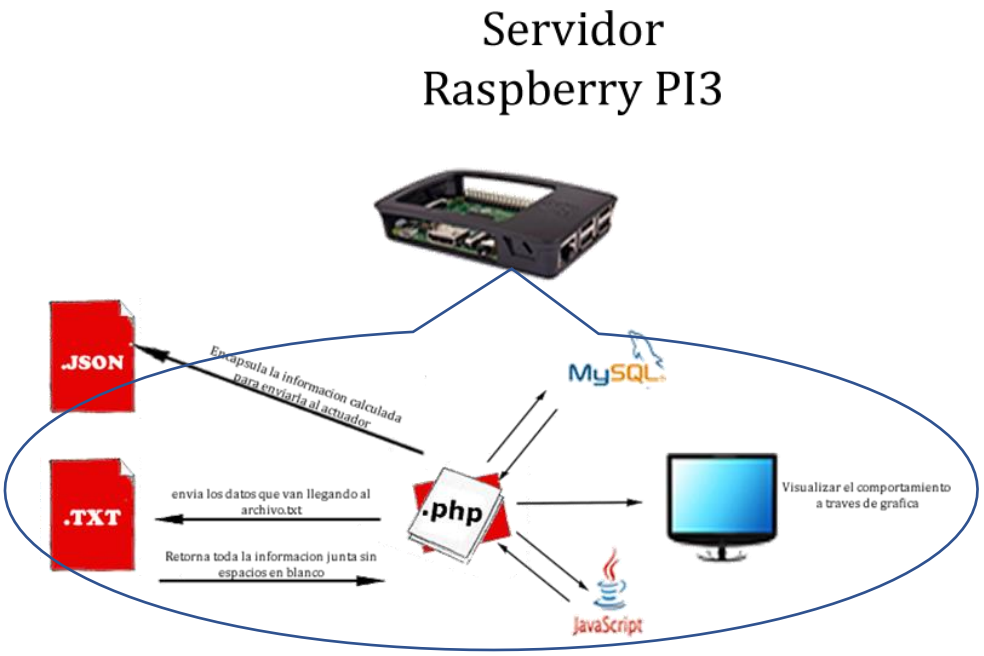
El flujograma de la Figura 2.14 se describe el funcionamiento global del controlador que se implementó en el servidor, en donde se puede observar la ejecución del proceso que se realiza en el controlador, el cual adquiere datos de entrada/salida(In/Out) que ejecuta el algoritmo PID para la panta Bean/Ball y visualiza el comportamiento de las variables a través de una interfaz gráfica realizada en JavaScript.

2.4.1 Esquema de Funcionamiento Local

Para el funcionamiento local se utilizó un Raspberry Pi3 como servidor en el que se ejecuta el siguiente procedimiento que se muestra en la Figura 2.15:

a. En la página principal se adquiere los datos obtenidos del sensor que se almacena temporalmente en un archivo tipo *.txt
b. Se adquiere el dato almacenado en el archivo temporal anterior y se realiza el cálculo del algoritmo PID
c. Se envía el dato resultante del PID a la base de datos, para posteriormente graficarlo
d. Se realiza el traslado de los datos mediante el formato de intercambio de información JSON
e. Se virtualiza el comportamiento de las variables de entrada que es el sensor y la variable de salida que es la respuesta del algoritmo PID a través de una interfaz gráfica creada en JavaScript

Figura 2.15: Esquema de Controlador Local



Elementos que intervienen en el controlador de forma local, Elaborado por: Diego Soberon

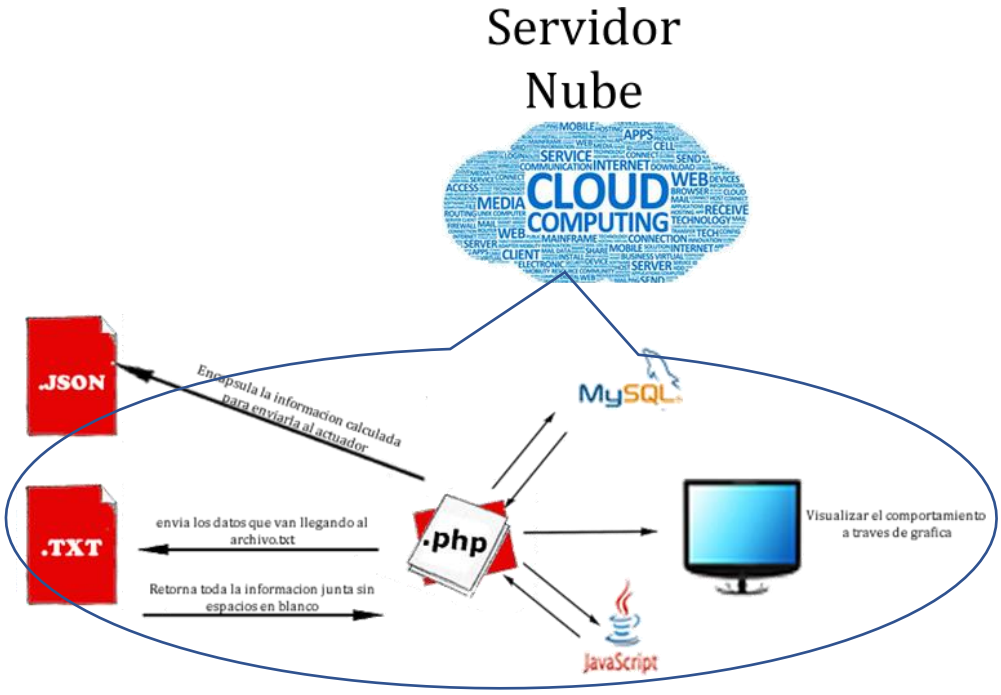
2.4.2 Esquema de Funcionamiento en la Nube

Para el funcionamiento remoto se utilizó un alojamiento en la nube en el cual se ejecuta el siguiente procedimiento que se muestra en la Figura 2.16:

a. En la página principal se adquiere los datos obtenidos del sensor que se almacena temporalmente en un archivo tipo *.txt
b. Se adquiere el dato almacenado en el archivo temporal anterior y se realiza el cálculo del algoritmo PID
c. Se envía el dato resultante del PID a la base de datos, para posteriormente graficarlo
d. Se realiza el traslado de los datos mediante el formato de intercambio de información JSON
e. Se virtualiza el comportamiento de las variables de entrada que es el sensor y la variable de salida que es la respuesta del algoritmo PID a través de una interfaz gráfica creada en JavaScript

Al llevar el controlador a la nube se debe trasladar los archivos PHP, JSON y JS para que el resto de elementos que se necesita se los crea en el proveedor de servicios SmarterASP.NET ya que al trasladarlos del servidor local al servidor remoto los archivos no se ejecutan.

Figura 2.16: Esquema de Controlador en la Nube



Elementos que intervienen en el controlador de forma remota o nube, Elaborado por: Diego Soberon

Existen restricciones las cuales el proveedor de servicios al tener un contrato temporal o de prueba tiene limitaciones la cual es que no se puede subir la base de datos para almacenar la información, no es de gran impacto en el controlador ya que la base de datos es un respaldo de información para generar reportes gráficos independientes.

CAPÍTULO 3
PROCEDIMIENTO PARA LA IMPLEMENTACION DEL SISTEMA

Este capítulo se describe el procedimiento que se utilizó para la implementación del sistema en donde se expone el esquema del controlador programado en el servidor, las simulaciones del comportamiento de la planta y la implementación de los algoritmos tanto para el cliente como para el servidor.

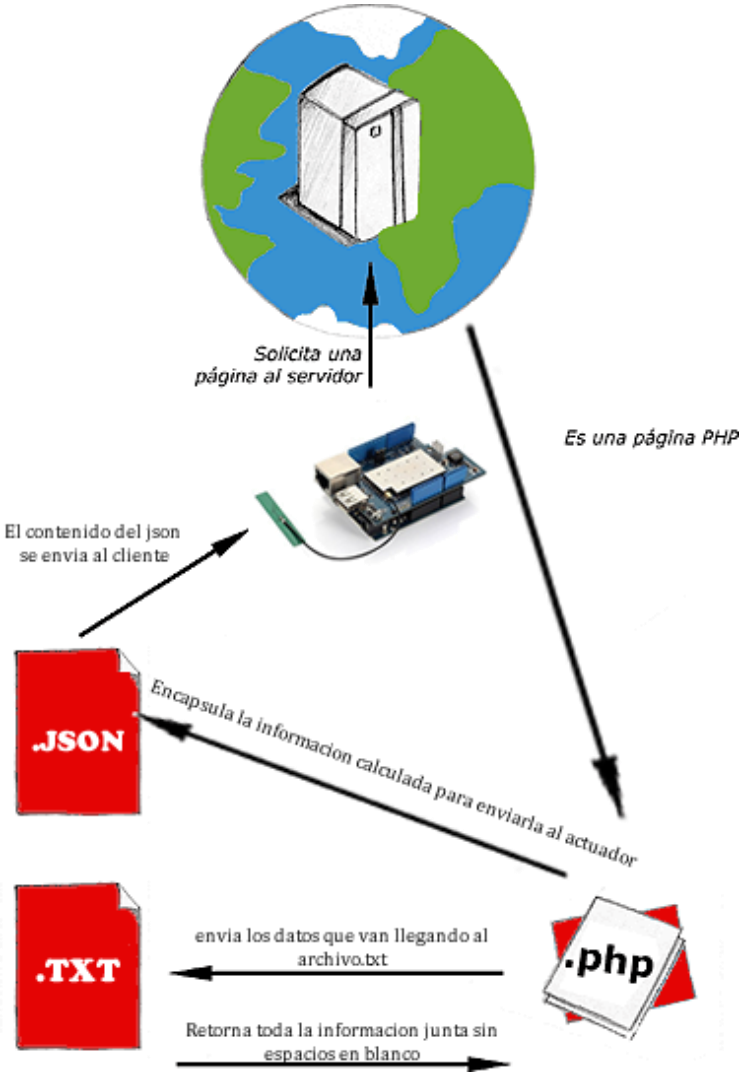
3.1 Esquema en PHP y MYSQL

La programación en PHP es la más eficiente y utilizada para trabajar con base de datos ya que se tiene un manual con comandos de funcionamiento y ejemplos en línea funcionales de este proceso de guardar y obtener datos. Al trabajar con PHP también se trabaja con un lenguaje adaptativo que es muy compatible con diferentes lenguajes de programación como lo son HTML y JavaScript, además de que puede trabajar como lenguaje web en el caso del proyecto se lo usa para:

a. La comunicación comienza con la petición del cliente que es el Arduino Mega con la Shield Dragino Yun donde envía los datos al servidor y recibe la respuesta del PID en cero
b. Se establece la comunicación con el servidor entre la interfaz principal y el usuario
c. En la interfaz principal se envía los datos a un archivo temporal como van llegando
d. Del archivo temporal se recibe lo almacenado se decodifica la información y se obtiene los datos de distancia del sensor para realizar el cálculo del controlador PID
e. Después de realizar los cálculos del controlador PID se realiza el mapeo de la respuesta para expresarlo en ángulo y enviarlo a través del método de intercambio de información JSON

El lenguaje de proceso para realizar la adquisición, calculo y envió de datos a la estructura JSON se muestra en la Figura 3.1.

Figura 3.1: Comportamiento de las paginas PHP



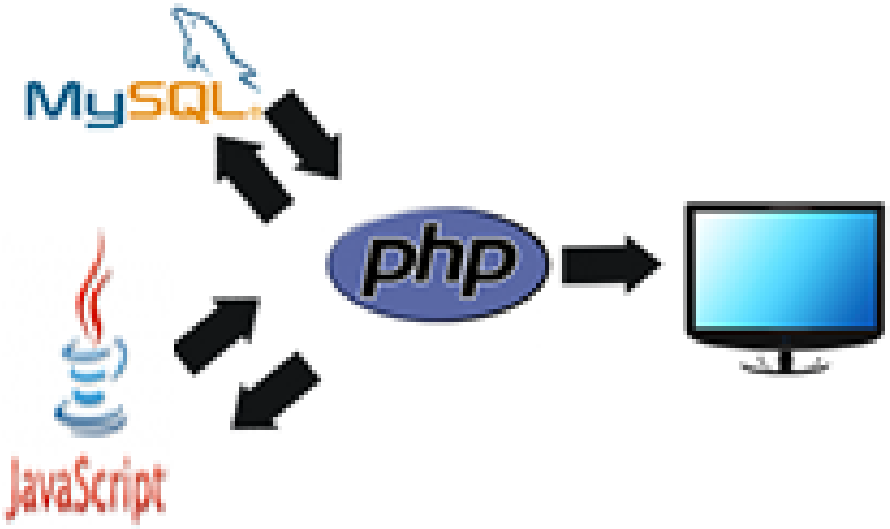
Funcionamiento y elementos que intervienen con la interfaz principal, Elaborado por: Diego Soberon

El lenguaje adaptativo para trabajar con la interfaz de envío de datos y gráfica la respuesta del comportamiento del proceso que puede implementar al algoritmo en lenguaje PHP y puede combinar HTML y JavaScript en la interfaz final.

En la Figura3.2 se describe el comportamiento entre los diferentes lenguajes utilizados para la implementación, como se puede observar existe una comunicación bilateral entre MySql y PHP para efectos de realizar consultas de los datos almacenados en la base de datos, también existe librerías que permiten realizar reportes gráficos de manera amigable. Cabe recalcar que las librerías son

editadas en JavaScript. En este proyecto se utilizó la librería de HighChart la cual se editó para realizar la visualización de las variables de entrada/salida (In/Out).

Figura 3.2: Comportamiento de Visualización



Elementos que intervienen en la visualizacion del controlador PID, Elaborado por: Diego Soberon

a. Cuando la interfaz principal entra en funcionamiento inmediatamente establece una comunicación PHP y MySql.
b. En MySql se obtienen los datos de la variable de entrada que es el sensor y la variable de salida que es la respuesta del PID.
c. En la interfaz de presentación se puede graficar los datos almacenados en la base de datos MySql para entender el comportamiento del sistema y observar cómo se estabiliza.
d. Sirve usar JavaScript para realizar la interpretación de la información y generar los reportes gráficos.

