

**Prototipo para la integración de información y generación de reportes de
producción para la empresa InfoDesign Colombia**



Daniela Ardila Penagos

**Trabajo de grado en Automática Industrial
Modalidad: Práctica profesional**

Director:

Mg. Oscar Amaury Rojas Alvarado

Asesor de la empresa:

Ing. Juan David Baquero

Universidad del Cauca

Facultad de ingeniería electrónica y telecomunicaciones

Programa de ingeniería en automática industrial

Popayán, Cauca

2021

**Prototipo para la integración de información y generación de reportes de
producción para la empresa InfoDesign Colombia**

Daniela Ardila Penagos

**Trabajo de grado presentado como requisito para optar al título de:
Ingeniera en Automática Industrial**

Director:

Mg. Oscar Amaury Rojas

Asesor de la empresa:

Ing. Juan David Baquero

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Electrónica, Instrumentación y control

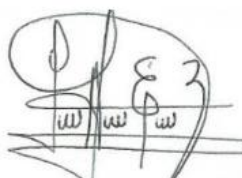
Programa de Ingeniería en Automática Industrial

Popayán, Colombia

2021

Nota de Aceptación:

Firma del jurado

A handwritten signature in black ink, consisting of several loops and a long horizontal stroke at the bottom.

Firma del jurado

A handwritten signature in black ink, featuring a large, bold initial 'A' followed by several vertical and horizontal strokes.

Popayán, Julio de 2021.

Dedicatoria

A mi madre quien desde el cielo ha sido mi guía, fortaleza y motivación en cada paso de mi vida y quien está presente en todos mis sueños y metas.

A mi padre quien con su esfuerzo, paciencia, amor y dedicación me ha permitido llegar a cumplir un sueño más, gracias por inculcar en mi el ejemplo de esfuerzo y valentía.

Finalmente a todas las personas que estuvieron presentes y de una u otra forma me acompañaron en este proceso.

Agradecimientos

Agradezco a Dios quien con su bendición llena siempre mi vida y la de mi familia.

Al Ingeniero Oscar, mi director de trabajo de grado y principal colaborador quien con su dirección, conocimiento y apoyo permitió el desarrollo de este proyecto.

Resumen

Este proyecto se realiza con la empresa InfoDesign Colombia y tiene como finalidad el desarrollo de un sistema de información capaz de integrar diferentes protocolos usados en la Industria tales como OPC, Modbus y DNP3, integrarlos en una base de datos SQL, almacenarlos en una interfaz web y transformarlos para generar valor, facilitar su análisis y servir como apoyo en la toma de decisiones.

Palabras clave: base de datos, protocolos industriales, integración de datos, desarrollo web.

Abstract

This project was performed in the company InfoDesign Colombia and its purpose is to develop an information system capable of integrate different industrial protocols such OPC, Modbus and DNP3, integrating them in a SQL database, storing them in a web interface and transforming them to generate value, facilitate their analysis and decision-making.

Keywords: database, industrial protocols, data integration, web development.

Tabla de Contenido

1.	<i>Introducción</i>	11
2.	<i>Capítulo I: Contextualización del problema</i>	12
2.1.	Planteamiento del problema	12
2.1.1.	Descripción del problema	12
2.1.2.	Formulación del problema	13
2.1.3.	Justificación del problema.....	13
2.2.	Requerimientos del cliente	15
2.3.	Objetivos del estudio	15
2.3.1.	Objetivo general	15
2.3.2.	Objetivos específicos.....	16
2.4.	Arquitectura	16
3.	<i>Capítulo II: Contextualización teórica</i>	18
3.1.	Marco conceptual teórico	18
3.1.1.	Protocolos industriales	18
3.1.2.	Modbus	18
3.1.3.	Modbus TCP/IP	18
3.1.4.	DNP3.....	19
3.1.5.	OPC UA (Unified Architecture).....	19
3.1.6.	Integración de datos	19
3.1.7.	Desarrollo web.....	19

3.1.8.	Frontend.....	20
3.1.9.	Backend.....	20
3.1.10.	KPIs.....	20
3.1.11.	Web Responsive.....	20
3.1.12.	API.....	20
3.1.13.	Rest API.....	20
3.1.14.	Base de datos relacional.....	20
3.1.15.	OT.....	21
3.1.16.	IT.....	21
3.1.17.	EEMUA 191.....	21
4.	<i>Capítulo III: Contextualización metodológica.....</i>	22
4.1.	Contribución del trabajo:.....	22
4.2.	Resultados esperados:.....	22
4.3.	Metodología:.....	23
5.	<i>Capítulo IV: Desarrollo del trabajo.....</i>	24
5.1.1.	ETAPA 1: Integración de Protocolos Industriales en una única base de datos	24
5.1.2.	ETAPA 2. Generación de KPIs y dashboards.....	38
5.1.3.	ETAPA 3. Crear aplicativo web responsive en Angular.....	43
6.	<i>Validación y aceptación del prototipo.....</i>	47
7.	<i>Lecciones aprendidas.....</i>	48

8.	<i>Conclusiones</i>	50
9.	<i>Bibliografía</i>	53
10.	<i>Anexos</i>	55

Lista de Figuras

FIGURA 1 ARQUITECTURA	17
FIGURA 2 CRONOGRAMA.....	23
FIGURA 3. SIMULACIÓN DE SEÑALES MODBUS ANÁLOGAS Y DISCRETAS.....	25
FIGURA 4. SIMULACIÓN DE SEÑALES OPC ANÁLOGAS Y DISCRETAS.....	26
FIGURA 5. SIMULACIÓN DE SEÑALES ANÁLOGAS Y DISCRETAS	27
FIGURA 6. LECTURA DE LAS SEÑALES OBTENIDAS DEL SIMULADOR	30
FIGURA 7. CANAL DE COMUNICACIÓN CON OPC DA.....	32
FIGURA 8. TAGS SELECCIONADAS.....	32
FIGURA 9. CONFIGURACIÓN OPC UA	33
FIGURA 10. ENDPOINT OPC UA.....	34
FIGURA 11. CONFIGURACIÓN DE PHPMYADMIN CON XAMPP.....	35
FIGURA 12. BASE DE DATOS Y TABLAS CREADAS PARA ALMACENAR LOS DATOS	36
FIGURA 13. TABLA CON LOS DATOS OPC.....	37
FIGURA 14. IMÁGENES CREADAS EN FIGMA	38
FIGURA 15. DATOS PUBLICADOS EN EL SERVIDOR.....	40
FIGURA 16. KPI SEÑALES FORZADAS DEL ÚLTIMO MES	42
FIGURA 17. KPI HISTÓRICO DE ALARMAS DEL ÚLTIMO MES.....	43
FIGURA 18. ANGULAR CLI	44
FIGURA 19. APLICATIVO WEB FINAL	45

1. Introducción

La movilidad de la información y la integración de los datos de proceso, se han convertido en una necesidad identificada en todos los sectores productivos del país. A pesar que hay diferentes soluciones en el mercado, los elevados precios han hecho que exista un rezago en los procesos de integración digital de las pequeñas y medianas empresas.

Es por ello que, en conjunto con la empresa InfoDesign, se entendió que el desarrollo de soluciones de bajo costo, que permitan realizar integración de datos de proceso y movilidad de la información en interfaces web, podría ser el cimiento de la transformación digital de muchas compañías que en la actualidad no cuentan con este tipo de soluciones. Esto tendría un efecto inmediato en la productividad de las mismas, gracias al uso adecuado de la información de proceso en la toma de decisiones en todos los niveles de la organización.

Se usó la metodología Scrum para el desarrollo del proyecto: todo el desarrollo se dividió en veinticuatro (24) entregas o sprints en un periodo de tiempo de seis (6) meses, lo que permitió tener un monitoreo semanal del avance del proyecto y las posibles barreras encontradas durante la implementación.

Los objetivos planteados dentro del proyecto incluyen la integración de protocolos industriales tales como OPC, Modbus y DNP3, al igual que el desarrollo de una interfaz web con la capacidad de visualizar los datos de proceso recopilados en cada uno de los protocolos mencionados.

2. Capítulo I: Contextualización del problema

2.1. Planteamiento del problema

El planteamiento del problema “nos conduce a saber qué es lo que deseamos investigar, a identificar los elementos que están relacionados con el proceso y a definir el enfoque” [1] .

2.1.1. Descripción del problema

En la industria actual, uno de los desafíos más importantes consiste en conseguir un eficiente manejo, análisis y transformación de la información que se genera alrededor de los sistemas de control. El requerimiento de información para la toma de decisiones se ha vuelto una constante en todas las capas organizacionales de las compañías del sector industrial en Colombia, se ha generado una alta demanda en soluciones y herramientas que permitan resolver esta necesidad.

Las tecnologías de la operación y de la información se han integrado para crear innovadores sistemas de información que permiten de manera versátil la adquisición y transformación de los datos, sin embargo, los costos de estas soluciones son elevados y la demanda en el mercado está restringida a unos pocos clientes que cuentan con grandes presupuestos de inversión.

Para la empresa InfoDesign Colombia, es de vital importancia desarrollar herramientas que permitan dar respuesta a los desafíos y necesidades presentados anteriormente y consecuentemente ofrecer soluciones comerciales en el ámbito industrial con productos con altos estándares de calidad y a precios accesibles a cualquier cliente.

Es por ello que el presente proyecto busca dar solución a dicha necesidad, a través del desarrollo de un sistema de información capaz de integrar diferentes fuentes de datos, almacenarlos y transformarlos en información útil.

2.1.2. Formulación del problema

El uso de la tecnología en la industria, ha sido un catalizador desde la revolución industrial en la evolución y progreso de las diferentes industrias. En la actualidad, el análisis de la información tiene un impacto directo en la toma de decisiones en todos los niveles de las organizaciones, y un efecto mucho mas directo en el sector industrial, es por ello que uno de los temas de interés de muchas compañías de software, está relacionado con los procesos de transformación digital y la movilidad de la información.

Para el caso particular de este proyecto InfoDesign recibió por parte de Schneider Electric, la solicitud de crear un sistema de información modular, capaz de leer diferentes protocolos industriales (Modbus TCP, OPC y DNP3) a través de Servidores Edge e integrarlos en implementaciones web basados en Angular.

2.1.3. Justificación del problema

Se ha identificado una creciente demanda en la inversión en transformación digital en el sector industrial del país. La transformación digital es una realidad y ya no es una opción.

La clave para la organización es ver la transformación digital como una oportunidad que permite combinar prácticas y formas de hacer que dan como resultado nuevas técnicas y habilidades.