3.1.1 Procedimiento de configuración del LAMP (Linux, Apache, Mysql, Php)

Para la implementación de este sistema se usa los siguientes comandos para crear un servidor LAMP(Linux,Apache,MYSQL,PHP) en Raspberry Pi3 usando los siguientes comandos:

Todos los comandos mostrados a continuación son ejecutados en el panel de comandos de Raspbian Jessy 2017. Actualización de la Raspberry antes de instalar los complementos de la LAMP:

- sudo apt-get update - Comando de descarga de paquetes de las paginas designadas.
- sudo apt-get upgrade - Comando de instalación de los paquetes descargados sin dañar la configuración prestablecida del sistema.
- sudo apt-get autoremove - Comando de eliminación de archivos residuales que quedan después de la instalación de las actualizaciones.

Se procede a instalar el LAMP server como se describe a continuación:

- sudo apt-get install apache2 – Comando de descarga e instalacion de los paquetes Apache de servidor local.
- sudo apt-get install mysql-server mysql-client – Comando de descarga e instalación de servidor y cliente MySql.
- sudo apt-get install php5 libapache2-mod-php5 php5-mysql –y – Comando de descarga e instalación de las librerías de comunicación entre Php y MySql.
- sudo service apache2 restart – Comando de reinicio del servicio Apache sin reiniciar el Raspberry Pi3

Se procede a instalar PhpMyAdmin una forma gráfica y agradable para poder ver el proceso de creación de base de datos y tablas:

- sudo apt-get install phpmyadmin –y – Comando de descarga e instalación de la interfaz gráfica de MySql
- sudo nano /etc/apache2/apache2.conf – Comando de configuración para abrir el fichero ejecutable del servidor Apache

Con el comando anterior se abre un fichero de configuración que se copiará el siguiente código para que funcione PhpMyAdmin:

- Include /etc/phpmyadmin/apache.conf

Se pega la línea de código anterior al final del dichero después se presiona control + x a continuación se presiona Y para aceptar los cambios realizados al fichero.

Una vez realizado estos pasos se procede a reiniciar el servidor apache con el siguiente comando:

- sudo service apache2 restart – Comando de reinicio del servicio Apache sin reiniciar el Raspberry Pi3

3.1.2 Configuración de permisos en el servidor

En Raspberry Pi 3 una vez instalado el LAMP se necesita dar permisos de lectura y escritura a todos los archivos que se generen en este servidor para poder ejecutar el control de la planta a continuación se describe los comandos que van en la consola de Raspberry para otorgar los permisos a la carpeta que contiene todo el proceso de control:

- chmod -R 777 /var/www/ - Comando que concede permisos de lectura y escritura a una carpeta completa y todos los elementos que se encuentran dentro del mismo.

A continuación, se detalla el listado de los archivos con los permisos obtenidos si quieren saber que permisos se tiene antes de cambiarlos realizar este paso antes del comando chmod

Para comprobar los permisos se usa los siguientes comandos:

- cd /var/www/html – Comando que lleva a la dirección establecida a través de consola.
- ls –l – Comando que permite desplegar la lista de contenido del fichero dando los atributos de lectura y escritura que tiene activados a través de consola.

3.2 Simulación de funcionamiento del sistema Ball/Beam

La simulación del sistema permite divisar el comportamiento, mediante algoritmos incorporados tanto en el servidor como en la planta, esta simulación permitirá determinar la funcionalidad del sistema.

3.2.1 Simulación cliente – planta

Para simular la planta se implementa una ecuación que representa el modelo del movimiento de la bola y es el que se programó en el arduino, en la siguiente

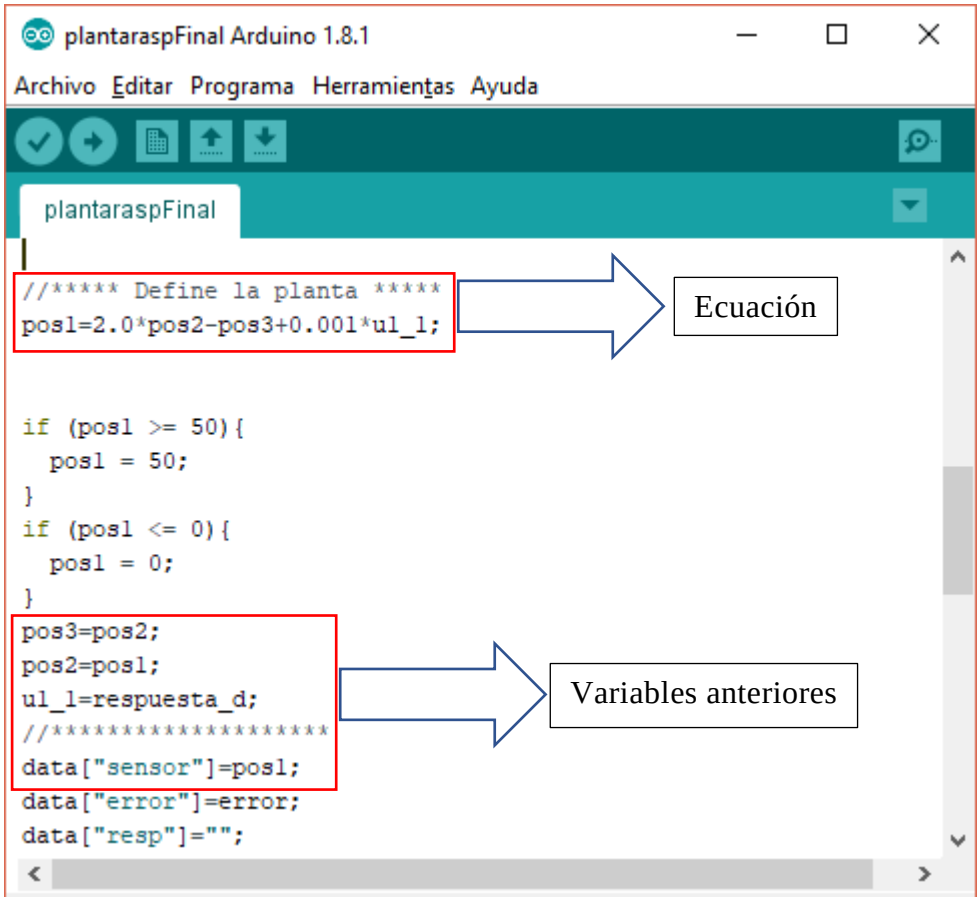
Ecuación 3.1 que se obtiene de la discretización del modelo dinámico de la planta tomada de "IMPLEMENTACIÓN DE UN CONTROLADOR PREDICTIVO BASADO EN EVENTOS PARA UN SISTEMA DE CONTROL EN RED INALÁMBRICO" (Pillajo Angos, 2017, pág. 103). Donde x(n) representa la distancia θ(n) es el ángulo del ServoMotor.

Ec. (3.1)

$$x(n) = 2x(n-1) - x(n-2) + 0.00015\theta(n-1)$$

Para implementar esta ecuación se necesita, guardar dos posiciones anteriores de la variable de entrada distancia en posiciones de memoria diferentes y también se utiliza la respuesta obtenida del controlador anteriormente en el código de arduino que se muestra en la Figura3.3:

Figura 3.3: Implementación de ecuación



Implementacion del modelo dinamico en el algoritmo de la planta, Elaborado por: Diego Soberon

3.2.2 Simulación servidor – controlador

Para la implementación del algoritmo de control PID se utiliza una ecuación que representa su comportamiento, este algoritmo es muy utilizado en controladores industriales los cuales son acoplados físicamente, pero en este proyecto se implementó en lenguaje PHP, como se indica a continuación:

Ec. (3.2)

$$error = Setpoint - Input$$

error: Es el cálculo del error que es la resta entre el parámetro y la entrada del sensor.

SetPoint: Es un valor ya establecido en el inicio que puede variar.

Input: Es el dato obtenido del sensor que envía la planta.

Ec. (3.3)

$$proporcional = error$$

proporcional: asume el valor del error calculado en la Ecuación 3.2.

Ec. (3.4)

$$integral = integral_{anterior} + \left(\left(\frac{error - error_{anterior}}{2}\right) * tiempo\right)$$

integral: Es un acumulador de errores de área bajo la curva.

integral_anterior: Es el dato anterior de error integral almacenado en un archivo temporal llamado errorint.txt.

error: Es el cálculo de error realizado en la Ecuación 3.2.

error:anterior: el dato anterior del error almacenado en un archivo temporal llamado error.txt.

tiempo: Esta dado por el comando “microtime(true)*10000” que genera el intervalo de tiempo en el cual se adquiere la información y está dado por el equipo donde se encuentra ubicada el controlador virtual.

Ec. (3.5)

$$derivativo = ((error - error_{anteriro})/tiempo)$$

derivativo: es la diferencia del error actual y el error anterior y que es dividido para el tiempo.

error: Es el cálculo de error realizado en la Ecuación 3.2.
error_anterior: el dato anterior del error almacenado en un archivo temporal llamado error.txt.
tiempo: Esta dado por el comando “microtime(true)*10000” que genera el intervalo de tiempo en el cual se adquiere la información y está dado por el equipo donde se encuentra ubicada el controlador virtual.

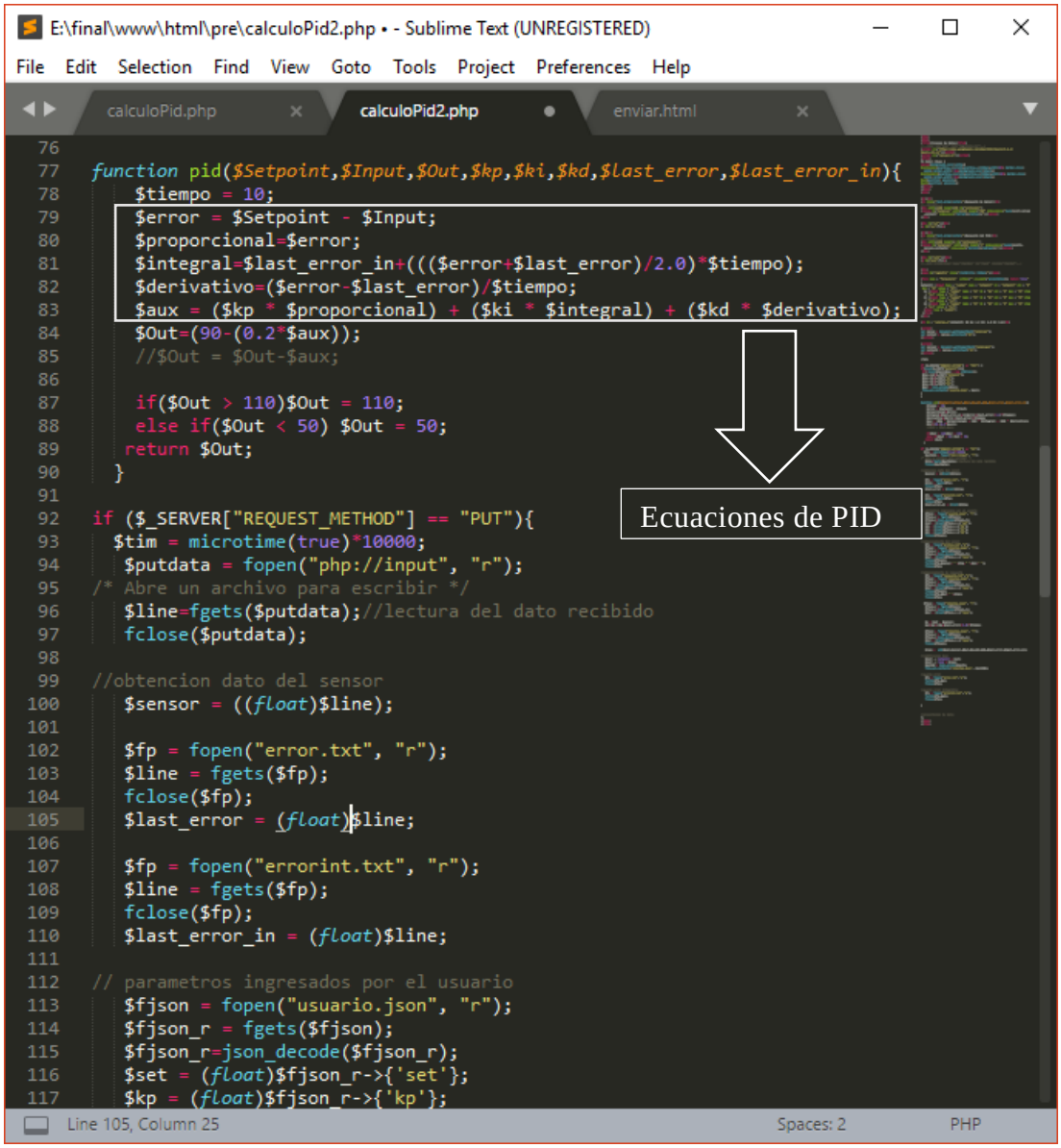
Ec. (3.6)

$$PID = (kp * proporcional) + (ki * integral) + (kd * derivativo)$$

PID: Es la suma de los errores derivativos e integrativos con sus respectivas constantes establecidas en el algoritmo, las constantes fueron obtenidas luego de realizar pruebas empíricas, en base al mejor comportamiento y llegando a la estabilización deseada.
kp: es un valor establecido en la interfaz principal que es igual a 1.5.
proporcional: es la respuesta obtenida en la Ecuación 3.3.
ki: es un valor establecido en la interfaz principal que es igual a 1.0.
integral: es la respuesta obtenida en la Ecuación 3.4.
kd: es un valor establecido en la interfaz principal que es igual a 0.0.
derivativo: es la respuesta obtenida en la Ecuación 3.5.

La descripción del programa se detalla en la Figura 3.4 donde se indica el algoritmo PID y las ecuaciones antes descritas, que son programadas en php como una función, para que se ejecute el proceso correctamente.

Figura 3.4: Implementación PID




```
function pid($Setpoint,$Input,$Out,$kp,$ki,$kd,$last_error,$last_error_in){
    $tiempo = 10;
    $error = $Setpoint - $Input;
    $proporcional=$error;
    $integral=$last_error_in+(($error+$last_error)/2.0)*$tiempo);
    $derivativo=($error-$last_error)/$tiempo;
    $aux = ($kp * $proporcional) + ($ki * $integral) + ($kd * $derivativo);
    $Out=(90-(0.2*$aux));
    //$Out = $Out-$aux;

    if($Out > 110)$Out = 110;
    else if($Out < 50) $Out = 50;
   return $Out;
  }

if ($_SERVER["REQUEST_METHOD"] == "PUT"){
  $tim = microtime(true)*10000;
   $putdata = fopen("php://input", "r");
/* Abre un archivo para escribir */
   $line=fgets($putdata);//lectura del dato recibido
   fclose($putdata);

//obtencion dato del sensor
   $sensor = ((float)$line);

   $fp = fopen("error.txt", "r");
   $line = fgets($fp);
   fclose($fp);
   $last_error = (float)$line;

   $fp = fopen("errorint.txt", "r");
   $line = fgets($fp);
   fclose($fp);
   $last_error_in = (float)$line;

// parametros ingresados por el usuario
   $fjson = fopen("usuario.json", "r");
   $fjson_r = fgets($fjson);
   $fjson_r=json_decode($fjson_r);
   $set = (float)$fjson_r->{'set'};
   $kp = (float)$fjson_r->{'kp'};
```

Implementacion de las ecuaciones PID en el algoritmo del controlador, Elaborado por: Diego Soberon

El error anterior y el error anterior integrativo se deben almacenar, pero al usar el lenguaje de PHP no se puede guardar la información en una variable temporal por lo tanto se debe guardar esta información en archivos temporales como se detalla en la Figura 3.5.

Figura 3.5: Código para guardar información en archivos

```
//guarda error
    $fp = fopen("error.txt","w");
    fwrite($fp,$e);
    fclose($fp);
```

```
//guarda error integrativo
    $fp = fopen("errorint.txt","w");
    fwrite($fp,$ei);
    fclose($fp);
```

Guardar error y error integral en archivos temporales, Elaborado por: Diego Soberon

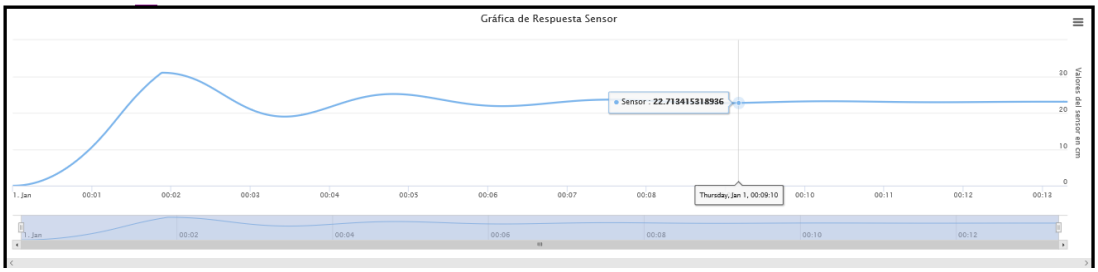
3.2.3 Simulación conjunta del sistema cliente-servidor

Se desarrolló un simulador en el cual se muestra el comportamiento tanto del cliente-sensor y del servidor-actuador, descritos anteriormente en los cuales se implementan el funcionamiento dinámico de la plata mediante su Ecuación 3.1, así como el comportamiento del controlador mediante el algoritmo de control PID, para lo cual se determinó cada una de las constantes Kp,Ki,Kd en base a datos obtenidos de pruebas empíricas del comportamiento real del sistema.

Al implementar la simulación conjunta del sistema cliente-servidor se emuló el comportamiento real del sistema, pero se debe tener en cuenta que en la simulación se tenía valores exactos, no así en la realidad, observando que en la realidad existe un error, que es determinado como aceptable respecto al comportamiento real, esto debido a que en el sistema real se tiene elementos mecánicos que degradan el comportamiento del sistema.

Para probar el emulador desarrollado se procedió a simular el comportamiento del sistema con varios parámetros, que ayuda a determinar las constantes óptimas, de las cuales se obtuvo: SP:20, Kp:1.5, Ki;1.0, Kd:0.0, con estos datos se ingresó en el interfaz principal del emulador, se procedió a la ejecución para observar la evolución de las variables del sistema, como se indica en la siguiente Figura3.6: donde se aprecia el comportamiento del sensor y del proceso que se realiza en el controlador PID mostrándonos el desarrollo de las variable observando que el proceso llega a una estabilidad que está referido en base al error con el cual se desarrolla el algoritmo del controlador.

Figura 3.6: Simulación del Sistema



Comportamiento estimado del sistema Ball/Bean, Elaborado por: Diego Soberon

3.3 Algoritmos implementados

A continuación, se describe el funcionamiento de los algoritmos, tanto en el cliente-planta que tiene el sensor y actuador como en el Servidor-controlador virtual, se detalla en líneas de código paso a paso para que se ejecute adecuadamente el sistema Bean/Ball.

3.3.1 Algoritmos en el cliente

El algoritmo del cliente se lo implemento en el arduino mega como se indicó en la Figura 2.7 del capítulo 2, a continuación, se describe brevemente la implementación del algoritmo de la planta:

a. Se incluyen las librerías, para la comunicación entre arduino mega y la shield dragino yun para usar como un solo dispositivo, la librería para el manejo del ServoMotor por ángulos y la librería de intercambio de información JSON.

b. Se declara e inicializa las variables de calculo que se van a usar durante todo el algoritmo para los cálculos de escalamiento del sensor y disparo del ángulo en el ServoMotor.

c. Se define el modo de salida para los tres pines indicadores luminosos y un pin de salida PWM para ejecutar el ServoMotor.

d. Se inicia el proceso de bucle o lazo donde se ejecutará lo siguiente:

El dato de distancia obtenido por el sensor Sharp GP2Y0A21YK0F a través de una entrada analógica del arduino mega, se procede a escalar del rango de 0 a 1023 dado por conversor ADC incorporado en el arduino a un rango de 10 a 30 centímetros que se ajusta a la realidad del sistema, hay que tomar en cuenta que los

datos convertidos en el arduino son enviados al servidor en formato float o punto flotante, como se puede visualizar en la Figura 3.7

Figura 3.7: Linealizar la entrada analógica

```
sensor=analogRead(0);
voltaje=(sensor*3.3)/1024.0;
Input=26.825*pow(voltaje,-1.227);
```

Codigo que linealiza la respuesta del sensor Sharp, Elaborado por: Diego Soberon

e. Una vez escalada el dato del sensor se procede a consultar si hay información desde el servidor, mediante el comando de la librería json:

"client.getAsynchronously("dirección IP/dirección del proyecto/nombre del archivo.json")"

f. Se ejecuta la comprobación de conexión entre servidor y cliente a través de un contador si el contador llega a un límite, establece que está en desconexión y procede a dar la señal luminosa de desconexión y además el algoritmo predictivo entra en funcionamiento.

g. Si el servidor está disponible se procede a conectar estableciendo la dirección IP o DNS junto con el tamaño de la cadena de caracteres, además que para ejecutar el envío de datos se usa la función de servidor que envía a través de la sentencia PUT, después de ejecutar esta sentencia limpia el espacio de memoria que usa para la transferencia de datos como se muestra en la Figura 3.8.

Figura 3.8: Conexión con servidor

```
char server[]={"http://balancin.ine4c.com/calculoPid2.php"};
char x[256];
void servidor(){
    dtostrf(Input,4,2,x);
    client.put(server,x);
    jsonBuffer.clear();
}
```

Codigo de conexión y envio de datos al servidor, Elaborado por: Diego Soberon

h. El algoritmo predictivo tiene una función independiente que usa la Ecuación 2.1 descrita anteriormente ya que, si llega a perderse la comunicación se llama a esta función en el lazo de ejecución, tal función se indica en la Figura 3.9.

Figura 3.9: Algoritmo predictivo

```
void predictivo(){
  Output = (2.9695 * Setpoint) + 1.7;
  if(Output>100.0)Output = 110.0;
  else if(Output<60.0)Output = 60.0;
}
```

Implementacion de la ecuacion del predictivo en la planta, Elaborado por: Diego Soberon

Al implementar el algoritmo del cliente existe una manera de visualizar el proceso de modo serial el cual se muestra en la Figura 3.10 que tiene los parámetros de envío y recepción de datos que se ejecuta en el cliente, después se trata la información recibida y se muestra el ángulo de disparo que es convertido a número entero y ejecutado por el ServoMotor.

Figura 3.10: Muestra de resultados de la planta

```
dragino-7132a7 at 192.168.0.107 (Arduino Yún)

4294045272
{"Setpoint":23,"resp":110}{"Setpoint":23,"resp":110
110.00
110

4294045464
{"Setpoint":23,"resp":91.845}{"Setpoint":23,"resp":91.845}
91.84
91

4294004252
{"Setpoint":23,"resp":65.646}{"Setpoint":23,"resp":65.646}
70.00
69
```

Datos tomados del serial del Arduino Mega con la Shield Dragino Yun, Elaborado por: Diego Soberon

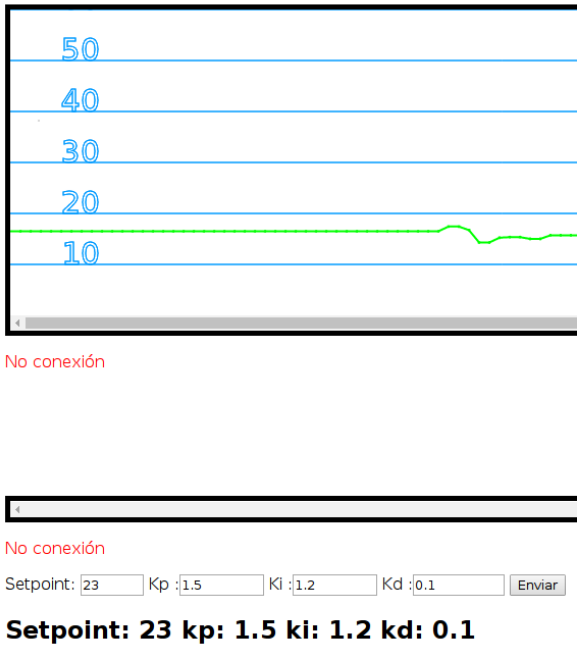
3.3.2 Algoritmos en el Servidor

El algoritmo del servidor se lo implemento de forma local en la Raspberry pi3 una vez terminado se sube el mismo algoritmo al espacio en la nube “SmarterASP.NET” a continuación, se describe brevemente la implementación del algoritmo del servidor:

a. En el algoritmo del servidor se inicia buscando las librerías de gráficos que se implementan en la interfaz principal.

b. La interfaz principal tiene cinco divisiones que permiten ejecutar acciones diferentes en una sola interfaz y se inicializa dándole el tamaño, los colores, la posición de los caracteres a utilizar. En la sección de dibujo, se proporciona la información de distancia de forma gráfica, en la sección de información se proporciona el dato numérico en forma de texto del sensor, la forma de texto de la respuesta que genera el controlador PID, los cuadros de textos para cambiar los valores constantes y las etiquetas de las constantes que se presenta en la última sección. En la Figura 3.11 se presenta la interfaz principal en modo de desconexión.

Figura 3.11: Estado de desconexión del sistema



Interfaz principal en modo de desconexion, Elaborado por: Diego Soberon

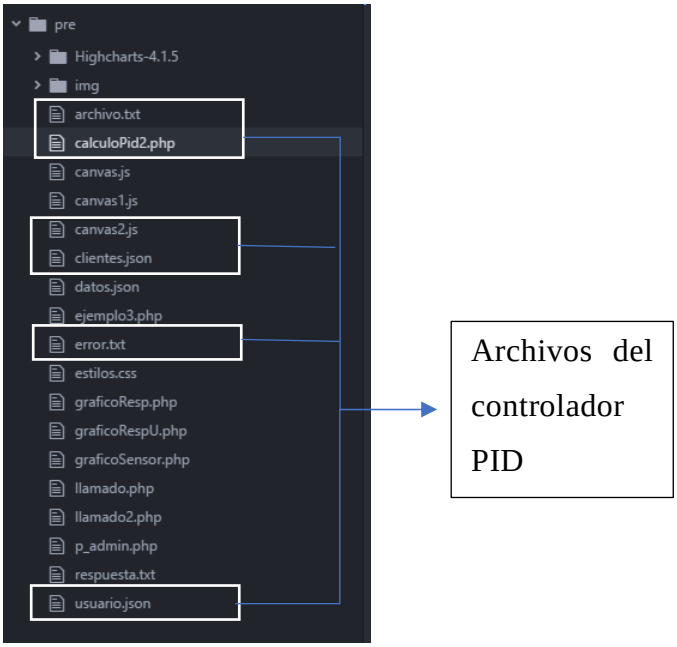
c. Se programa la interfaz gráfica de interacción con el usuario donde se establecen el tamaño del cuadro de respuesta del sensor y el dato que recibe de la planta, a continuación, se presenta el dato del cálculo del controlador PID el cual es presentado en ángulo, además se presentan: los cuadros de entrada de datos y la sección de visualización de los parámetros de configuración para PID datos que fueron enviados al controlador como datos iniciales.

d. Una vez terminada la interfaz de interacción con el usuario se procede a programar la implementación del algoritmo del controlador en lenguaje PHP, el cual obtiene los datos del cliente mediante la sentencia input, una vez que se obtuvo los datos se envían a un archivo temporal mediante formato json, a la vez devuelve el dato de distancia.
e. Se ejecuta el comando "microtime(true)*10000" que obtiene el dato de tiempo que genera la computadora cada vez que llega la información en microsegundos además se procede a ejecutar varias etapas de lectura de archivos temporales que se usan para implementar las ecuaciones 3.2, 3.3, 3.4, 3.5 y 3.6 del controlador PID descritas anteriormente.
f. Una vez que se termina de ejecutar el controlador PID se procede a abrir los archivos de escritura para guardar la información de respuesta para el envío a través del método de intercambio de información JSON y guardar también en los archivos temporales los errores generados para usarlos posteriormente, ya que no se puede guardar en espacios de memorias temporales.

Cuando se ejecuta el algoritmo de manera local se guarda los datos obtenidos del sensor y los datos generados por el controlador PID en la base de datos MySql, En cambio cuando se ejecuta el algoritmo de forma remota o en la nube no se implementó la base de datos, debido a que el espacio proporcionado por el proveedor de servicios no lo permite.

Para describir el comportamiento de interacción de archivos que se ejecuta durante todo el proceso del controlador PID de forma global se puede visualizar en la Figura 3.12, en donde en el archivo de extensión" *.php", se encuentra: el código de programación de interfaz de usuario, la presentación de datos tanto gráfica como textual. También en este archivo se realiza el cálculo de la información obtenida de la planta, que en este caso da el sensor de distancia, esta información se obtiene de archivos temporales de extensión "*.txt" y envía los datos de respuesta del controlador PID por medio de archivos de extensión "*.json"

Figura 3.12: Archivos relacionados al proyecto



Descripcion de los archivos que se implementan en el proyecto, Elaborado por: Diego Soberon

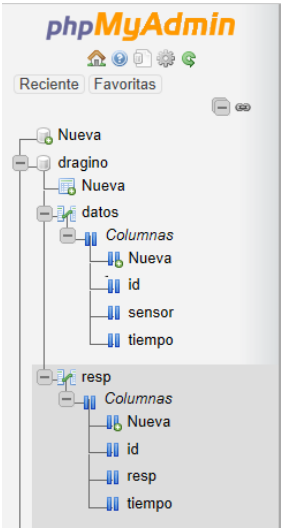
Los archivos descritos en la Figura 3.12 están vinculados a la interfaz principal que es “calculoPid2.php” donde las tramas de información recibida desde la planta son enviados a un archivo temporal el cual es llamado “archivo.txt” y devuelve a la interfaz principal solo el dato de distancia del sensor.