Algunas ventajas de la transformación digital en el sector industrial son:

- Generar experiencias nuevas al cliente
- Movilidad de la información
- Mejorar la eficiencia operativa
- Generar nuevas fuentes de acceso a la información
- Desarrollar ventajas competitivas para la organización.
- Profundizar el análisis de datos (Big Data)
- Implementación de tecnologías de predicción de modelos dinámicos

Lo que se ha visto es que en la mayoría de las pequeñas y medianas organizaciones en Colombia y en el resto de Latinoamérica han entendido que no se trata de adoptar tecnología, sino de transformarse digitalmente a partir de ellas. Muchas compañías le apuestan a la diferenciación a través de la inversión en tecnologías que permiten la digitalización y optimización de sus procesos.

En la actualidad, la transformación digital no es una opción. Las empresas de hoy ya no pueden resistirse a este nuevo panorama, pues no hay otra manera de renovarse y competir que mediante la transformación digital.

Hoy las pymes tienen las mismas necesidades que cualquier gran organización, sin embargo limitaciones presupuestales impiden que puedan acceder a este tipo de soluciones, lo que las pone en una clara desventaja y es en este nicho, donde se ve una oportunidad de brindar soluciones de fácil acceso para la pequeña y mediana empresa.

2.2. Requerimientos del cliente

Infodesign es un integrador del area digital de Schneider Electric, y fue incluido en el desarrollo de una solución de Software llamada Remote MT, la cual permite administrar los activos del sistema de control sin importar la marca y generar KPIs que impacten areas operativas, técnicas y de producción, los cuales fueron segregados en diferentes tabs dentro del módulo.

Los requerimientos específicos de la empresa InfoDesign para este proyecto fueron los siguientes:

- Simular señales analógicas y discretas en los siguientes protocolos: Modbus TCP, OPC y DNP3
- Desarrollar un módulo de adquisición de datos escrito en JavaScript
- Desarrollar un módulo para almacenar los datos obtenidos en una base de datos SQL
- Procesar y transformar los datos obtenidos mediante JavaScript
- Desarrollar un módulo utilizando las librerías de angular para generar KPIs
- Desarrollar una aplicación web responsive para visualizar los resultados mediante angular

2.3. Objetivos del estudio

2.3.1. Objetivo general

Implementar un prototipo para la integración de información y generación de reportes de producción para la empresa InfoDesign Colombia

2.3.2. *Objetivos específicos*

- Integrar los datos generados en diferentes protocolos industriales (Modbus, OPC UA y DNP3) en una única base de datos relacional.
- Transformar los datos en información a través de la generación de Indicadores de Desempeño y tableros de control (dashboards).
- Desarrollar un sistema de Información Web responsive en Java y Angular para visualizar la información colectada y tabulada.

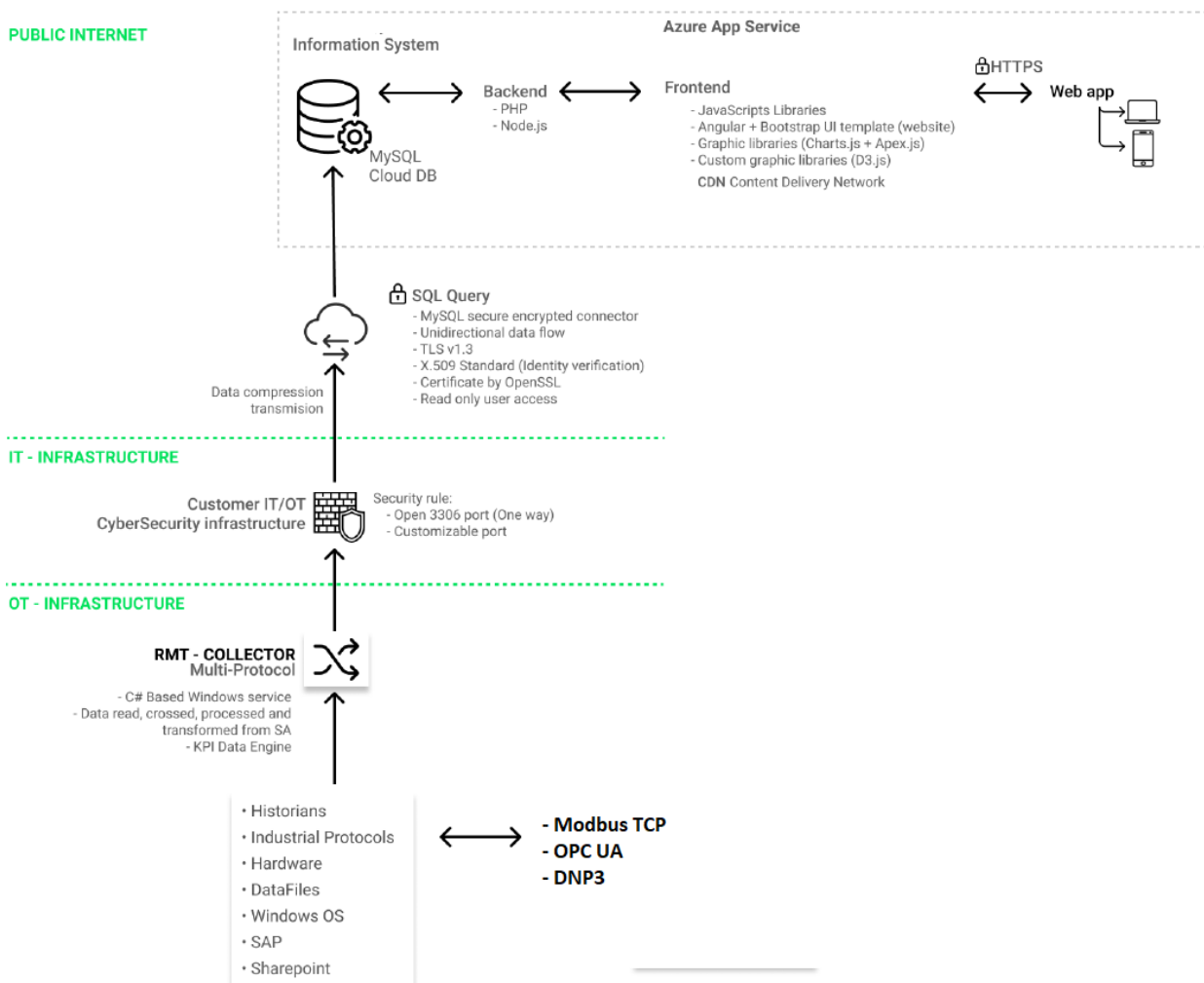
2.4. Arquitectura

La arquitectura general utilizada se muestra en la Figura 1, donde se detallan, cada una de las capas que hacen parte del proyecto.

- **Adquisición de Datos:** La capa de adquisición de datos que hace parte de la infraestructura OT, la cual se realiza a través de librerías de Node;
- **Unificación SQL:** La unificación de los datos en bases de datos SQL, se hace en el colector, creado en C# y localizado en la DMZ (Zona Desmilitarizada), la cual tiene conexión directa a la nube, usando la infraestructura IT.
- **Frontend:** El desarrollo de la capa de visualización, donde se desarrolla y visualiza el frontend (KPIs y Reportes), se almacena en la nube de Microsoft Azure, usando como fuente de datos las bases de datos SQL, y usando

Angular a través de librerías de JavaScript para el desarrollo del Sistema de Información con funcionalidad web para escritorio y móvil(Web Responsive).

Figura 1 Arquitectura



Fuente: InfoDesign

3. Capítulo II: Contextualización teórica

En este capítulo se abordan algunos conceptos necesarios con el fin de ayudar a comprender de manera adecuada el presente trabajo.

3.1. Marco conceptual teórico

Los principales conceptos y planteamientos teóricos utilizados son:

3.1.1. Protocolos industriales

“Los protocolos de comunicación son la plataforma sobre la cual dos o más dispositivos se comunican en una red, sea esta de cualquier complejidad. Los instrumentos de medición, protección y control en una planta industrial se rigen bajo el mismo concepto: una serie de reglas y normas sobre las cuales se comunicarán, responderán y actuarán.” [2] es decir, es un conjunto determinado de normas a cumplir por dos o más dispositivos que desean comunicarse entre sí.

3.1.2. Modbus

Modbus es un protocolo de comunicación industrial, utilizado para transmitir información a través de redes en serie entre dispositivos electrónicos.

Tiene una comunicación maestro-esclavo, el dispositivo que solicita la información se llama maestro Modbus y los dispositivos que suministran la información son los esclavos.

Existen diferentes versiones en el protocolo Modbus: Modbus RTU, Modbus ASCII, Modbus TCP.

3.1.3. Modbus TCP/IP

Es una variante del protocolo Modbus, y esta definida como probabilística, la cual permite la transmisión de datos sobre la capa TCP/IP. Se utiliza para monitorear,

gestionar e intercambiar información entre dispositivos, donde uno de ellos funciona como maestro y el otro como esclavo.

3.1.4. DNP3

DNP3 o protocolo eléctrico, usado típicamente en sistemas Scada, permite una transmisión cifrada con estampa de tiempo, garantizando una alta integridad de la calidad de los datos transmitidos.

3.1.5. OPC UA (Unified Architecture)

Es una tecnología de comunicación industrial multiplataforma, abierta, orientada a servicios, con cifrado TLS, la cual funciona en cualquier sistema operativo. Los datos vienen encriptados a través de certificados de seguridad.

3.1.6. Integración de datos

“Integrar significa combinar datos que se encuentran en diferentes fuentes para permitirle al usuario final tener una vista unificada de los mismos para una accesibilidad idónea, que sirva a las necesidades de negocio.” [3]

En otras palabras, la integración de datos se define como la capacidad de integrar fuentes de información convencionales y no convencionales en plataformas únicas que permiten comparar, analizar y simplificar la interpretación de estos.

3.1.7. Desarrollo web

El desarrollo web consiste en la implementación de interfaces HTML desarrolladas a través diferentes lenguajes de programación, tales como Angular, JavaScript, entre otros. La interfaz se almacena en una URL privada o pública.

3.1.8. Frontend

Es la parte de la aplicación web que interactúa con los usuarios, esta en el lado del cliente y se encarga del diseño y la interactividad, se logra utilizando HTML, CSS Y JavaScript.

3.1.9. Backend

Es el conjunto de aplicaciones que residen dentro del servidor, y sirven como motor de la interfaz web para garantizar un adecuado funcionamiento.