Luego realiza el cálculo del algoritmo del controlador PID, en donde se calcula el error y el error integral, estos datos son guardados en archivos temporales “error.txt” y “errorint.txt” respectivamente, para luego utilizarlos en la función PID programada en la interfaz principal en php. La respuesta del controlador PID está expresada en ángulo que va a disparar el ServoMotor y esta información es enviada al archivo “clientes.json” que realiza el intercambio de información con el Arduino Mega y Shield Dragino Yun.

Este proyecto tiene componente local y remoto, la opción remota culmina solo con los archivos descritos anteriormente, los cuales son subidos al servidor de la nube, para la opción local se utilizar el Raspberry Pi3 como servidor en donde a más de los archivos que ejecutan el proceso descritos anteriormente también envía los datos recibidos y calculados a una base de datos realizada en MySql que permite

respaldar la información y usarla para generar un reporte que puede ser exportado e impreso como se indica en la Figura 3.13.

Figura 3.13: Base de datos MySql forma local



Descipcion general de la base de datos dragino, Elaborado por: Diego Soberon

La base de datos está vinculada a la interfaz principal que es “calculoPid2.php” donde usando sentencias especiales de comunicación entre PHP y MySql se envía a la base de datos la información de la distancia que llega del sensor de la planta y la respuesta del controlador PID que se va a enviar al actuador de la planta.

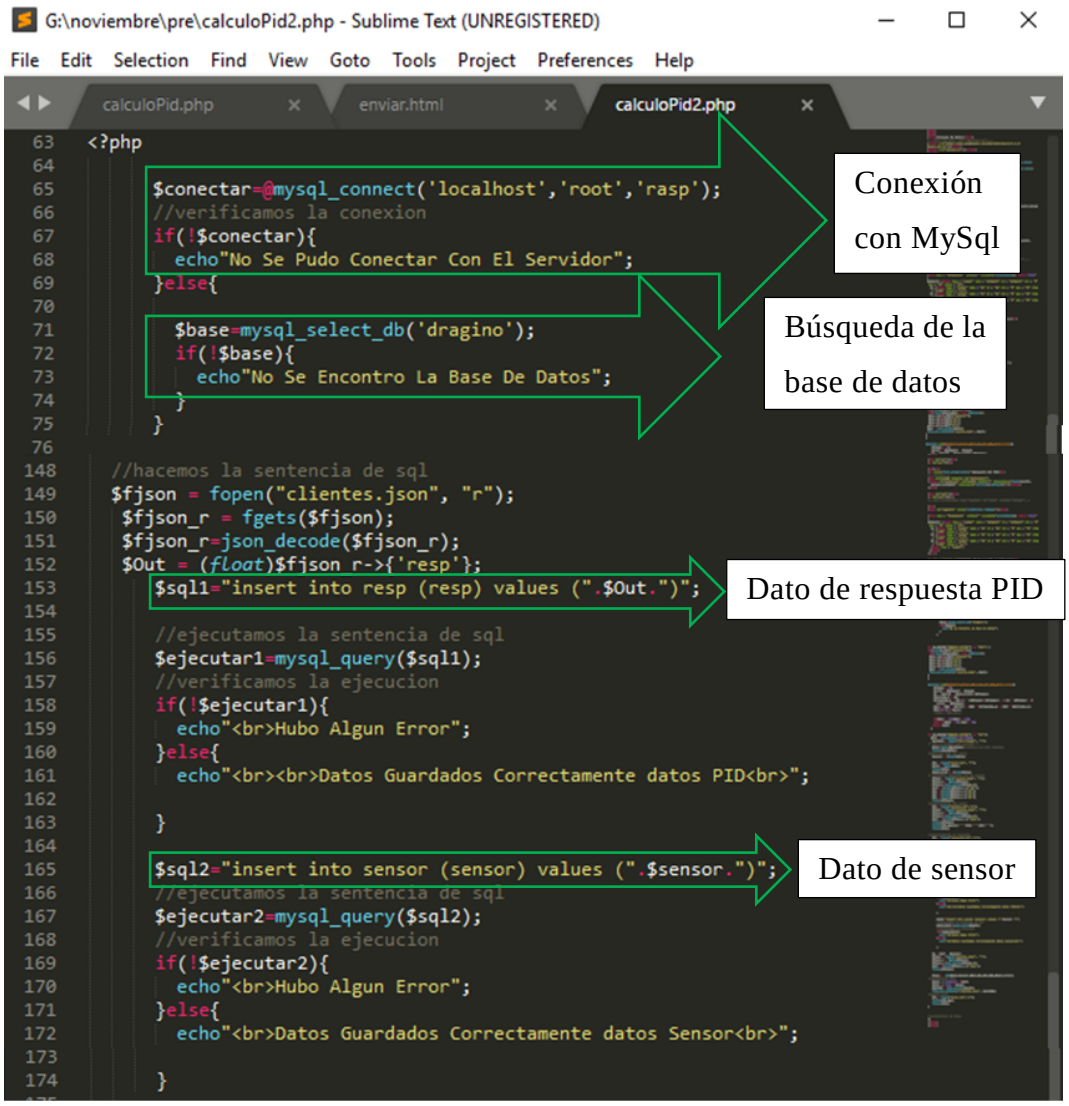
En la Figura 3.14 se procede a explicar de forma global el código implementado para realizar el proceso de conexión y envió de datos entre Php y Mysql:

a. Se conecta a la base de datos MySql con la dirección que es “localhost”, el usuario” root” y la contraseña” rasp”.

b. Al buscar la base de datos tenemos que poner el nombre “dragino” para que establezca la conexión y así poder realizar lectura y escritura de la base de datos.

c. Al tener las condiciones anteriores de conexión con MySql y Php, además de establecer la comunicación con la base de datos dragino se procede a escribir los datos de la respuesta del controlador PID abriendo el archivo que contiene la información para añadirla a la tabla de “resp” de la base de datos, la variable de distancia del sensor que se obtiene de la planta se añade a la tabla de “datos” de la base de datos.

Figura 3.14: Sentencias MySql utilizadas




```
63  <?php
64
65      $conectar=@mysql_connect('localhost','root','rasp');
66      //verificamos la conexion
67      if(!$conectar){
68        echo"No Se Pudo Conectar Con El Servidor";
69      }else{
70
71        $base=mysql_select_db('dragino');
72        if(!$base){
73          echo"No Se Encontro La Base De Datos";
74        }
75      }
76
148   //hacemos la sentencia de sql
149   $fjson = fopen("clientes.json", "r");
150    $fjson_r = fgets($fjson);
151    $fjson_r=json_decode($fjson_r);
152    $Out = (float)$fjson_r->{'resp'};
153      $sql1="insert into resp (resp) values (".$Out.")";
154
155      //ejecutamos la sentencia de sql
156      $ejecutar1=mysql_query($sql1);
157      //verificamos la ejecucion
158      if(!$ejecutar1){
159        echo"<br>Hubo Algun Error";
160      }else{
161        echo"<br><br>Datos Guardados Correctamente datos PID<br>";
162
163      }
164
165      $sql2="insert into sensor (sensor) values (".$sensor.")";
166      //ejecutamos la sentencia de sql
167      $ejecutar2=mysql_query($sql2);
168      //verificamos la ejecucion
169      if(!$ejecutar2){
170        echo"<br>Hubo Algun Error";
171      }else{
172        echo"<br>Datos Guardados Correctamente datos Sensor<br>";
173
174      }
```

Codigos de conexión, envio y consulta de datos de la base de datos MySql, Elaborado por: Diego Soberon

La información que se guardan en la base de datos es únicamente numérica, pero tiene la opción de cada dato guardado adjuntar el tiempo en el que se lo guarda es por eso que al realizar los reportes gráficos la información es completa tanto el dato como la fecha y hora.

CAPÍTULO 4
PRUEBAS DE CAMPO

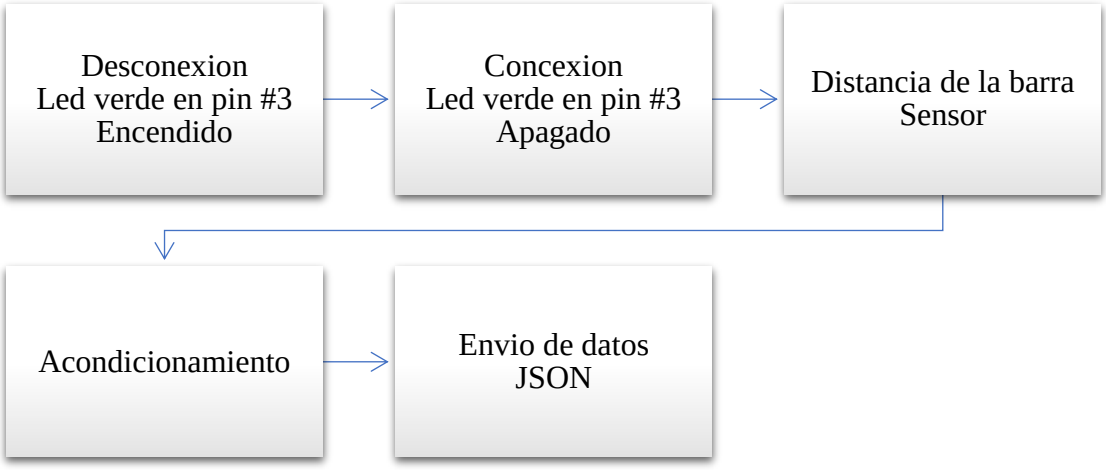
Este capítulo se describe el procedimiento que se utilizó para los ensayos del sistema en donde se expone: el ensayo de forma local y el ensayo de forma remota o en la nube.

4.1Ensayo de forma local

Al realizar las pruebas de forma local los valores iniciales tomados de la simulación que sirve para comparar los resultados del controlador virtual implementado en la Raspberry Pi3, se ejecuta el controlador PID y la respuesta generada por el simulador son parecidos.

En la Figura 4.1 se describe el comportamiento inicial de la planta que va desde el indicador luminoso de desconexión, al envió de datos mediante el método de intercambio de información JSON.

Figura 4.1: Comportamiento normal del sistema

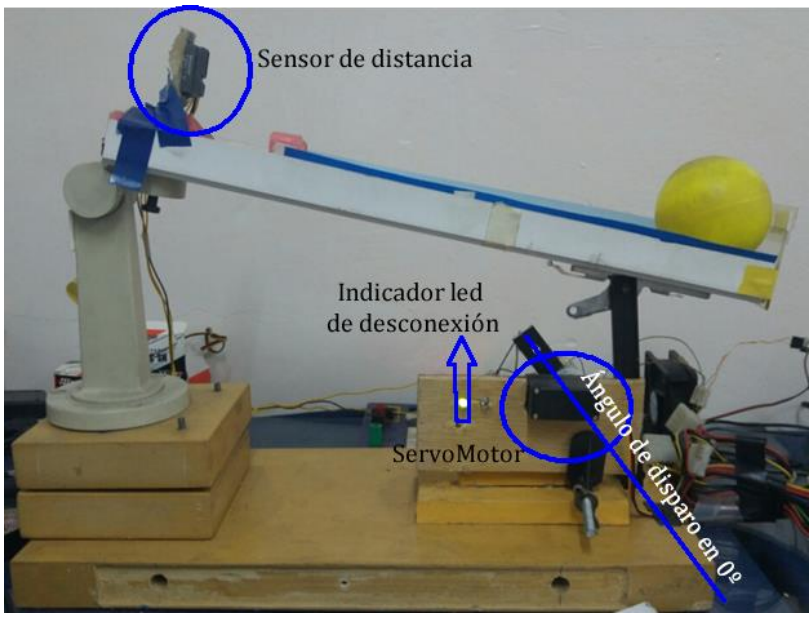


Procedimeinto de la planta para envio de datos, Elaborado por: Diego Soberon

Cuando la planta se encuentra en modo desconexión el led de color verde conectado al pin #3 del Arduino Mega y Shield Dragino Yun se activa y como se muestra en la Figura 4.2, el sensor de distancia da la respuesta en 30 centímetros aproximadamente, el actuador que es el servo motor tiene un ángulo de 0º y la

interfaz principal del controlador establece la no conexión con la planta como se ve en la Figura 4.3.

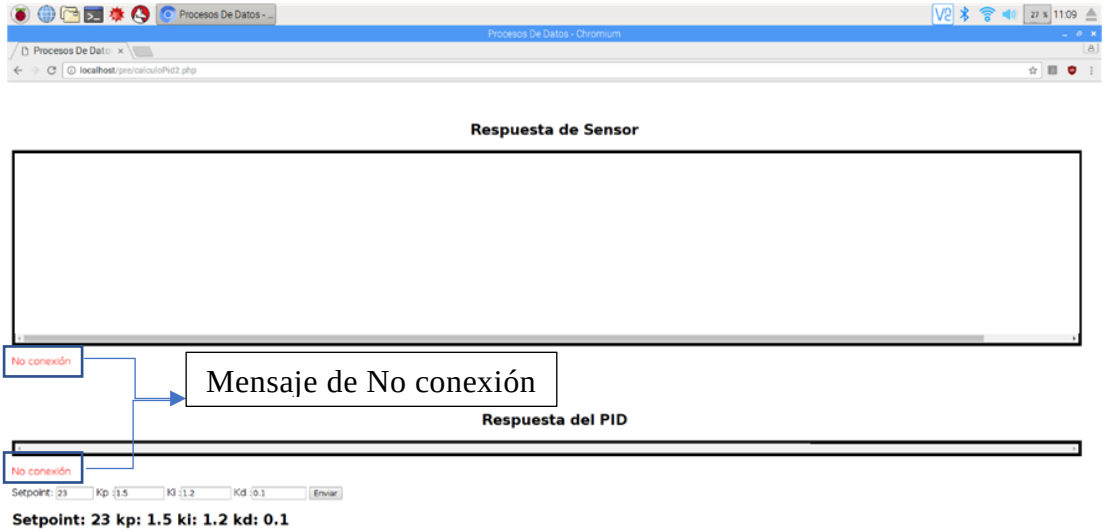
Figura 4.2: Encendido de la planta y modo desconexión



Encendido de la planta en estado de desconexion con el servidor local, Elaborado por: Diego Soberon

En la interfaz principal la desconexión se muestra en las secciones dos y tres que son la expresión numérica o textual de los datos tanto de distancia como de respuesta del controlador PID con el mensaje es "No conexión".

Figura 4.3: Información de desconexión

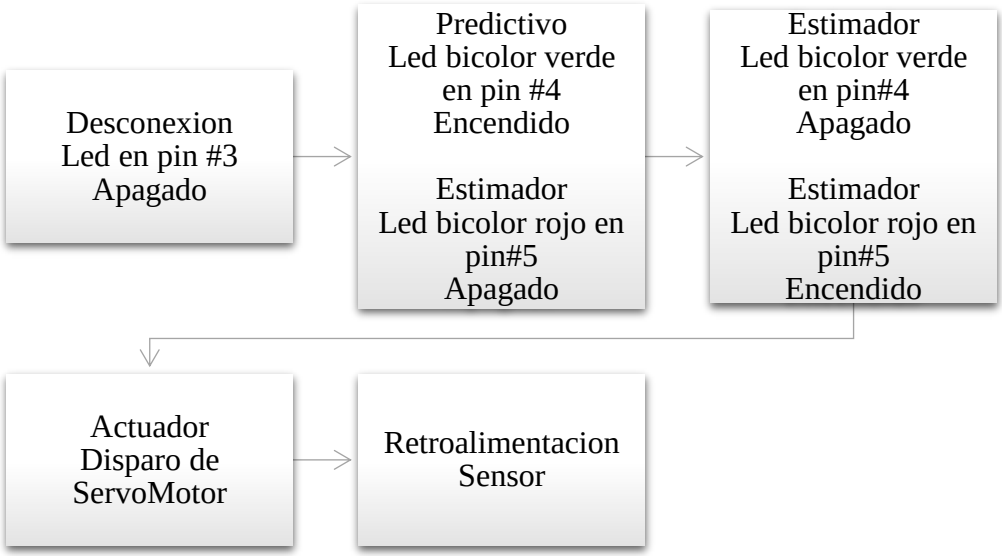


Comporatamiento de la interfaz principal en estado de desconexion, Elaborado por: Diego Soberon

Se debe tener en cuenta que el sistema puede tener pérdidas de información en el cual se necesita estabilizar la bola en el centro de la barra aproximadamente para cuando se realice la reconexión con el controlador no reanude desde un ángulo tan inclinado si no desde un punto medio, ahí es donde se aplica un algoritmo predictivo para equilibrar el proceso físico y es aplicado en la planta que es conformado por el Arduino Mega y la Shield Dragino Yun, al aplicar estos procesos el led indicador bicolor se enciende en el color verde o rojo para expresar el predictivo o el estimador respectivamente.

Para representar el comportamiento del algoritmo predictivo que se muestra en la Figura 4.4 se usa un led bicolor que representa si se necesita un ángulo positivo que llega a un límite de 110º que se lo denomina estimador o se necesita un ángulo negativo que tiene un límite de 50º que se denomina predictivo.

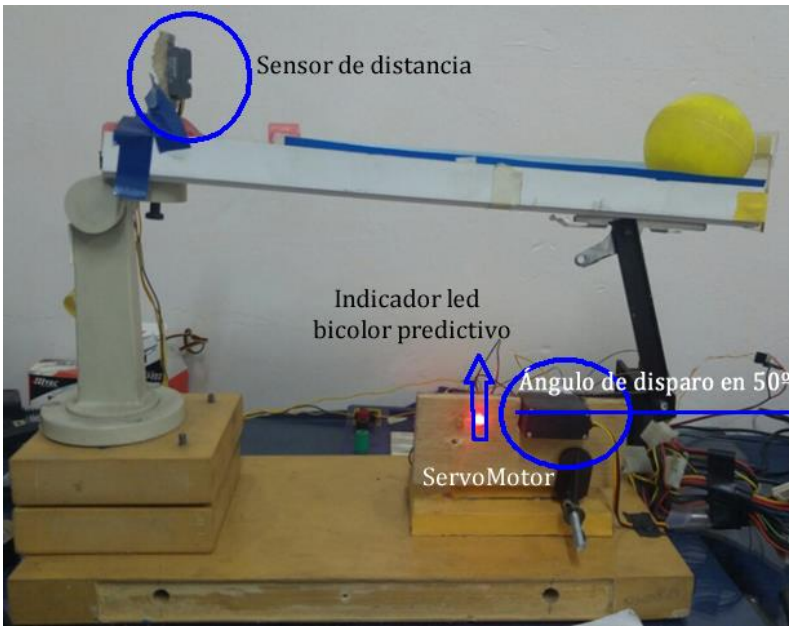
Figura 4.4: Comportamiento predictivo



Comportamiento de la planta en ejecucion del predictivo negativo, Elaborado por: Diego Soberon

En la Figura 4.5 se muestra el led bicolor encendido en color rojo que indica el ángulo negativo del predictivo, está estimado entre 50° a 70° de inclinación donde el ángulo límite mínimo es de 50° para el disparo del ServoMotor., El actuador no tendrá un ángulo de inclinación menor a 50º, para que no exista oscilaciones y perdidas de equilibrio.

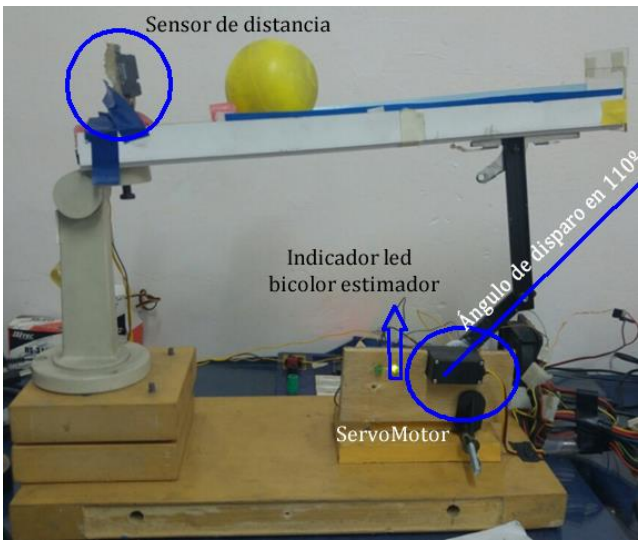
Figura 4.5: Funcionamiento del predictivo ángulo negativo



Comportamiento del sistema en ejecucion predictivo negativo, Elaborado por: Diego Soberon

En la Figura 4.6 se muestra el led bicolor encendido en color verde que indica el ángulo positivo del predictivo, está estimado entre 75° a 110° de inclinación donde el ángulo límite máximo es de 110° para el disparo del ServoMotor., El actuador no tendrá un ángulo de inclinación mayor a 110º, para que no exista oscilaciones y perdidas de equilibrio.

Figura 4.6: Funcionamiento del predictivo ángulo positivo



Comportamiento del sistema en ejecucion predictivo positivo, Elaborado por: Diego Soberon

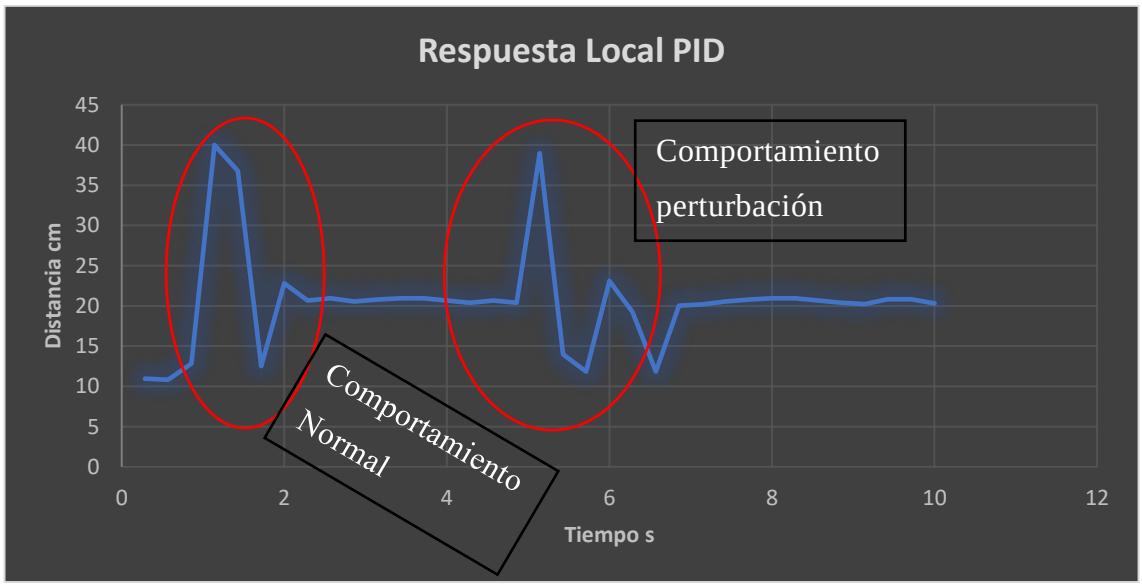
Una vez que se estabiliza el proceso en la planta de modo grafico se muestra la estabilización en la Figura 4.7 usando los datos de la Tabla 4.1 por lo tanto, si llega a tener una perturbación vuelven a ejecutar el controlador para estabilizar el proceso por lo que está en constante monitoreo y a la mínima perturbación busca estabilizarse no se desconecta a menos que se apaguen o se dejen de alimentar los equipos tales como la planta que está constituida por Arduino Mega y Dragino Yun o el controlador que viene a ser el Raspberry Pi3.

Tabla 4.1: Valores del PID local

Valores	
SetPoint	20
Kp	1.5
Ki	1.0
Kd	0.0

Valores usados en el controlador PID local, Elaborado por: Diego Soberon

Figura 4.7: Prueba local del controlador PID



Comportamiento del interfaz de respuesta con datos emitidos de la planta, Elaborado por: Diego Soberon

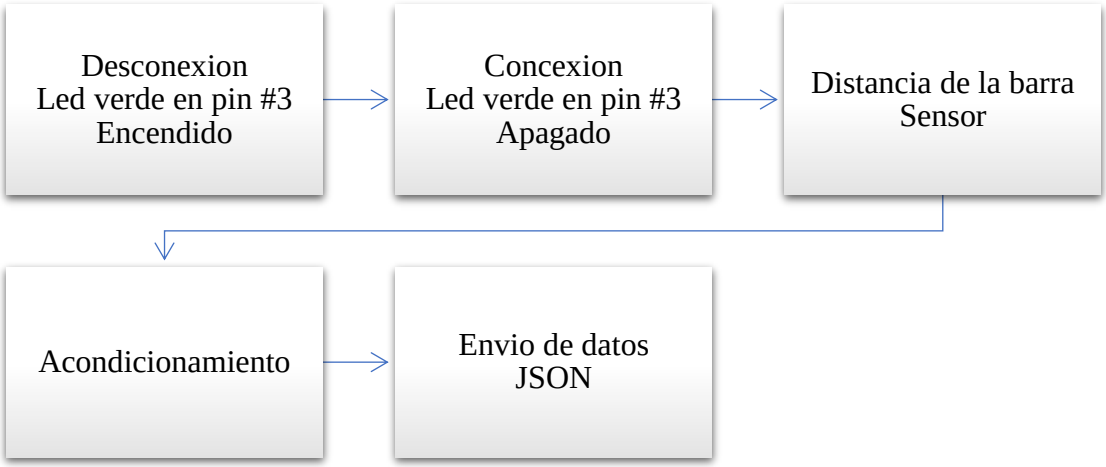
4.2 Ensayo de forma remota o en la nube

Al realizar las pruebas de forma remota en la nube con el proveedor de servicios "SmarterAPS" los valores iniciales tomados de la simulación y la prueba local que se realizó anteriormente, no tenemos una desconexión en la interfaz de inicio ya

que el servidor siempre tiene un dato el cual se dibuja en la respuesta al ejecutar el controlador PID la respuesta generada por la prueba local y la prueba remota de la nube son semejantes.

En la Figura 4.8 se describe el comportamiento inicial de la planta que va desde el indicador luminoso de desconexión, al envió de datos mediante el método de intercambio de información JSON.

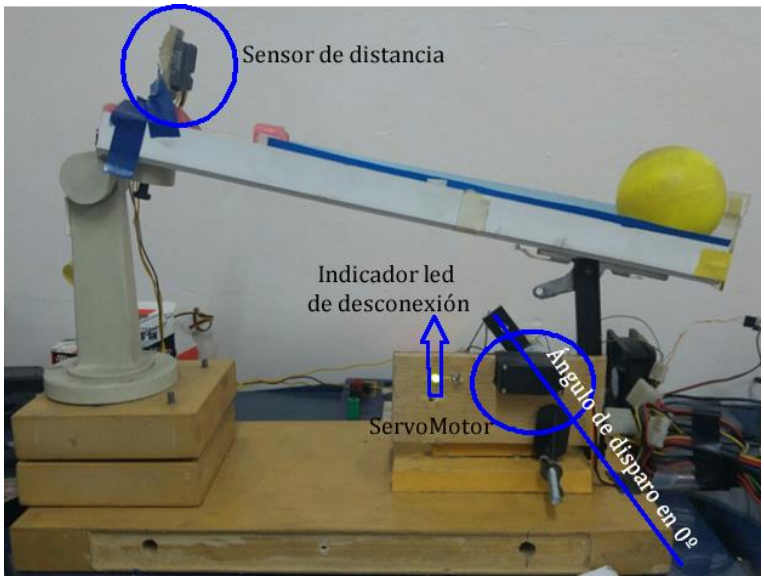
Figura 4.8: Comportamiento normal del sistema



Comportamiento de la planta en desconexion con el servidor remoto, Elaborado por: Diego Soberon

Cuando la planta se encuentra en modo de desconexión el led de color verde conectado al pin #3 del Arduino Mega y Shield Dragino Yun se activa como se muestra en la Figura 4.9, el actuador que es el servo motor tiene un ángulo de 0º y la interfaz principal del controlador subido en la nube siempre dibujara el último dato que tiene almacenado en su archivo de intercambio de información "cliente.json" así se ve en la Figura 4.10, ya no despliega el mensaje de no conexión como se mostraba en el ensayo local.