3.1.10. KPIs

Es una medida del nivel de rendimiento de una variable y/o conjunto de variables

3.1.11. Web Responsive

El diseño web responsive es una técnica de desarrollo web que logra adaptar un sitio web al entorno del dispositivo en el que se esté visualizando, mediante el uso de estructuras flexibles.

3.1.12. API

Conjunto de reglas y formatos que permiten a un programa comunicarse con otro módulo y/o equipo.

3.1.13. Rest API

Conjunto de buenas prácticas utilizadas en las requisiciones HTTP realizadas por una API en una aplicación web.

3.1.14. Base de datos relacional

Tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Se basan en el modelo relacional, es una forma de representar

datos en tablas con columnas y filas, cada columna de una tabla contiene atributos y cada fila es un registro con ID único.

3.1.15. OT

Tecnologías de la operación, es el uso de hardware y software para monitorear y controlar procesos físicos, equipos, infraestructura relacionados con la industria.

3.1.16. IT

Tecnologías de la información, es el conjunto de equipos y aplicaciones que permiten la transmisión, administración y almacenamiento de los datos. Es muy usado en el ámbito de los negocios y relacionado con los conceptos de infraestructuras de red y computo.

3.1.17. EEMUA 191

Es un conjunto de directrices para la gestión de alarmas, sus recomendaciones no son obligatorias, describe buenas prácticas y es utilizado por muchos organismos reguladores.

4. Capítulo III: Contextualización metodológica

En este capítulo se abordan aspectos relacionados con la contribución del trabajo, los resultados esperados y la metodología empleada en el desarrollo del mismo.

4.1. Contribución del trabajo:

Para la empresa InfoDesign, es de vital importancia desarrollar herramientas que permitan la integración de datos, movilidad de la información y reportes que faciliten la toma de decisiones a bajo costo.

El proyecto a desarrollar pretende impactar aquellos sectores industriales que no cuentan con un gran capital de inversión para adquirir soluciones que fomenten la transformación digital del país. El desarrollo planteado en esta pasantía, tiene como objetivo integrar tecnologías relacionadas con analítica de datos, almacenamiento en la nube y la integración con tecnologías de la operación (TO), dejando un camino base para que muchas compañías y futuras investigaciones puedan incursionar de manera fácil, económica y sencilla en los procesos de transformación digital del sector industrial del país.

4.2. Resultados esperados:

Dentro del ámbito industrial, es frecuente integrar diferentes tipos de protocolos, dada la diversidad de tecnologías que existen al interior de las plantas de producción; es por eso que dentro de los resultados que se esperan obtener está la integración de diferentes protocolos tales como Modbus TCP, DNP3 y OPC UA.

De igual manera se espera elaborar una infraestructura WEB, que permita la visualización y monitoreo de los datos.

4.3. Metodología:

Para el desarrollo y cumplimiento de los objetivos de este proyecto se utilizó la metodología Scrum que permite hacer entregas parciales y regulares del producto final, segmentar tareas y obtener victorias tempranas.

Se hizo una segmentación de tareas a un plazo de 24 semanas (6 meses), con seguimientos semanales, para hacer una revisión de los resultados.

En la siguiente imagen se muestra el plan de trabajo pactado para el desarrollo del proyecto:

Figura 2 Cronograma

ACTIVIDAD	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W15	W16	W17	W18	W19	W20	W21	W22	W23	W24
Apropiar conceptos JavaScript	■	■																						
Simular señales Modbus			■																					
Simular señales OPC				■																				
Simular señales DNP3					■																			
Adquisición datos Modbus						■	■																	
Adquisición datos OPC								■	■															
Adquisición datos DNP3										■														
Almacenar datos en base de datos SQL											■	■												
Obtener los datos de la base de datos del historico de alarmas													■											
Obtener los datos de la base de datos del historico de señales forzadas														■										
Procesar y transformar los datos obtenidos mediante Java script utilizando librerías															■	■	■	■						
Desarrollo aplicación Web																			■	■	■	■		
Evaluación de la herramienta																							■	■

Fuente: Elaboración propia

5. Capítulo IV: Desarrollo del trabajo

En el presente capítulo se describe el procedimiento realizado para cumplir cada uno de los objetivos descritos al inicio de este documento, dando a conocer a detalle cómo se llevaron a cabo cada una de las etapas, actividades y fases realizadas.

Para cumplir con los objetivos propuestos en cada una de las etapas de este proyecto, éstas se dividen en una serie de fases y actividades, las cuales se describen detalladamente a continuación.

5.1.1. ETAPA 1: Integración de Protocolos Industriales en una única base de datos

Esta etapa se divide en tres fases y con su culminación se dará cumplimiento al primer objetivo de este proyecto.

5.1.1.1. Fase 1. Simulación de Señales

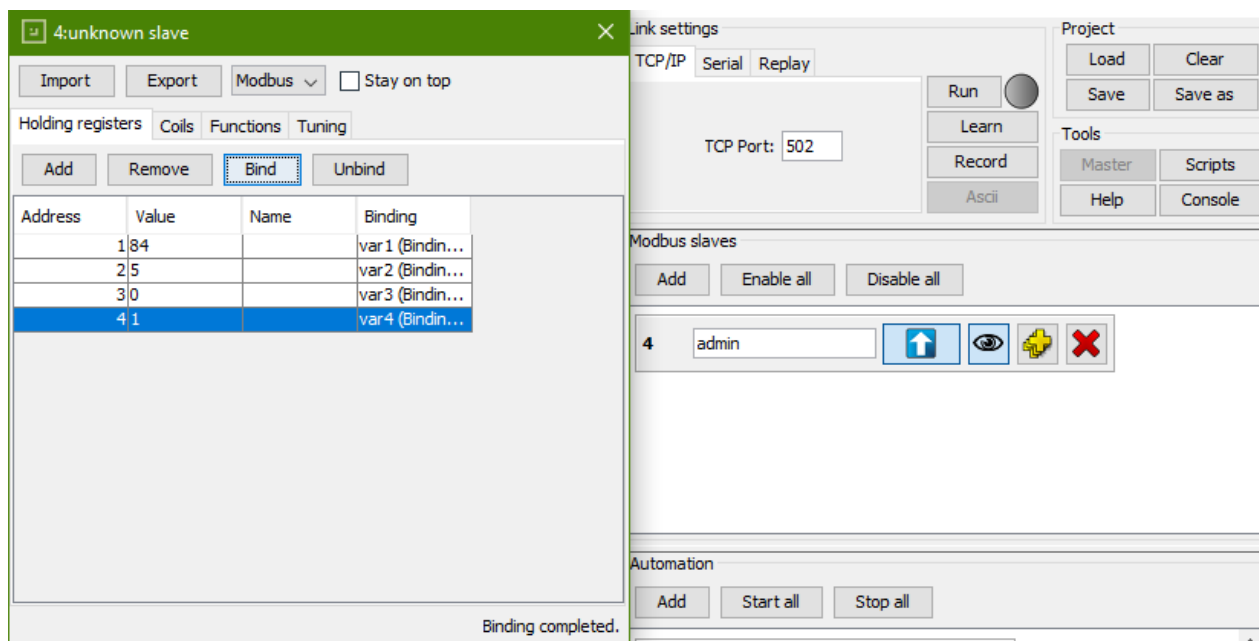
5.1.1.1.1. Actividad 1: Simular señales analógicas y discretas de Modbus TCP.

Se inicia con la simulación de señales Modbus TCP, para esto se utiliza **Modbus Pal** que es un simulador de esclavos Modbus, el cual tiene la capacidad de simular entornos Modbus complejos y realísticos.

Modbus Pal puede simular hasta 247 esclavos Modbus, cada esclavo puede tener diferentes registros. En este caso se simuló un esclavo con cuatro registros.

Cada registro se asoció a un generador de valores dinámicos llamado “automation” y así obtuvimos dos señales analógicas y dos señales discretas que generan valores aleatorios cada cierto tiempo, luego se asigna un puerto TCP, en este caso el 502 y se ejecuta.

Figura 3. Simulación de señales Modbus análogas y discretas



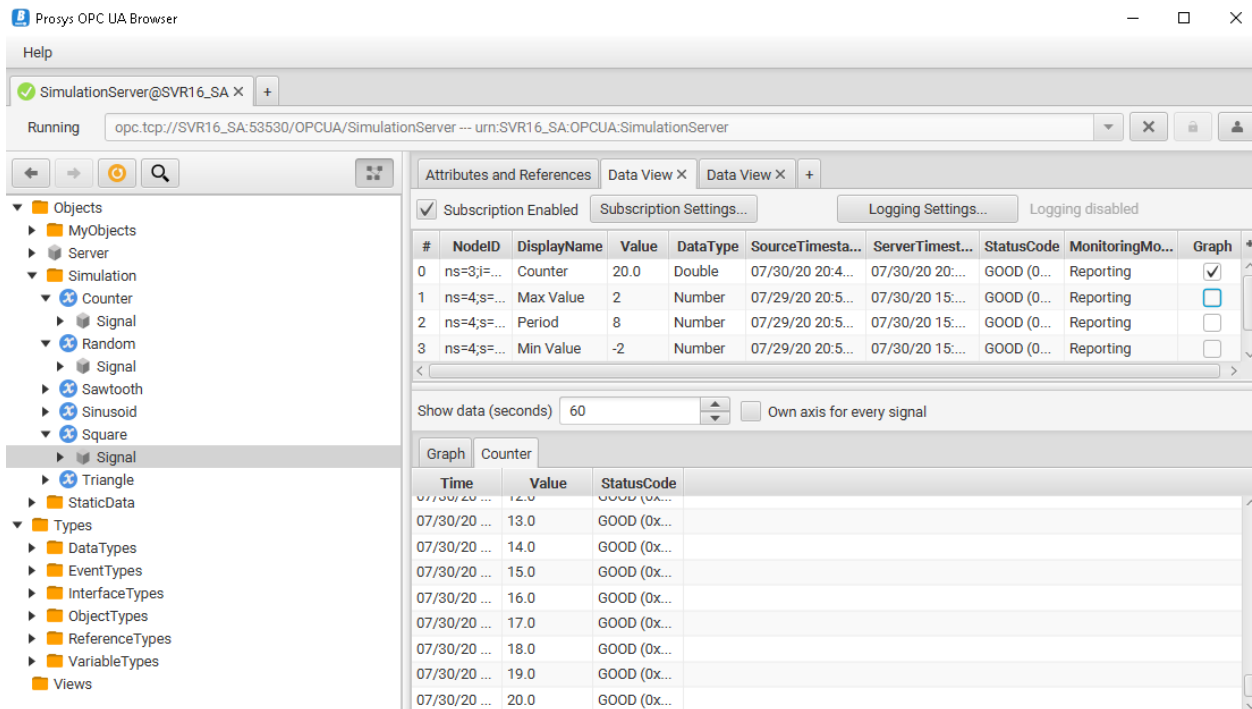
Fuente: Elaboración propia

5.1.1.1.2. Actividad 2: Simular señales análogas y discretas de OPC UA.

Para el desarrollo de esta actividad se utilizó **OPC UA Simulation Server**(Software de Simulación adquirido por InfoDesign) que es un servidor OPC que proporciona datos simulados. Dentro del servidor se crearon los nodos, cada nodo tiene un único ID, se asignó el tipo de señal a cada nodo y se empiezan a generar valores aleatorios, en este caso se simulaban dos señales análogas y dos señales discretas.