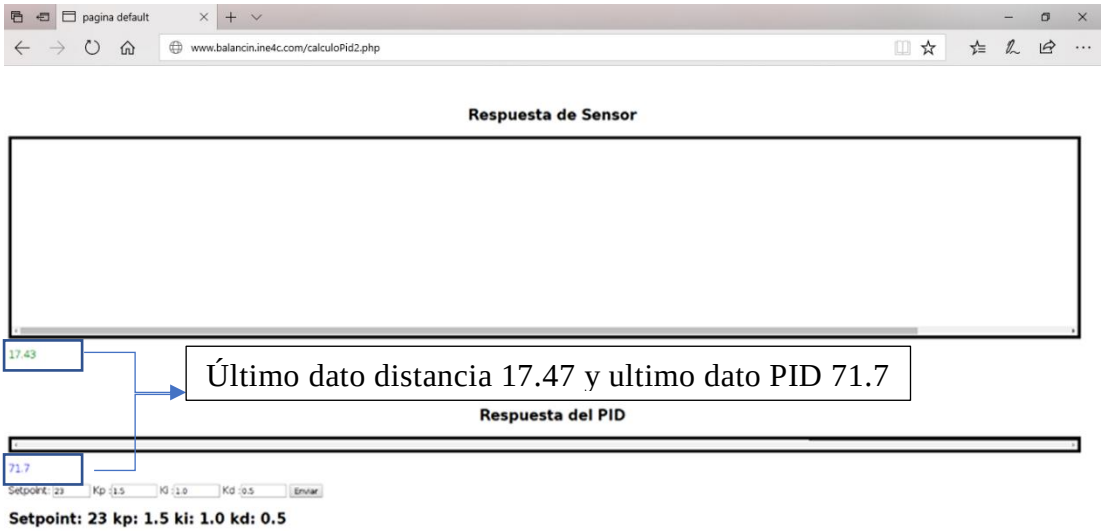
Figura 4.9: Estado de desconexión en la Nube



Comportamiento de la planta en desconexion con el servidor remoto, Elaborado por: Diego Soberon

En la interfaz principal se muestra en las secciones dos y tres que son la expresión numérica o textual de los datos tanto de distancia como de respuesta del controlador PID con el último dato obtenido antes de desconectar la planta y graficara ese dato hasta establecer la conexión entre el controlador PID y la planta.

Figura 4.10: Datos sostenidos en la nube

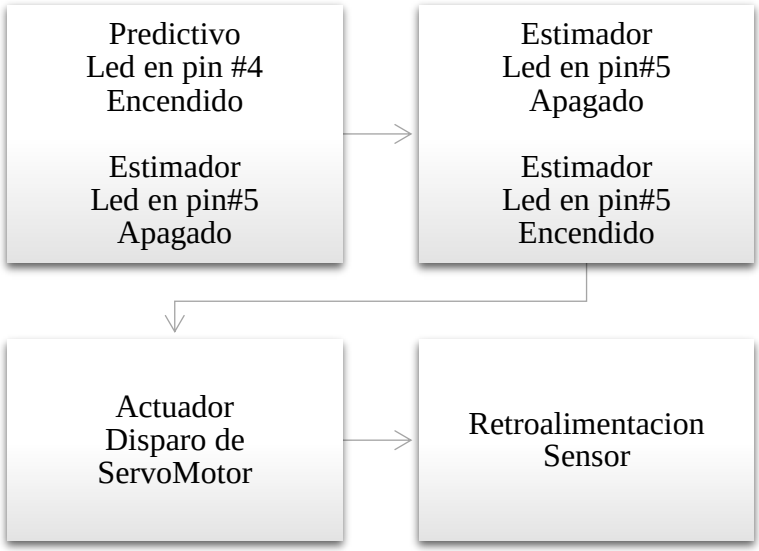


Comportamiento de la interfaz principal implementado en la nube, Elaborado por: Diego Soberon

El sistema en la nube tiene mayor pérdida de información y para estabilizar la bola en el centro de la barra aproximadamente hasta realizar la reconexión con el controlador remoto se aplica un algoritmo predictivo para equilibrar el proceso físico y es aplicado en la planta que es conformado por el Arduino Mega y la Shield Dragino Yun, al aplicar estos procesos, el led indicador bicolor se enciende en color verde o rojo para expresar el ángulo predictivo o el ángulo estimador respectivamente.

Para representar el comportamiento del algoritmo predictivo que se muestra en la Figura 4.11 se usa un led bicolor que representa si se necesita un ángulo positivo que llega a un límite de 110º que se lo denomina estimador o se necesita un ángulo negativo que tiene un límite de 50º que se denomina predictivo.

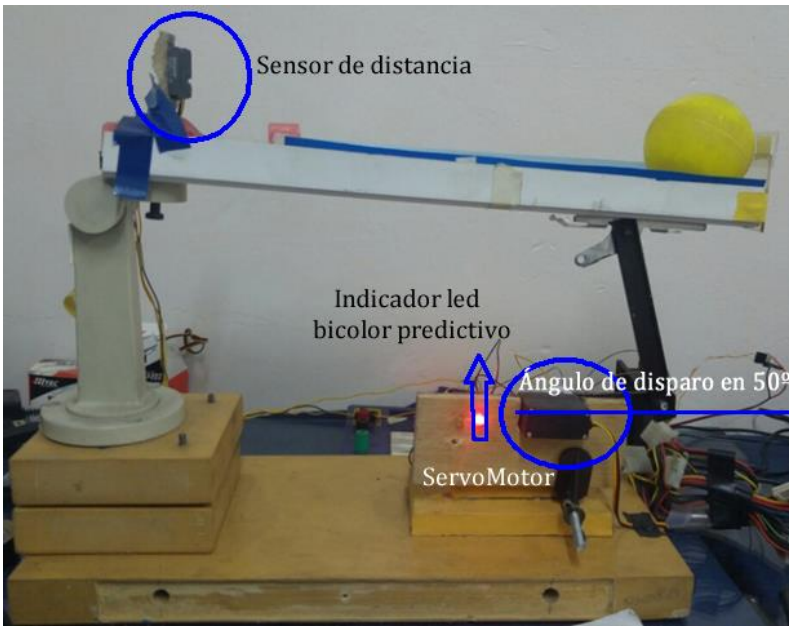
Figura 4.11: Comportamiento predictivo



Comportamiento del predictivo de forma remota disparo angulo negativo, Elaborado por: Diego Soberon

En la Figura 4.12 se muestra el led bicolor encendido en color rojo que indica el ángulo negativo del predictivo, está estimado entre 50° a 70° de inclinación donde el ángulo límite mínimo es de 50° para el disparo del ServoMotor, no tendrá un ángulo de inclinación menor a 50º

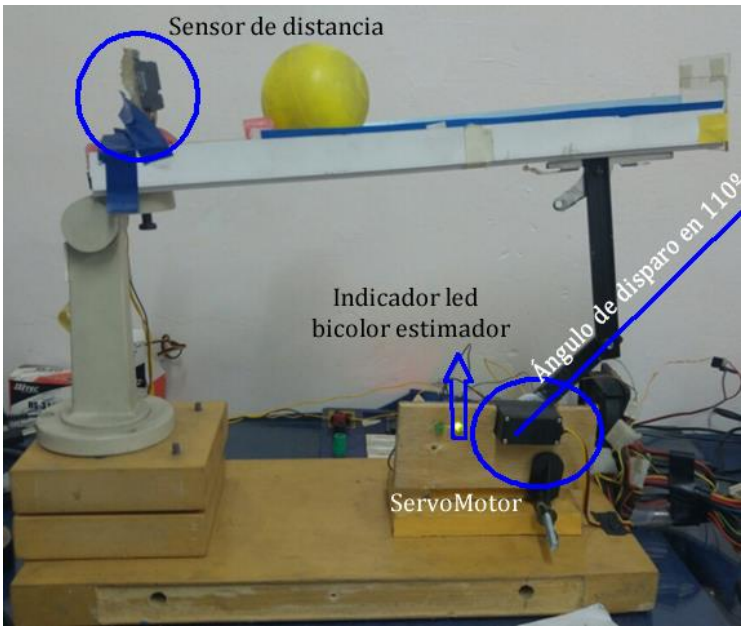
Figura 4.12: Funcionamiento del predictivo ángulo negativo



Planta ejecutando predictivo negativo en controlador remoto, Elaborado por: Diego Soberon

En la Figura 4.13 se muestra el led bicolor encendido en color verde que indica el ángulo positivo del predictivo, está estimado entre 75° a 110° de inclinación donde el ángulo límite máximo es de 110° para el disparo del ServoMotor, no tendrá un ángulo de inclinación mayor a 110º.

Figura 4.13: Funcionamiento del predictivo ángulo positivo



Planta ejecutando predictivo positivo en controlador remoto, Elaborado por: Diego Soberon

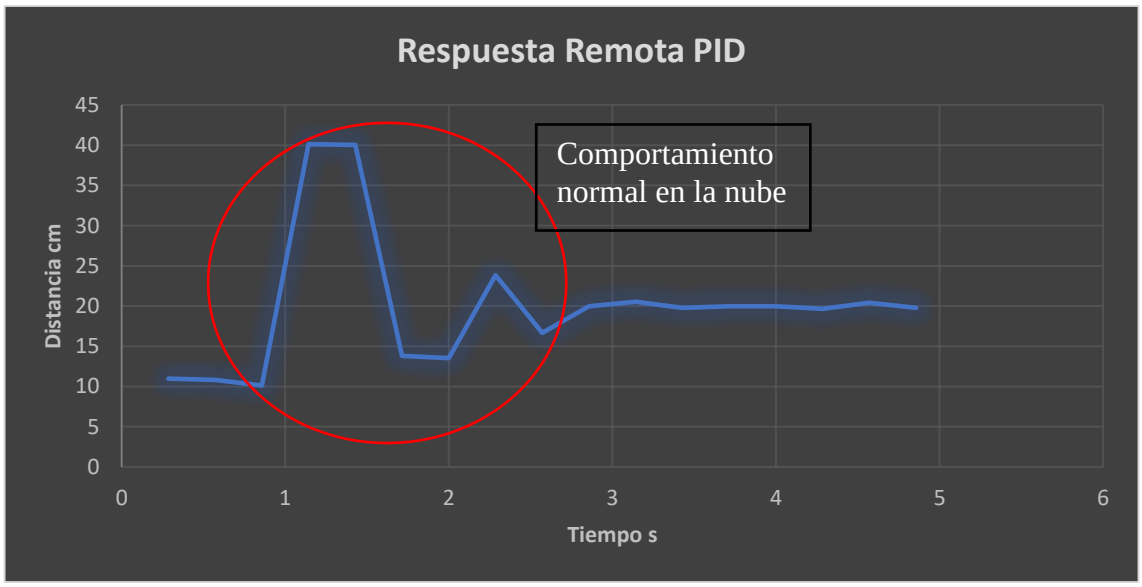
Una vez que se estabiliza el proceso en la planta de modo grafico se muestra la estabilización en la Figura 4.14 usando los datos de la tabla 4.2 por lo tanto, si llega a tener una perturbación vuelven a ejecutar el controlador para estabilizar el proceso por lo que está en constante monitoreo y a la mínima perturbación busca estabilizarse no se desconecta así se deje de alimentar eléctricamente la planta que está constituida por Arduino Mega y Dragino Yun, el controlador remoto por ser un proveedor de servicios nunca se desconecta.

Tabla 4.2: Valores de PID remoto

Valores	
SetPoint	20
Kp	1.5
Ki	1.0

Valores usados en el controlador PID remoto, Elaborado por: Diego Soberon

Figura 4.14: Respuesta del controlador en la nube

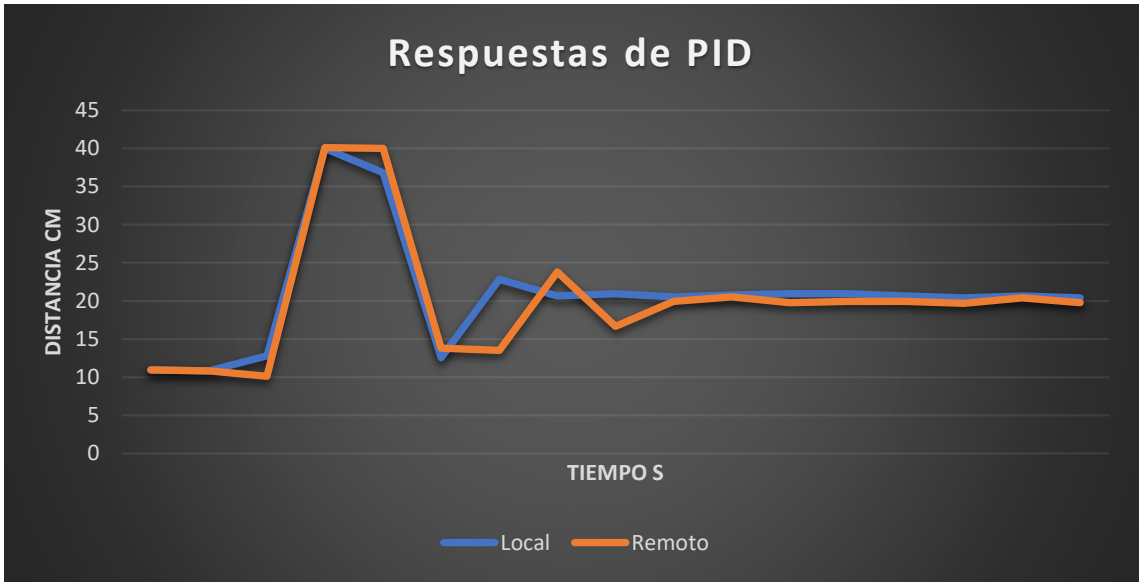


Interfaz de respuesta del controlador PID remoto, Elaborado por: Diego Soberon

Con respecto a la respuesta del actuador que es el ServoMotor se puede notar en la ejecución del controlador de forma local y de forma remota que la respuesta de disparo de ángulo es más lenta de forma remota que de forma local no está establecido un rango de tiempo, pero es perceptible a siempre vista.

Se realiza una comparación de graficas del controlador local y remoto en el cual se debe tener en cuenta que de forma local y de forma remota tiene un comportamiento muy similar que se observa en la Figura 4.15 estimando que el error es menor al 10% que es aceptable, esto debido a que el tiempo de respuesta del actuador es más lento estimado en 2milisegundos que ya influye directamente en el comportamiento del controlador PID.