Para detalles del proceso de simulación, ver Anexo 10.2 Simulación OPC UA.

Figura 4. Simulación de señales OPC análogas y discretas



Fuente: Elaboración propia

5.1.1.1.3. Actividad 3: Simular señales análogas y discretas de DNP3.

Para finalizar la fase de simulación de señales, se procede con el protocolo DNP3. Para ello se utiliza el simulador **MatrikonOPC Server for DNP3**, en el que se configuran los parámetros necesarios como el tipo y número de señales, en este caso fueron tres señales con valores aleatorios, una digital y dos analógicas.

Figura 5. Simulación de señales análogas y discretas

Current configuration:

- Server Configuration
 - Network Channel
 - Host
 - Dnp3 Unit**
- Alias Configuration

DNP3 Unit settings for 'Dnp3 Unit'

Name: Enabled

Description:

Unit Settings:

Communication | Data Acquisition | Optimization

Master:

Station:

Base poll time: Hora est. Pacífico, Sudamérica minutos

Time bias:

Operate outputs: Allow writes

Synchronize time: Latch data

Log static objects

Log event objects

Sent: 0
 Received: 0
 Timed Out: 0
 Retried: 0
 Failed: 1
 Overrun: 0

Reset Statistics

Apply Cancel

Contents of 'Group0'

Item ID	Access Path	Value	Quality	Timestamp	Status
Random.Boolean		False	Good, non-specific	10/15/202...	Active
Random.DoubleFloat		6915	Good, non-specific	10/15/202...	Active
Random.Real8		5917,018...	Good, non-specific	10/15/202...	Active

Fuente: Elaboración propia

5.1.1.2. Fase 2: Adquisición de los datos.

El Plan definido para la adquisición de los datos contempla lo siguiente:

- Banco de Pruebas con Simuladores de datos digitales y analógicos emulando los tres protocolos: Modbus TCP, OPC UA y DNP3
- Almacenamiento de la información en bases de datos SQL a través de la librería mysql de Node.js
- Uso de la librería Socket.io de Java Script para actualizar la base de datos en tiempo real

Se hace la adquisición de los datos con el fin de posteriormente unificar la información en una base de datos SQL y así poder generar información que facilite la toma de decisiones a partir de ellos.

Este segmento del código habilita la importación de la librería **Socket.io** que permite establecer la comunicación bidireccional entre clientes y servidores web en tiempo real. De igual manera se detalla el código para importar la librería Express, la cual facilita la integración de la información vía HTTP.

```
const express = require("express");  
const socketIO = require("socket.io");
```

5.1.1.2.1. Actividad 1: Obtención de los datos OPC.

En este segmento del código, en el mismo entorno de ejecución (Node.js) se importa la librería **node-opcua**, esta librería es la que permite obtener los datos OPC del simulador.

```
const {  
  AttributeIds,  
  OPCUAClient,  
  TimestampsToReturn,  
} = require("node-opcua");
```

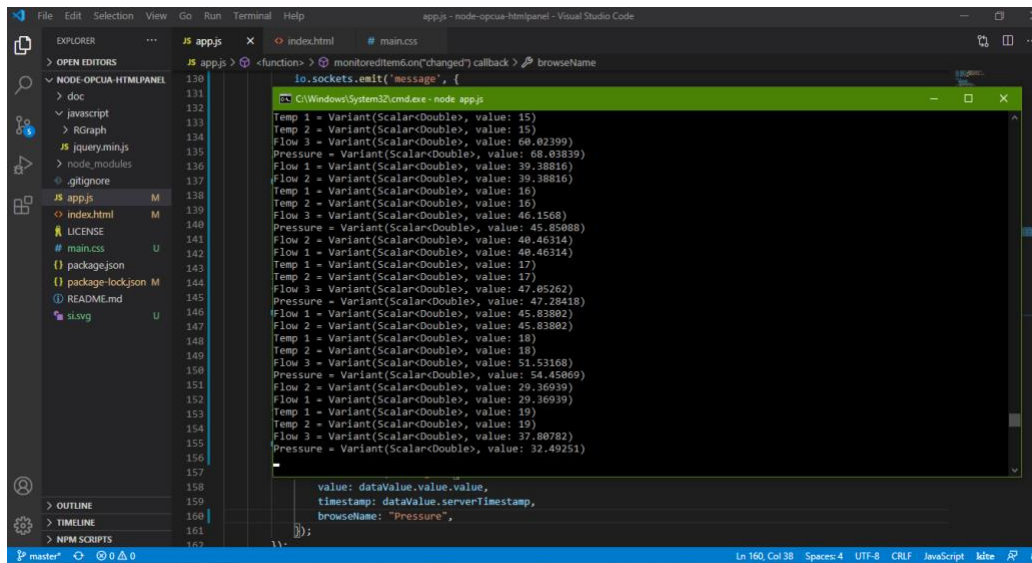
En este segmento del código se asigna una variable para cada Id de cada nodo OPC, al igual que para el endpoint que es el punto de acceso al servidor OPC.

```
const endpointUrl = "opc.tcp://LAPTOP-DGV3S6G0:53530/OPCUA/SimulationServer";  
const nodeIdToMonitor = "ns=3;i=1010;s=Flow1";  
const nodeIdToMonitor2 = "ns=3;i=1011;s=Flow2";  
const nodeIdToMonitor3 = "ns=3;i=1007;s=Temp1";  
const nodeIdToMonitor4 = "ns=3;i=1008;s=Temp2";  
const nodeIdToMonitor5 = "ns=3;i=1012;s=Flow3";  
const nodeIdToMonitor6 = "ns=3;i=1009;s=Pressure";
```

Posteriormente se crea el cliente OPC, se establece la conexión con el simulador y se habilita la lectura de los datos OPC suministrados por el simulador

En la figura 4 se puede observar la lectura de los datos que se están obteniendo del servidor, vistos en la consola.

Figura 6. Lectura de las señales obtenidas del simulador



Fuente: Elaboración propia

5.1.1.2.2. Actividad 2: Obtener los datos Modbus TCP.

En este segmento del código en el entorno de ejecución Node.js se importa la librería **Jsmodbus** que nos facilita la comunicación y lectura de datos con dispositivos que utilizan este protocolo.

Se crea el cliente Modbus TCP para establecer la conexión con el simulador como se muestra a continuación:

```
let Modbus = require("jsmodbus");
let socketmb = new net.Socket();
let client = new Modbus.client.TCP(socketmb);
let options = {
  host: "127.0.0.1",
  port: 502,};
```

Este segmento del lenguaje JavaScript se encarga de habilitar la lectura de los registros en protocolo Modbus:

```
client.readHoldingRegisters(0,4)
  .then(function(data) {
```

```
console.log(data.response._body.valuesAsArray.toString());  
let variable = data.response._body.valuesAsArray[0].toString();  
let variable2 = data.response._body.valuesAsArray[1].toString();  
let variable3 = data.response._body.valuesAsArray[2].toString();  
let variable4 = data.response._body.valuesAsArray[3].toString();  
}
```

5.1.1.2.3. Actividad 3: Obtener los datos DNP3.

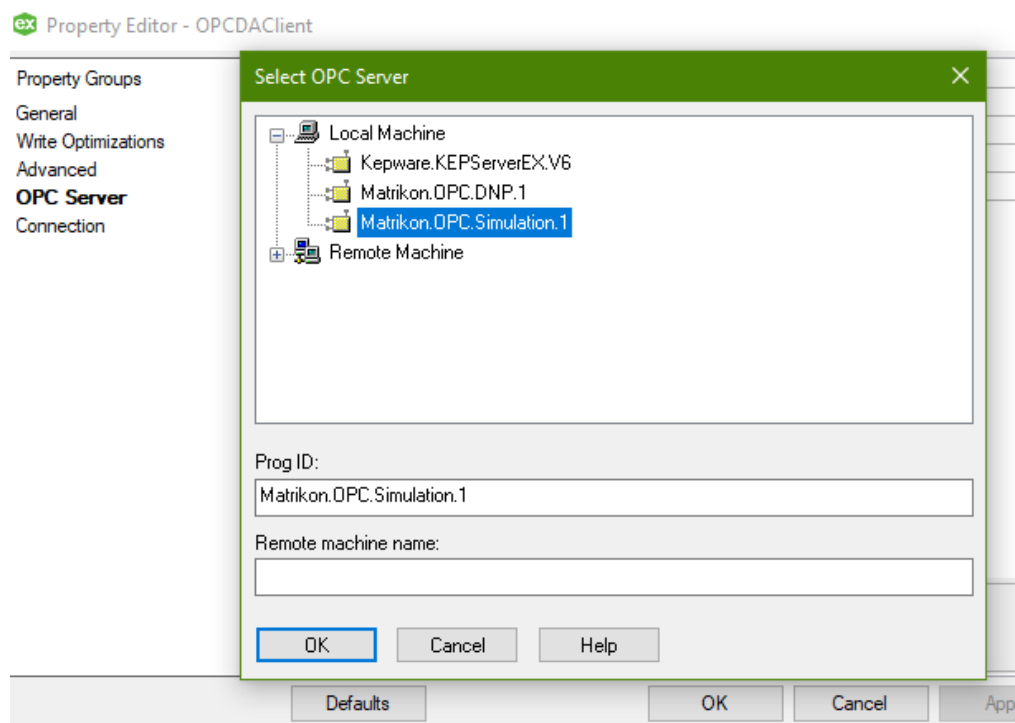
Finalmente, en esta actividad se obtienen los datos DNP3. En aras de optimizar costos y simplificación de los algoritmos de adquisición usamos las herramientas de conversión de protocolo DNP3 a OPC embebidas en el software MatrikonOPC for DNP3 asegurando que la estampa de tiempo y la calidad del dato se mantiene en todo momento.