Figura 4.15: Comparación de respuestas



Compracion de respuestas del controlador local y remoto respectivamente, Elaborado por: Diego Soberon

4.3 Tabla comparativa entre controlador local y controlador remoto en la nube

Al realizar los controladores local y remoto que tiene un espacio en la nube se tiene varias características que destacan a cada controlador y presentamos estas diferencias en la Tabla 4.3 describiendo en la misma las características singulares del sistema.

Tabla 4.3: Tabla comparativa entre Controlador

CONTROLADOR LOCAL	CONTROLADOR REMOTO EN LA NUBE
El controlador se ejecuta desde un Raspberry Pi3 que asume el rol de un servidor local	El controlador se ejecuta desde un proveedor de servicios

	"SmarterASP.NET" que actúa como un servidor con un espacio en la nube
Para establecer la conexión de (cliente – planta) y (servidor – raspberry Pi3) se realiza por medio de la dirección IP que se genera por el router de la red LAN local	Para establecer la conexión entre cliente y servidor se realiza por medio de la dirección DNS que proporciona el proveedor de servicios "SmarterASP.NET" y es estático ya que no cambia el nombre del espacio activado.
Los archivos creados en el servidor Raspberry Pi3 al ser creados se requiere dar permisos de lectura y escritura para que el controlador pueda recibir y enviar los datos a la planta	Los archivos que usa el controlador remoto en la nube son una copia del controlador local con la diferencia que no se necesita dar permisos de lectura y escritura para el funcionamiento del controlador PID, además que el archivo para intercambio de información "cliente.json" debe ser creado en el servidor remoto ya que al copiarlo del servidor local este archivo no genera la comunicación con la planta.
El porcentaje de uso del procesamiento del sistema que involucra la cantidad de información que se intercambia entre la planta y el controlador PID se puede observar en la parte superior izquierda de la pantalla de la Raspberry Pi3 en forma de gráfica. Al ejecutar el procesamiento y algoritmo del controlador PID, en el grafico se puede observar que tiene un alto porcentaje esto significa que al estar conectado con la planta el servidor requiere gran procesamiento,	En el proveedor de servicios "SmarterASP.NET" no se tiene un indicador grafico del porcentaje de información que se intercambia entre el controlador y la planta, al subir los archivos del controlar PID al espacio en la nube solo se especifica el ancho de banda que puede manejar el proveedor de servicios.

es decir utiliza casi todos los recursos del Raspberry Pi3 para su ejecución.	
Al usar el dispositivo Raspberry Pi3 los recursos del servidor son dedicados únicamente al proceso del controlador PID sin compartir los recursos con ningún otro usuario, por ende se tiene un servidor dedicado ocasionando menores tiempos de retardo en la transmisión de información	Al usar el controlador en la nube no se tiene un proceso dedicado, sino que se utiliza recursos que son compartidos y la información recorre una mayor trayectoria por diferentes nodos de comunicación lo cual hace que la respuesta de procesamiento del sistema sea más lenta.
La respuesta del actuador, es decir el ServoMotor es visiblemente más rápida al ser ejecutado el Algoritmo del controlador PID, la velocidad de reacción se debe a que usa recursos del servidor que son dedicados ejecutados en el Raspberry Pi3 para el intercambio de información con la planta.	La respuesta del actuador, el ServoMotor fue visible más lenta con respecto al controlador local, pues además que se tiene más retardo debido al intercambio de información por la red, se produce momentos de inercia propios de elementos físicos cuando no se tiene una información continua.

Tabla comparativa entre controlador PID local y remoto, Elaborado por: Diego Soberon

CONCLUSIONES

Para la implementación de controlador PID, en este proyecto se investigó y se utilizó lenguajes de programación web, con los cuales se ejecutó el algoritmo predictivo para el controlador en la nube, previamente se realizó un simulador del controlador PID, el cual está programado en lenguaje PHP enlazado JS, mediante librerías para presentación de reporte gráficos.

Se implementó un controlador en la nube para el sistema Ball/Bean, aplicando los conceptos de servicios en la nube, obteniendo un proceso eficiente utilizando recursos inalámbricos con dispositivos que soportan métodos de intercambio de información como lo es el JSON.

Al realizar los ensayos empíricos se obtuvo que el controlador PID remoto tiene una respuesta lenta que es aproximadamente dos milisegundos con respecto al controlador PID local debido a que los recursos que genera el servidor remoto son compartidos con cientos o miles de usuarios por lo tanto no tiene recursos dedicados para la mejor ejecución del controlador PID con mayor velocidad.

Este proyecto como tal representa una guía para el funcionamiento e implementación del controlador PID de forma local y remota para el sistema Ball/Bean, para lo cual se buscó migrar los dispositivos de un medio guiado como lo es el cableado a un medio no guiado como lo es una red WLAN y se virtualizó el controlador. La aplicación desarrollada uso de lenguajes de programación web como los es PHP, JavSscipt, esta vinvulada a una base de datos MySql en el Raspberry Pi3 que se configuro como servidor local y que actúa como controlador.

Actualmente, se implementan la comunicación inalámbrica solo con sensores y actuadores ya que tienen niveles de seguridad que no dejan interceptar ni manipular la información, en cambio al usar un controlador PID virtual es vulnerable a ataques esto quiere decir que la información puede ser modificada debido a que esta implementado en un sistema de acceso libre y sigue en estudios para mejorar los sistemas.

RECOMENDACIONES

Se recomienda tener niveles de seguridad para que la información no pueda ser interceptar porque tiene una gran vulnerabilidad a ataques maliciosos y manipular los datos que se recepta y genera debido a que es un software libre sin ningún tipo de seguridad informática.

Para mejorar el proyecto se puede tratar de conseguir software compatible con el sistema operativo de raspbian ya que los paquetes utilizados a lo largo del proyecto eran compatibles con sistemas como Windows y Mac y no se podían aplicar directamente desde el Raspberry Pi3 los paquetes como arduino aunque tiene su versión para Linux no es compatible con las librerias que se necesitaban agregar para la programación del Dragino Yun por dar un ejemplo.

Aunque un limitante importante en la Raspberry Pi3 es la micro sd que comparado con los equipos robustos que son dedicados para servidores se puede incrementar el tamaño de la tarjeta Sd que permite explotar de mejor manera los recursos de servidor en la Raspberry Pi3.

REFERENCIAS BIBLIOGRÁFICAS

Antonio Mancina, G. F. (2009). A Real-Time Service-Oriented Architecture for Industrial Automation. IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS.

Bonilla Agualongo, P. A. (2016). *Diseño e implementacion de un sistema de control en red (NCS), utilizando sensores y actuadores inteligentes para los mecanismos de control: PID, PWM y Control On/Off, monitoreados mediante una aplicacion en la nube.* Quito: UPS.

Caparrini, F. (11 de Diciembre de 2011). *slideshare*. (Slideshare.net) Recuperado el 17 de Febrero de 2017, de http://es.slideshare.net/FernandoCaparrini/sistemas-multiagente

Cobos, J. S. (2 de Noviembre de 2006). *Universitat de Barcelona.* Recuperado el 16 de Agosto de 2017, de http://bid.ub.edu/17serra2.htm

CodeCademi company. (2017). *codecademi.com.* Recuperado el 16 de Agosto de 2017, de https://www.codecademy.com/es/tracks/php-clone

Dragino. (15 de Diciembre de 2017). *Dragino Corporation.* Obtenido de Dragino Corporation Web site: http://wiki.dragino.com/index.php?title=Yun_Shield

ecma internacional. (2013). *The Json Data Interchange Format.* Standard ECMA-404. Recuperado el 16 de Agosto de 2017, de http://www.json.org/json-es.html

Fernandez, A. (26 de Junio de 2013). *Diseño e implementación de sistemas de control de tiempo real.* Obtenido de Proyecto final de carrera de Ingeniería en Automática y Electrónica Industrial: http://upcommons.upc.edu/bitstream/handle/2099.1/19455/Article.pdf?sequence=2

geekland. (5 de Febrero de 2016). *geekland.eu*. (geekland) Recuperado el 17 de Febrero de 2017, de https://geekland.eu/que-son-los-servicios-en-la-nube/

Highcharts. (2017). *Highcharts.* Recuperado el 16 de Agosto de 2017, de https://www.highcharts.com/demo/dynamic-master-detail/dark-unica

Hitecrcd. (15 de Diciembre de 2017). *Hitecrcd Corporation.* Obtenido de HitecrcdCorporation Web site: http://hitecrcd.com/products/servos/sport-servos/analog-sport-servos/hs-311-standard-economy-servo/product

K.Mukherjee, G. (2010). Development of Mathematical Model for Market-Oriented Cloud Computing. Internacional Journal of computer aplications.

Ken, Y. (2007). Cloud Control with Distributed Rate Limiting. California: ResearchGeat.

Marco Mendes, P. L. (s.f.). Service-oriented Agents for Collaborative Industrial Automation and Production Systems.

Maria Ganzha, M. M. (2014). Inserting “Brains” into Software Agents – Preliminary Considerations. Calabria, Italy: Alfredo Cuzzocrea Mukaddim Pathan.

Morilla, F. (2006). Controladores PID, ajuste empirico. Madrid: UNED.

Omid Givehchi, H. T. (s.f.). Cloud Computing for Industrial Automation Systems - A Comprehensive Overview. Lemgo, Alemania.

Pillajo Angos, C. G. (Marzo de 2017). *Universidad Pontificia Bolivariana.* Obtenido de Universidad Pontificia Bolivariana Web site: https://repository.upb.edu.co/bitstream/handle/20.500.11912/3111/T.G.%20Carlos%20Pillajo%20Final%20Mar%2010.pdf?sequence=1&isAllowed=y

Pololu. (15 de Diciembre de 2017). *Pololu Corporation.* Obtenido de Pololu Corporation Web site: https://www.pololu.com/file/0J85/gp2y0a21yk0f.pdf

Prats, P. M. (26 de Febrero de 2016). *VNIVERSITAT DO VALENCIA.* Recuperado el 16 de Agosto de 2017, de http://www.uv.es/uvweb/master-ingenieria-telecomunicacion/es/blog/lan-wan-man-otras-redes-1285954593702/GasetaRecerca.html?id=1285959494096

RASPBERRY PI FOUNDATION. (s.f.). *raspberry.org.* Recuperado el 16 de Agosto de 2017, de https://www.raspberrypi.org/learning/teachers-guide/

Shucheng Yu, C. W. (2010). Achieving Secure, Scalable, and Fine-grained Data. IEEE INFOCOM.

SpikenzieLabs. (15 de Diciembre de 2017). *Arduino Corporation.* Obtenido de Arduino Corporation Web site: https://www.arduino.cc/en/Main/arduinoBoardMega

Stamatis Karnouskos, A. W. (s.f.). A SOA-based architecture for empowering future collaborative cloud-based industrial automation.

Suempresa.cloud. (2016). Tipos de servicios en la nuve que hoy encuentras.

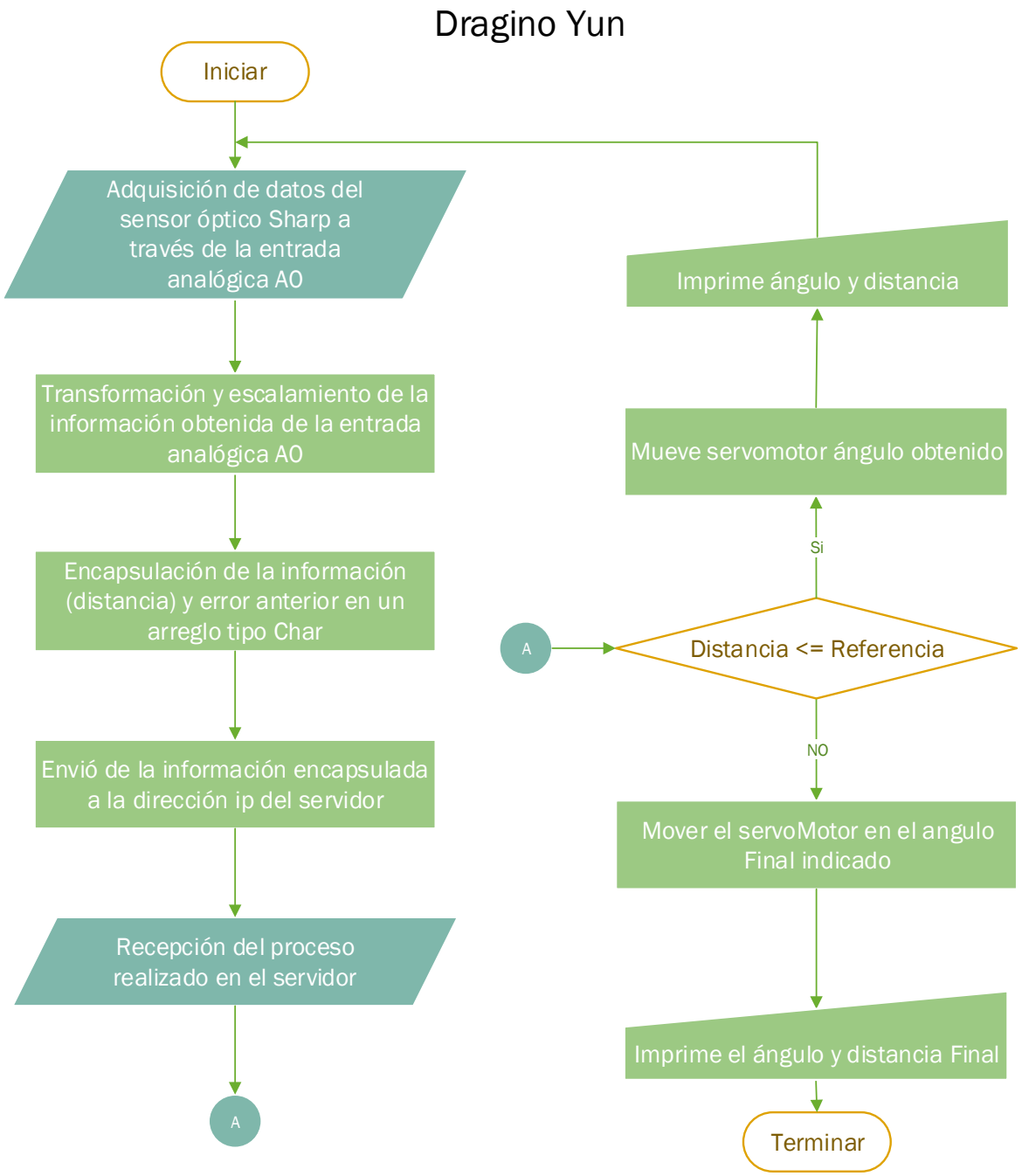
Tharindu Patikirikorala, A. C. (s.f.). Feedback controllers in the cloud. Victoria, Australia.

The PHP GROUP. (2001). *php.net*. Recuperado el 16 de Agosto de 2017, de http://php.net/manual/es/intro-whatis.php

Velazquez, E. (8 de Enero de 2009). *pymesyautonomos*. (pymesyautonomos.com) Recuperado el 17 de Febrero de 2017, de https://www.pymesyautonomos.com/tecnologia/que-es-la-virtualizacion

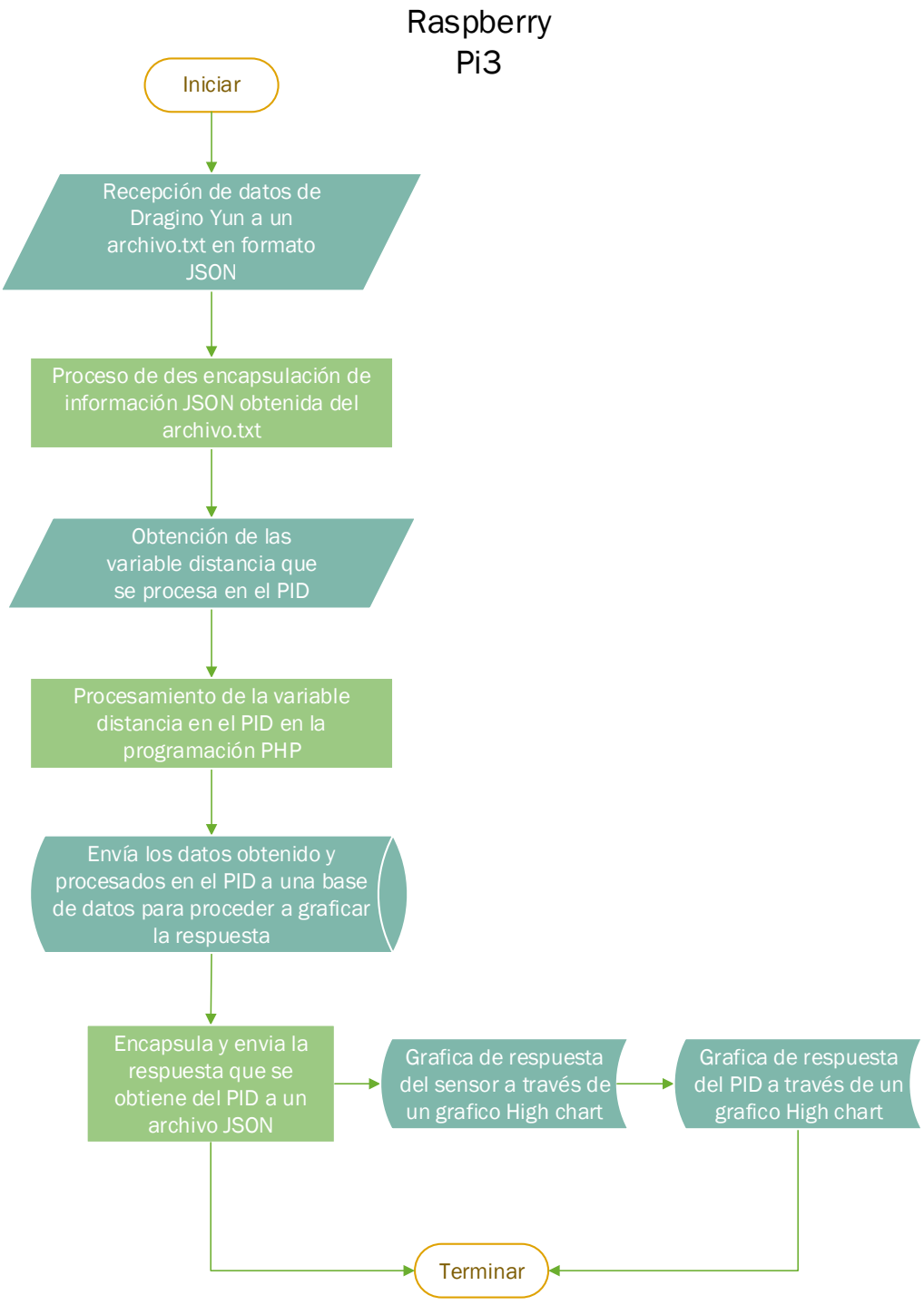
ANEXOS

Anexo 1: Funciones Dragino Yun



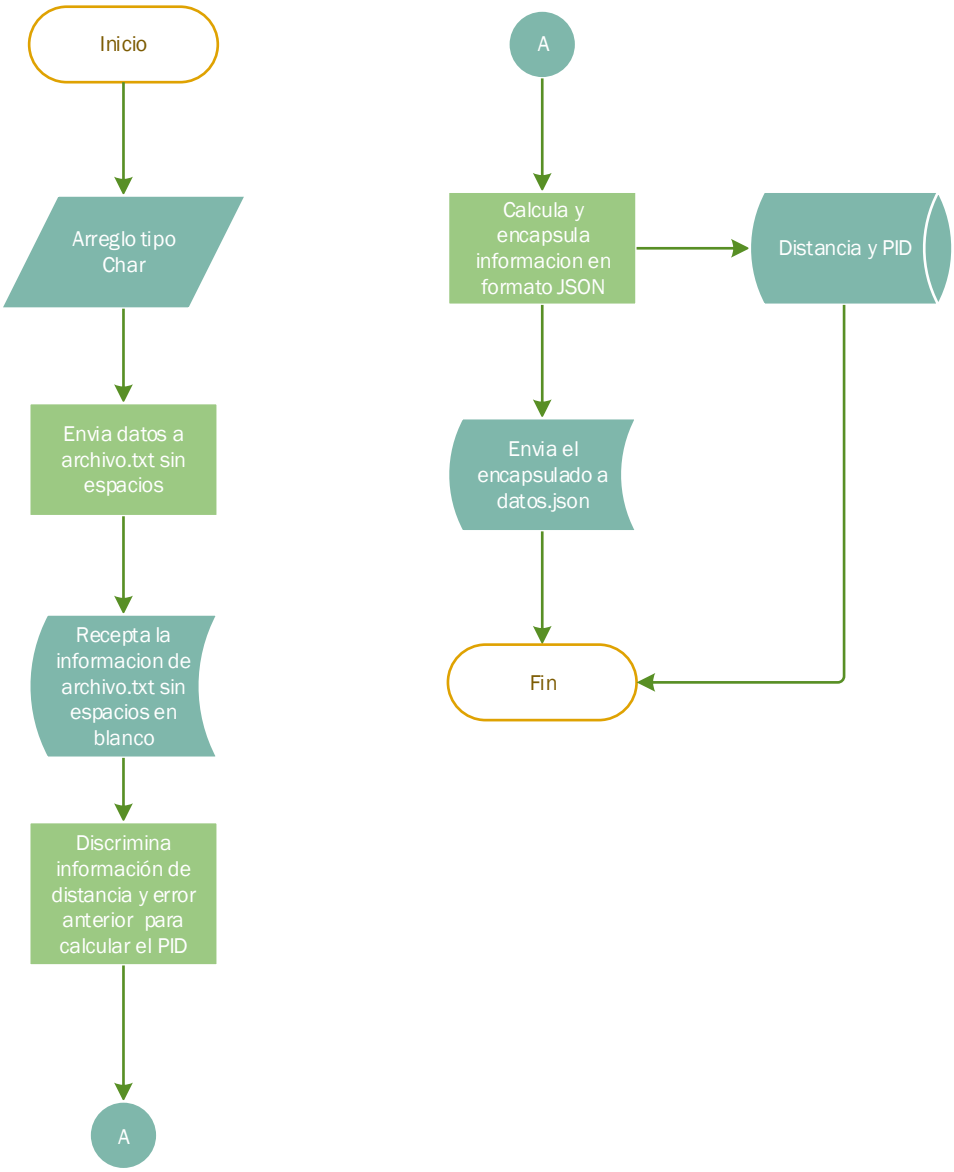
Elaborado por: Diego Soberon

Anexo 2: Funciones Raspberry Pi3



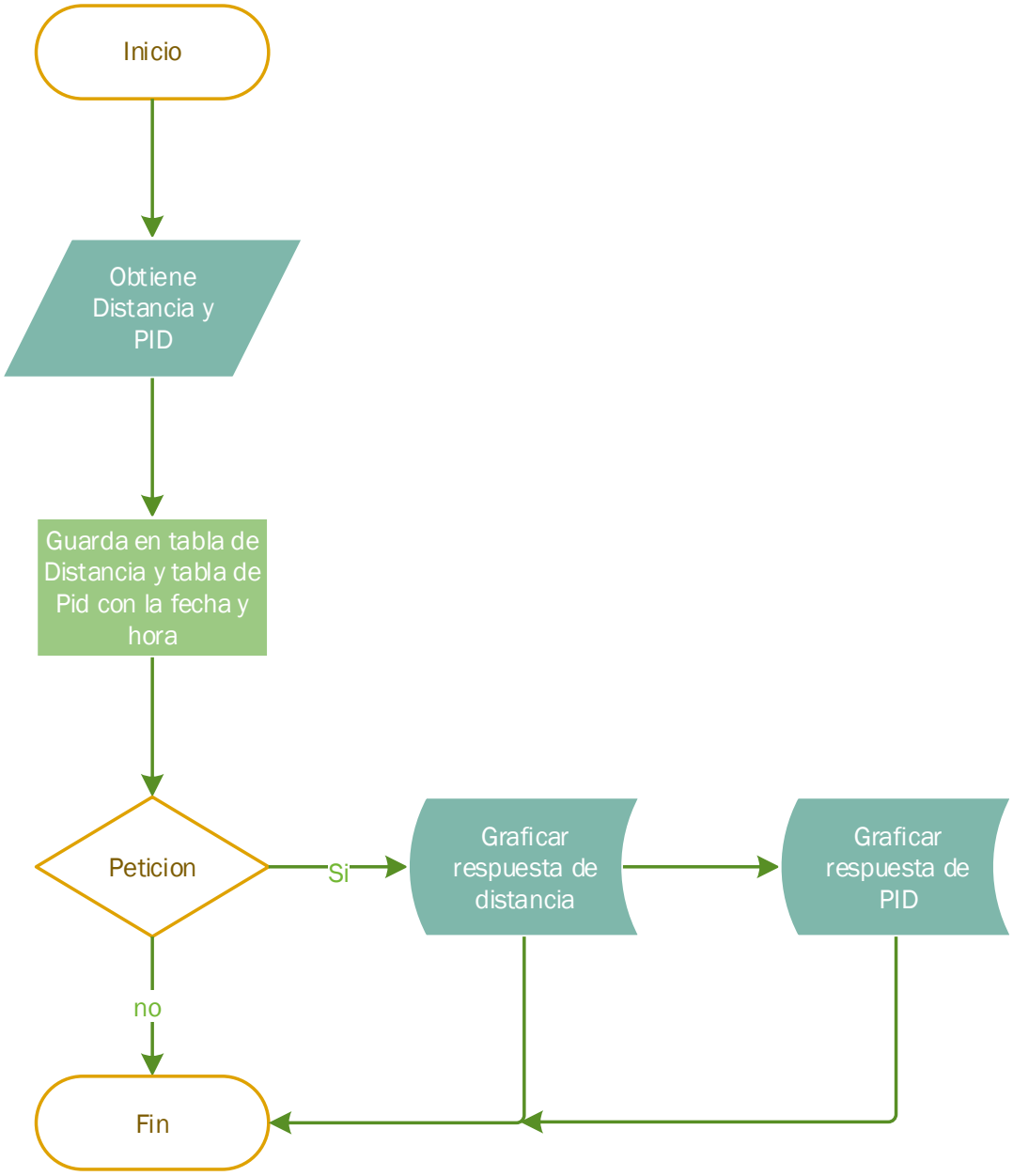
Elaborado por: Diego Soberon

Anexo 3 : Estructura del sistema PID con planta



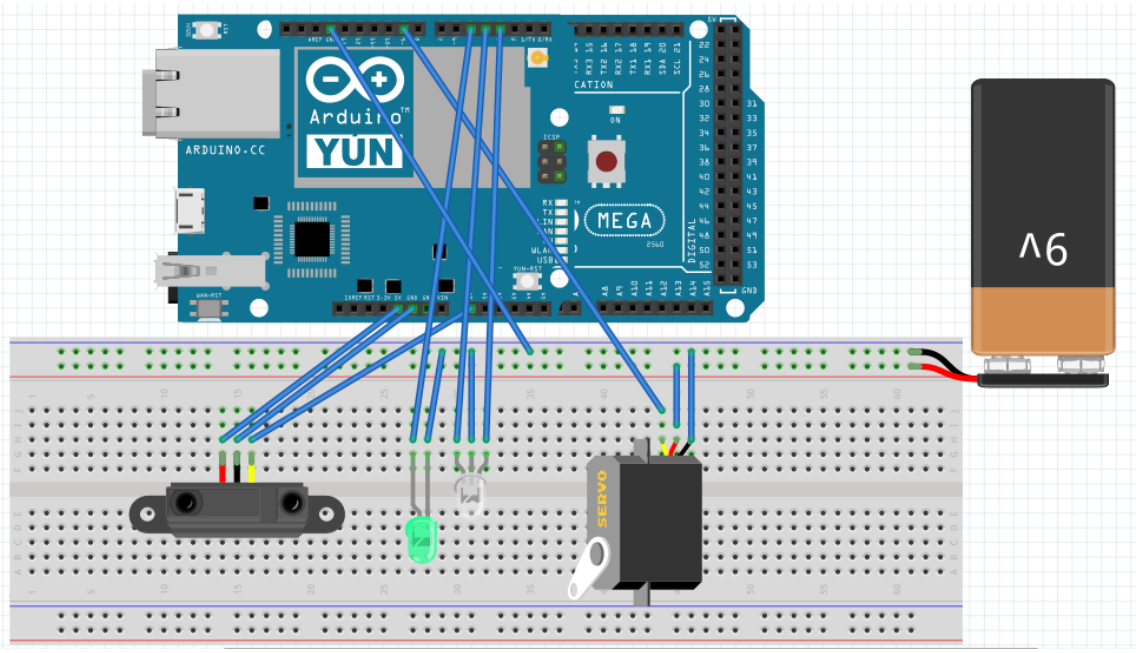
Elaborado por: Diego Soberon

Anexo 4 : Estructura de la base de datos



Elaborado por: Diego Soberon

Anexo 5 : Esquema de conexión en Fritzing



Elaborado por: Diego Soberon