Es necesario la creación de un túnel para convertir OPC DA a OPC UA utilizando el software KEPServer.

A continuación se detallan los parámetros de configuración del KEPServer para la conversión a OPC UA.

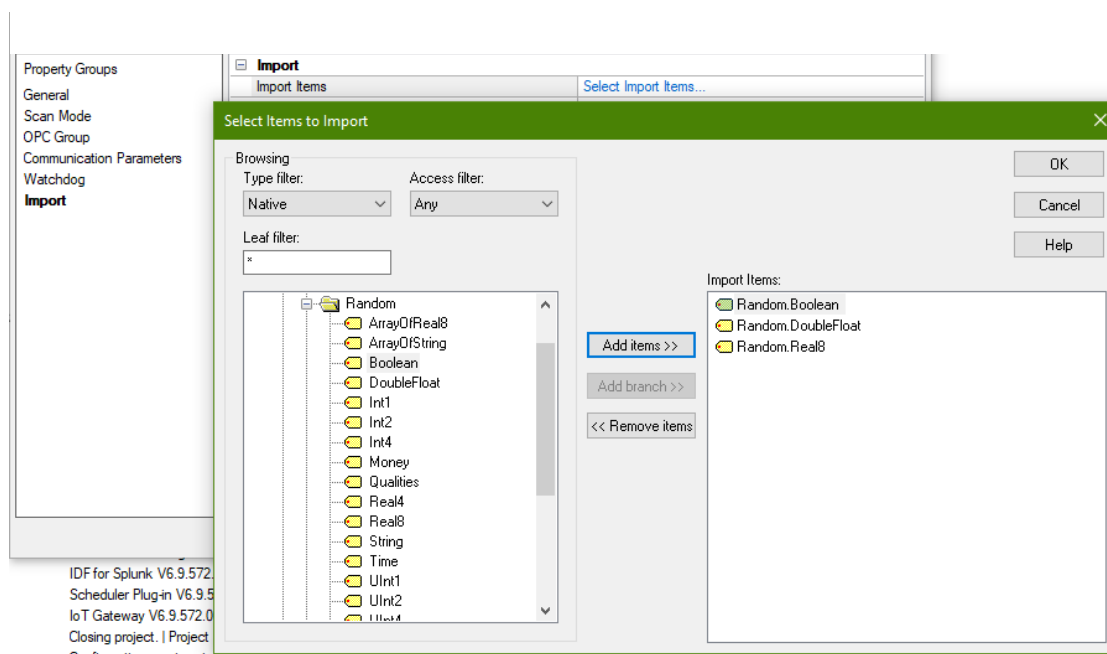
Una vez se han simulado las señales, se configura el túnel en el lado del servidor DA. Para crear el canal de comunicación, se selecciona el driver que tiene los datos OPC DA, en este caso MatrikonOPC, una vez creado el canal de comunicación se añade el tag de cada dato y se verifica que lea las señales correctamente como se ve en las figuras 6 y 7.

Figura 7. Canal de comunicación con OPC DA



Fuente: Elaboración propia

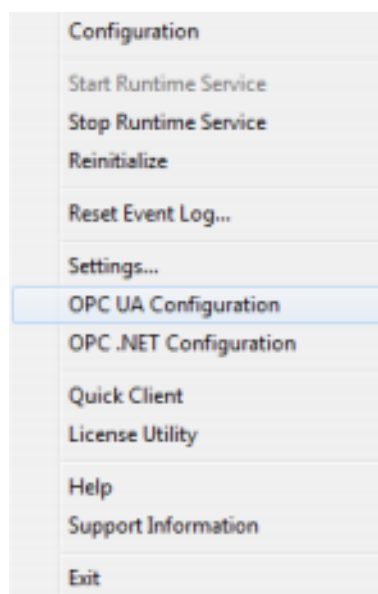
Figura 8. Tags seleccionadas



Fuente: Elaboración propia

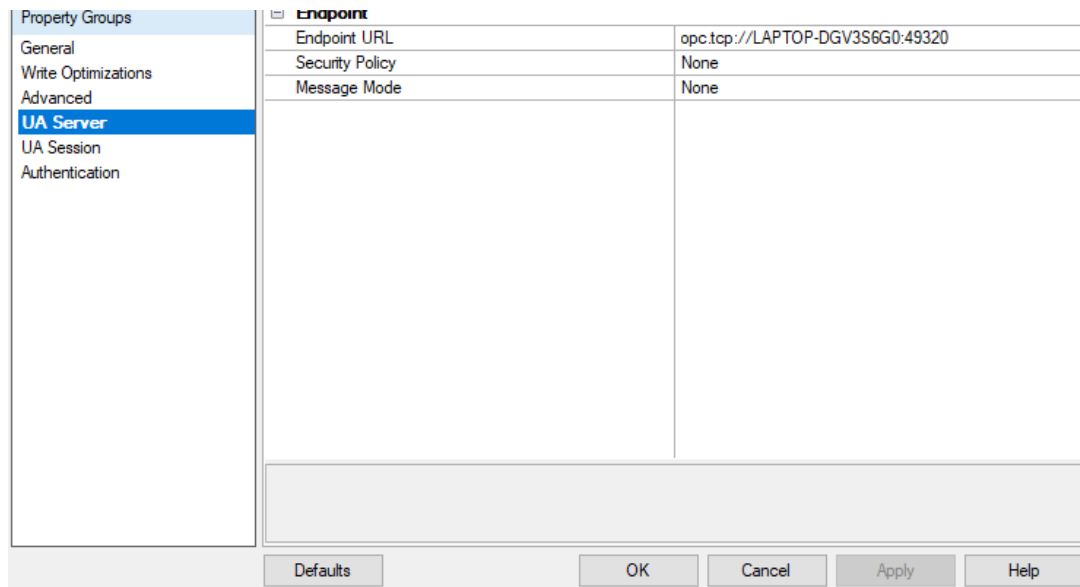
Posteriormente se configura el servidor OPC UA; en el administrador de configuraciones se selecciona la ventana de endpoints que es el punto de acceso al servidor OPC, una vez obtenido el endpoint se selecciona importar elementos y se seleccionan las tags que se añadieron anteriormente provenientes del simulador.

Figura 9. Configuración OPC UA



Fuente: Elaboración propia

Figura 10. Endpoint OPC UA



Fuente: Elaboración propia

Una vez se tienen las tags y el endpoint es posible establecer la comunicación y leer los datos como se hizo anteriormente con los datos OPC.

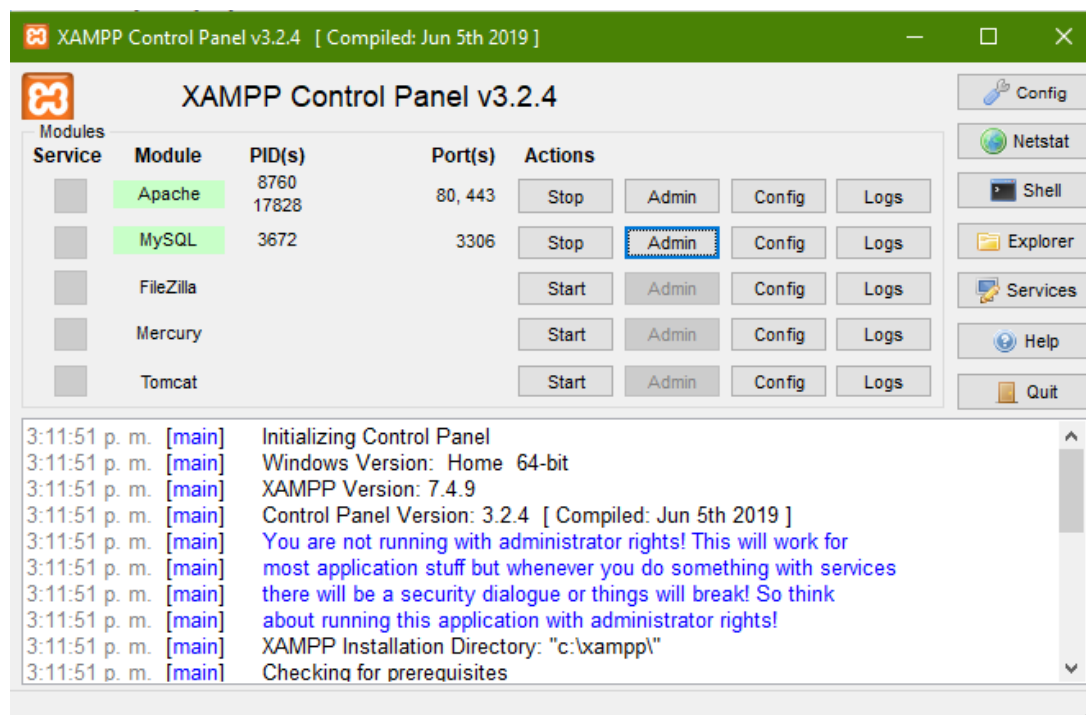
5.1.1.3. Fase 3: Almacenar los datos obtenidos en una base de datos relacional MySQL.

5.1.1.3.1. Actividad 1: Creación de la base de datos.

Se crea una base de datos en phpMyAdmin que es una herramienta escrita en PHP Y permite manejar y administrar bases de datos de MySQL.

En esta base de datos se van a almacenar los datos obtenidos en la fase anterior.

Figura 11. Configuración de PhpMyAdmin con XAMPP



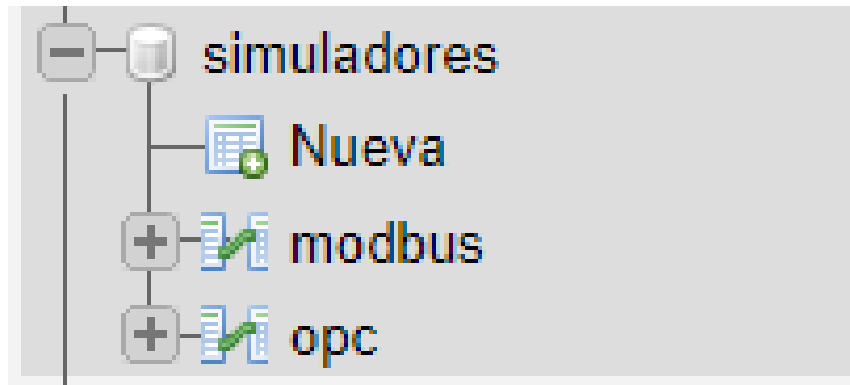
Fuente: Elaboración propia

5.1.1.3.2. Actividad 2: Creación de las tablas.

Posteriormente se crearon tres tablas en las cuales se van a almacenar los datos, una tabla para cada protocolo (Modbus TCP, OPC y DNP3).

Cada tabla tiene 4 parámetros que son: ID, NOMBRE, VALOR y FECHA.

Figura 12. Base de datos y tablas creadas para almacenar los datos



Fuente: Elaboración propia

5.1.1.3.3. Actividad 3: Almacenar los datos obtenidos en la base de datos.

Para el desarrollo de esta actividad es necesario establecer la conexión entre MySQL y Node.Js, a continuación se muestra el código para habilitar los queries (consultas) .

En esta parte del código se habilita la importación de las librerías necesarias para conectarse con la base de datos:

```
const express = require("express");
const mysql = require('mysql');
```

Posteriormente se establece la conexión con la base de datos creada anteriormente y por último se hacen las consultas en la base de datos, para actualizarla cada que se presente un cambio en la señal.

Figura 13. Tabla con los datos OPC

id	Nombre	Valor	Fecha
1	Flow 1	1.69	2020-09-16 14:19:14
2	Flow 2	true	2020-09-16 14:19:14
3	Temp 1	5.00	2020-09-16 14:19:14
4	Temp 2	5.00	2020-09-16 14:19:14
5	Flow 3	true	2020-09-16 14:19:14
6	Pressure	1.69	2020-09-16 14:19:14
7	Temp 1	6.00	2020-09-16 14:19:18
8	Flow 1	-0.39	2020-09-16 14:19:18
9	Flow 2	false	2020-09-16 14:19:18
10	Temp 2	6.00	2020-09-16 14:19:18
11	Flow 3	false	2020-09-16 14:19:18
12	Pressure	-0.39	2020-09-16 14:19:18
13	Temp 1	7.00	2020-09-16 14:19:24

Fuente: Elaboración propia

Con esto se da cumplimiento al primer objetivo de este trabajo: Integrar datos generados por diferentes protocolos industriales en una única base de datos.

5.1.2. ETAPA 2. Generación de KPIs y dashboards

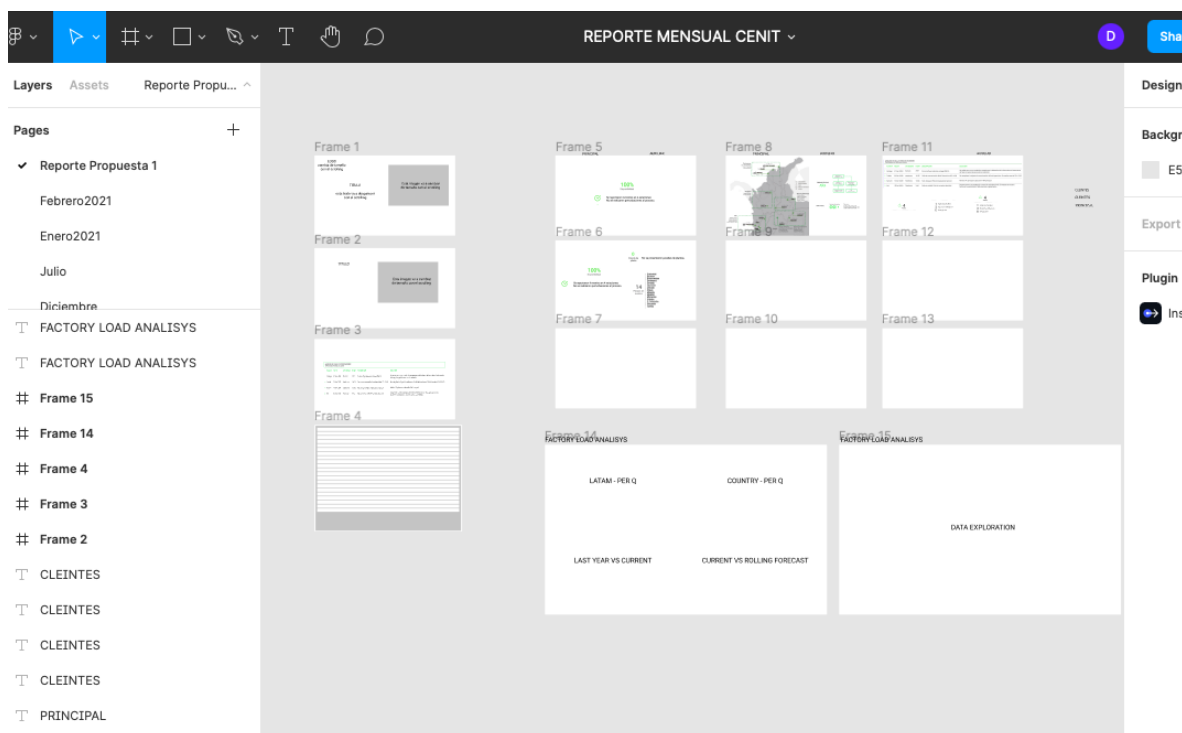
5.1.2.1. Fase 1: Visualización y monitoreo de señales.

5.1.2.1.1. Actividad 1. Crear archivos SVG.

Para el desarrollo de esta actividad se crearon diferentes imágenes en formato SVG en la aplicación FIGMA.

A solicitud de InfoDesign se desarrolló la visualización de diferentes estaciones de bombeo de crudo, los cuales se van a utilizar en aplicativo web para permitirle al usuario final visualizar, monitorear y analizar datos de sus procesos en tiempo real.

Figura 14. Imágenes creadas en Figma



Fuente: Elaboración propia

5.1.2.1.2. Actividad 2: HTML.

Posteriormente, en esta actividad se desarrolla el código HTML, CSS y la publicación de los datos de campo simulados en el servidor web.

En este segmento del código se establece la comunicación del Frontend con el Backend usando la librería socket.io. De igual manera se hace la publicación de los datos en el servidor web

```
const socket = io.connect("http://localhost:3700");

socket.on('message', function (data){

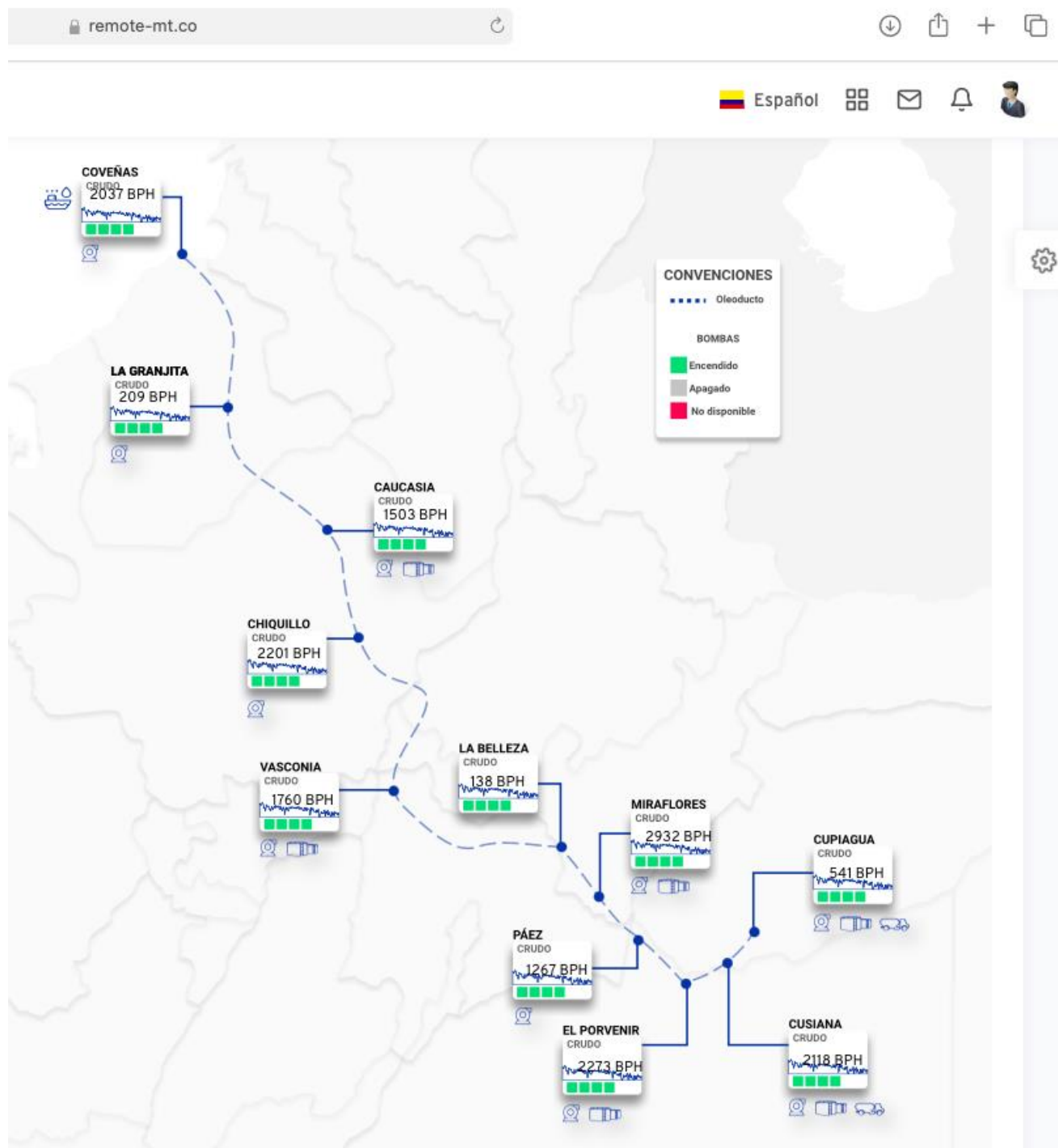
TextCov.innerHTML = data.value + "BPH";
TextGranjita.innerHTML = data.value + "BPH";
TextCaucasia.innerHTML = data.value + "BPH";
TextChiquillo.innerHTML = data.value + "BPH";
TextVasconia.innerHTML = data.value + "BPH";
TextBelleza.innerHTML = data.value + "BPH";
TextPaez.innerHTML = data.value + "BPH";
TextPorvenir.innerHTML = data.value + "BPH";
TextCopiagua.MirafloresHTML = data.value + "BPH";
TextCusiana.innerHTML = data.value + "BPH";
```

5.1.2.1.3. Actividad 3: CSS

En este código se ajustan los estilos, fuente , color, margen y posición de cada elemento.

La Figura 19 muestra los datos más importantes de un conjunto de estaciones de transporte de crudo, tales como flujo, disponibilidad de las bombas booster entre otros.

Figura 15. Datos publicados en el servidor.



Fuente: Elaboración propia

5.1.2.2. Fase 2: Generación de KPIs.

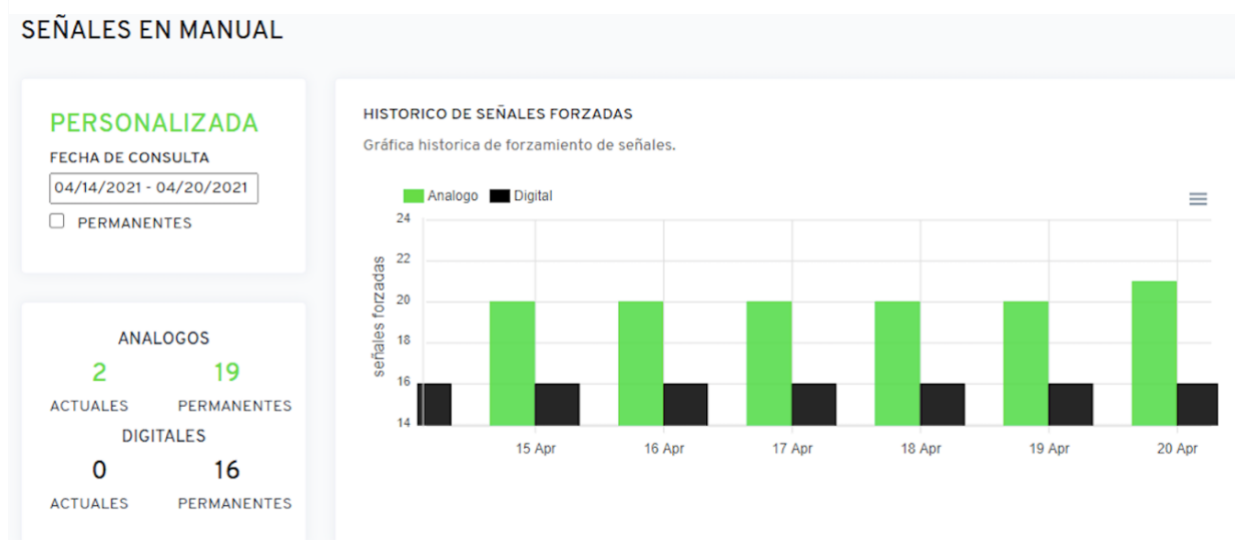
5.1.2.2.1. Actividad 1: Graficar

Para el desarrollo de esta actividad se establece una conexión con una base de datos SQL que almacena datos históricos de señales forzadas análogas y digitales, también históricos de alarmas, una vez obtenidos los datos se procede a graficarlos con ayuda de la librería Chart.js.

Durante el proceso de investigación de requerimientos por parte del personal operativo de diferentes plantas del sector productivo que actualmente trabajan con InfoDesign, se identificaron patrones comunes, todos relacionados con el manejo de base de datos de control(**Señales con su modo de operación en manual, alarmas promedio, acciones del operador depuración de registros y/o log**) y administración de alarmas y acciones operativas.

En la figura 19 se puede observar el KPI para el área de operaciones que muestra la cantidad de señales en manual o forzadas en un periodo de tiempo establecido por el usuario, a través de la interfaz web. Este KPI le permita identificar a los administradores de los sistemas de control, instrumentación en falla, señales en condición subestandar entre otras.

Figura 16. KPI señales forzadas del último mes



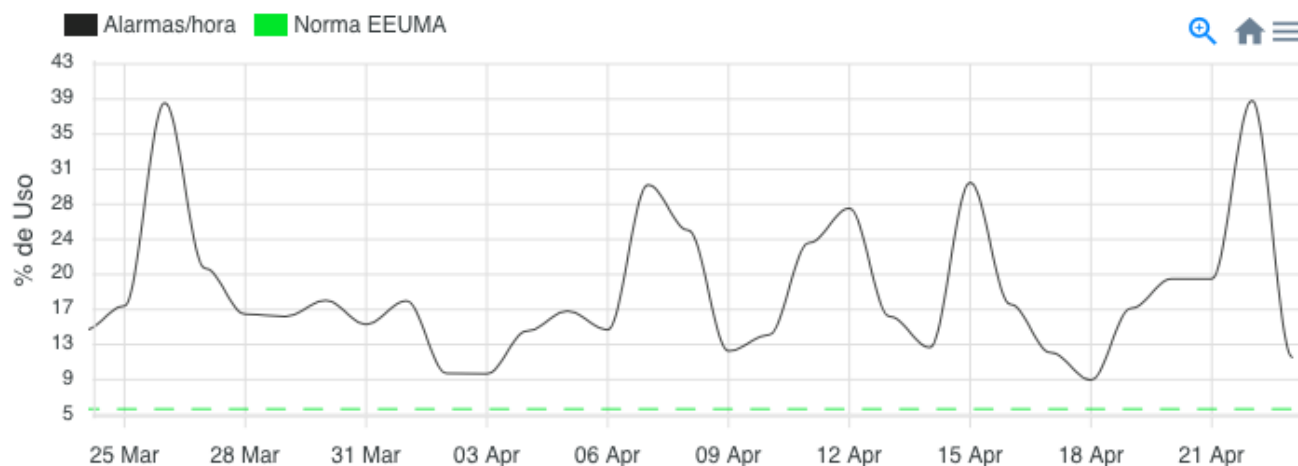
Fuente: Elaboración propia

Ahora bien, es muy frecuente que las aseguradoras identifiquen si en las compañías existen sistemas de gestión y/o medición de alarmas de proceso para cada puesto operacional, y dentro del desarrollo base, incluimos un KPI que muestra con frecuencia diaria la cantidad de alarmas recibidas por hora Vs la normativa EEMUA 191, y de esta manera identificar si la planta o puesto operacional, están en cumplimiento de la norma, como se puede ver a continuación en la Figura 20:

Figura 17. KPI histórico de alarmas del último mes.

HISTORICO ALARMAS/HORA

Este indicador sirve para determinar la cantidad de alarmas/hora del sistema cumpliendo la recomendación EEMUA.



Fuente: Elaboración propia

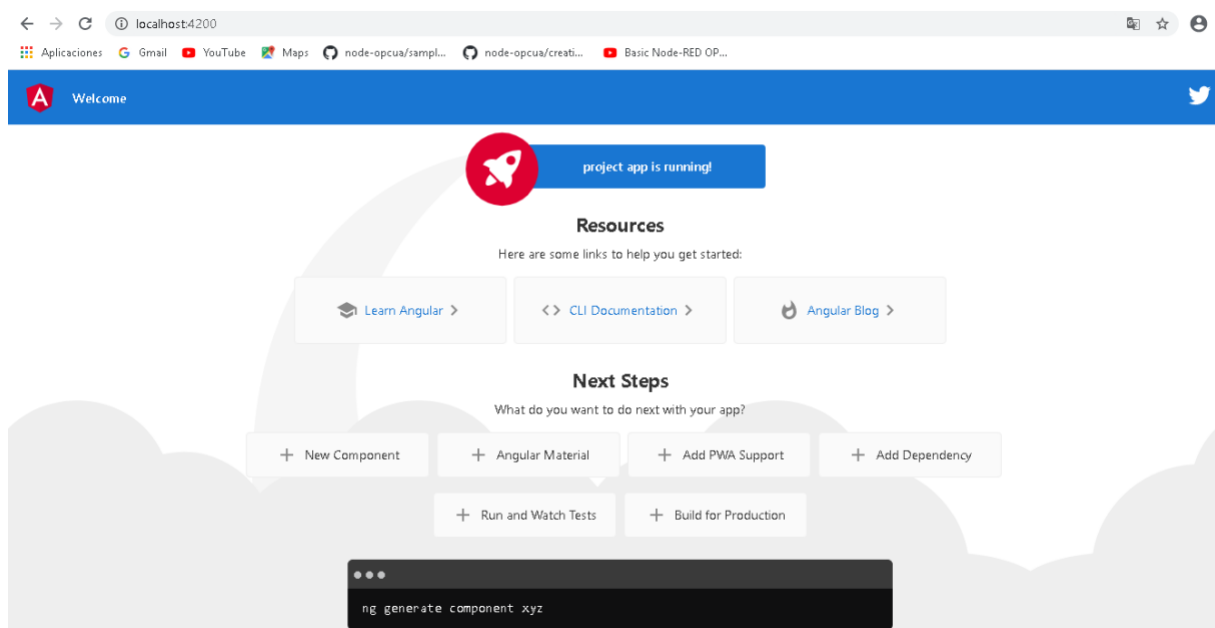
Con esto se da cumplimiento al segundo objetivo de este trabajo, ya que se desarrolló un dashboard e indicadores de desempeño, transformando así los datos en información útil para el usuario.

5.1.3. ETAPA 3. Crear aplicativo web responsive en Angular

5.1.3.1. Fase 1: Migración del aplicativo web a Angular.

5.1.3.1.1. Actividad 1: Desarrollo de Frontend.

Para el desarrollo de esta actividad se instaló Angular CLI, una herramienta de línea de comandos que facilita la creación, generación y ejecución de proyectos, ya que ayuda a armar la base y estructura del proyecto propia de angular.

Figura 18. Angular CLI

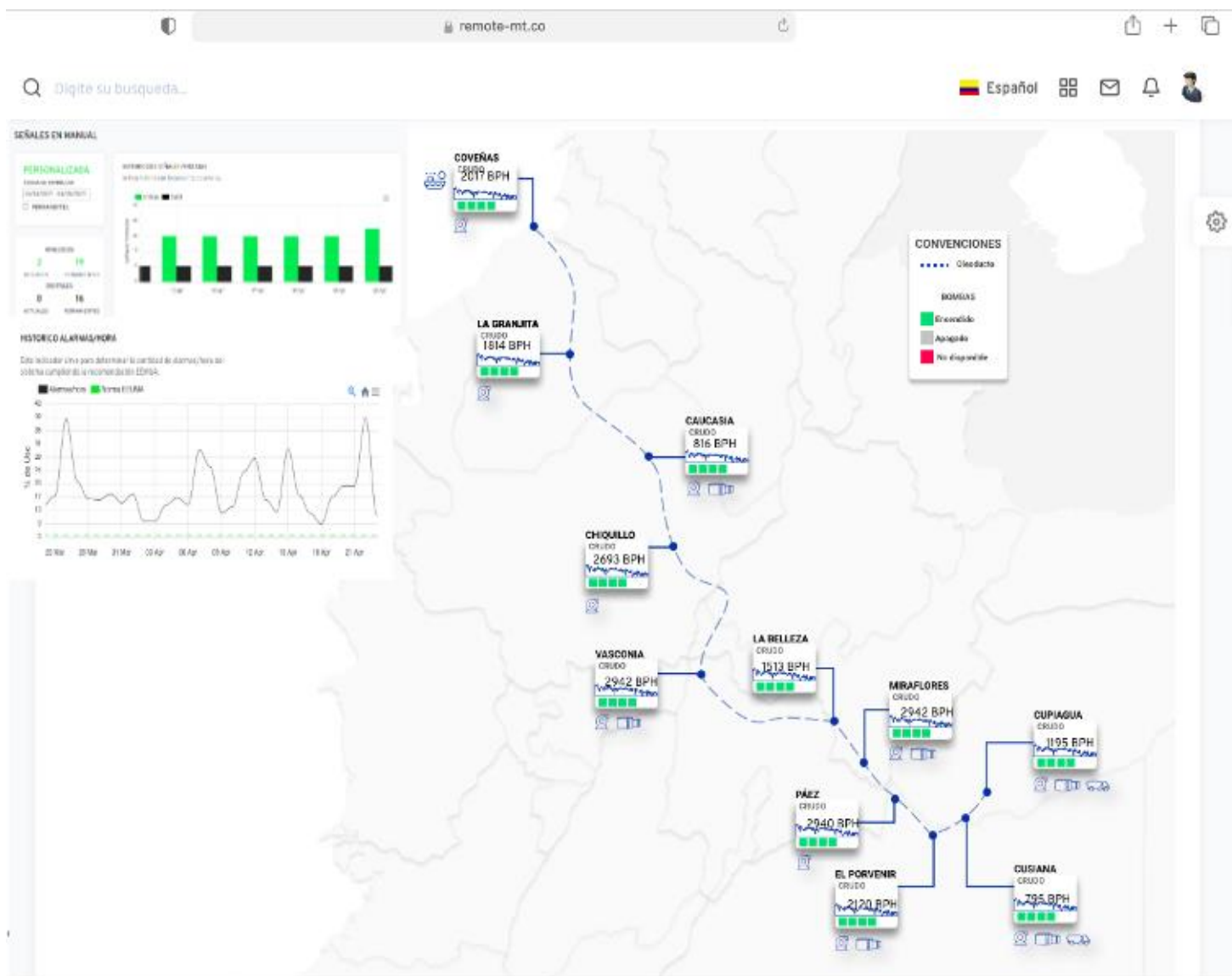
Fuente: Elaboración propia

Primero se habilita un servicio que es el encargado de conectar el Frontend con el Backend por medio de una petición HTTP.

Una vez establecida la conexión entre Backend y Frontend, se crea un componente en el que se desarrolla el código encargado de leer los datos y enviarlos a la vista HTML.

Finalmente se desarrolla el código HTML donde se va a habilitar la publicación de los datos en el servidor web.

Figura 19. Aplicativo Web Final





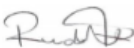

Fuente: Elaboración propia

En la figura 17 se observa el resultado final del aplicativo web, donde se pueden monitorear los datos mas importantes de un conjunto de estaciones de transporte de crudo, y también el histórico de alarmas y señales forzadas en determinado periodo de

tiempo seleccionado por el usuario; con esto se da cumplimiento al tercer objetivo de este proyecto.

6. Validación y aceptación del prototipo

Figura 20 Aceptación del proyecto

		FORMATO PRUEBAS PROTOTIPO DE APLICATIVO WEB					ANEXO	
							Rev	
							0	
INVENTARIO DE EQUIPOS								
Item	Descripción	Aprobado						
		SI	NO	N/A				
1	Señales Modbus TCP	X						
	Señales OPC UA	X						
	Señales DNP3	X						
2	Módulo de adquisición de datos	X						
	Módulo de almacenamiento de datos	X						
3	Módulo para generar KPIs	X						
4	Aplicativo web responsive con los resultados obtenidos	X						
5								
OBSERVACIONES (EN CASO DE SER NECESARIO)								
INSPECCIÓN		FIRMA			INSPECCIÓN		FIRMA	
Por SCHNEIDER ELECTRIC:					Por InfoDesign :			
Ricardo Neira					Sergio Gonzales			

Fuente: InfoDesign-Schneider Electric

7. Lecciones aprendidas

La ejecución de este proyecto dejó aprendizajes y experiencias que permitieron culminar con éxito los objetivos planteados.

1. Los desarrollos web responsive requieren siempre iniciar los diseños usando como plataforma base el diseño para los dispositivos móviles, y obliga a descartar el uso de librerías pesadas, aplicaciones flash e impone ciertos retos y restricciones en el diseño.
2. Aunque los inicios del desarrollo del Frontend de este proyecto fue realizado en JavaScript, la solución final seleccionada fue Angular, debido su facilidad de programación, integrabilidad de módulos, implementación de framework web responsive y estabilidad del código.
3. Debido a COVID-19, gran parte de las compañías del sector industrial han abierto sus sistemas, para la integración de la información de proceso a cada uno de los diferentes niveles corporativos, con el objetivo de optimizar sus procesos y facilitar la toma de decisiones, sin embargo es clave plantear soluciones escalables y de un costo que le permita a la pequeña y mediana empresa el acceso a este tipo de soluciones.
4. Nuevos conocimientos tales como configuración en JavaScript, Angular, manejo de bases de datos SQL, manejo de diferentes protocolos

industriales, soluciones cloud, y tratamiento y filtrado de datos fue parte del aprendizaje requerido, para culminar este proyecto de Grado.

8. Conclusiones

El desarrollo de este proyecto de grado, fue un desafío mayor, debido a que para culminarlo con éxito, fue necesario desarrollar nuevos conocimientos, adentrarse en las necesidades y realidades de diferentes sectores de la industria de Colombia en temas de transformación digital, teniendo claro que para lograr darle movilidad a la información de proceso, la integración de Tecnologías de la Operación con Tecnologías de la Información era el siguiente escalón en esta jornada digital. A continuación las conclusiones mas relevantes del proyecto:

1. El uso de metodologías ágiles(Scrum) y la mezcla de diferentes módulos tales como el uso de los servicios cloud, implementación de algoritmos web, simuladores de protocolos industriales, unificación de información en bases de datos SQL nos permitió lograr con éxito la Integración de diferentes protocolos industriales(Modbus, OPC y DNP3), manteniendo la integridad y calidad de los datos.
2. El desarrollo de KPIs e integración de datos en frameworks web, son la respuesta a una necesidad identificada en gran parte de las compañías con las que InfoDesign ha trabajado los últimos tres(3) años; y permite el fácil acceso e integración de los mundos de TO(Tecnologías de la Operación) y TI(Tecnologías de la Información) de manera confiable y con costos de implementación bajos. La funcionalidad de la interfaz gráfica y los indicadores desarrollados(Alarmas y Señales en Manual) para el proyecto fueron

culminados a satisfacción y probados de manera exitosa en diferentes compañías para los que InfoDesign presta servicios.

3. Los desarrollos web en la actualidad deben ser adaptativos o responsive, ya que el uso de dispositivos móviles para consultarlos es cada vez mas frecuente por parte de los usuarios finales, y para nuestro caso de estudio particular, era una característica que debíamos incluir. Los diseños base, se hicieron en angular, inicialmente pensando en un ambiente móvil, pero en su etapa final, nos aseguramos de su completa funcionalidad en ambos ambientes. El comportamiento en cada uno de los diferentes navegadores como Chrome, Safari, Firefox arrojó los resultados esperados, sin demoras en la actualización de los datos, ni las consultas para la generación de los indicadores de desempeño establecidos.

4. Durante el desarrollo de este proyecto, se llegó a la conclusión que la integración de tecnologías de la Operación con tecnologías de la Información, es el paso que las compañías están dando para aumentar la productividad, apalancándose en las herramientas digitales con las que se cuenta en la actualidad. Es una realidad que hay una alta demanda de soluciones de software, dentro del estudio realizado por InfoDesign con empresas del sector productivo en Colombia para identificar necesidades comunes en temas relacionados con transformación digital, encontramos que la pequeña y mediana empresa en Colombia tiene claro que sus planes de desarrollo deben

incluir inversiones en transformación digital de sus procesos, específicamente en movilidad de la información, analítica de datos, soluciones cloud, machine learning y gemelos digitales, entre otros; sin embargo, se evidenciaron limitaciones al acceder a este tipo de soluciones, debido a su alto costo y donde la base inicial, requiere un nivel de automatización alto de los procesos, para poder asegurar que la adquisición de los datos se haga a través de protocolos conocidos y de esta manera evitar reprocesos manuales.

Con base en todo lo anterior y con una necesidad común identificada para el sector industrial se decidió realizar este proyecto, para definir una base tecnológica que permita: la estandarización e integración de diferentes fuentes de datos, dar movilidad a la información entre los mundos de TI y TO y generar información importante a través de KPI y tableros de control con soluciones web responsive.

La suma de las partes descritas en sí, son una solución costo efectiva, que ayudará a cerrar esa brecha tecnológica con la pequeña y mediana empresa generando un impacto directo en la productividad de las mismas.

9. Bibliografía

Orozco, E. A. (2006). *Los protocolos de comunicacion en el entorno industrial, su fundamento y su importancia en el sistema de automatizacion de una planta de generacion de energia geotermica.* guatemala.

Seshadri, S. (2018). *Angular UP and Running.* O'Reilly Media, Inc.

R Hernandez, C. F. (s.f.). *Metodología de la investigación* . Ciudad de Mexico: McGraw-Hill.

H, S. (2014). *Java the Complete Reference* . New York: McGraw-Hill Education.

Essentials of socket.io. (2020). Macque Terrain Publications.

Z, K. (2012). *Building Web Applications.* O'Reilly Media Incorporated.

G, C. (2004). *Practical Modern Scada Protocols.* Burlington: Elsevier.

ecom-ex.com. (30 de Julio de 2018). *ECOM-EX.COM.* Obtenido de Sistema Ciberfisico (CPS):

[https://www.ecom-ex.com/es/seguridad-intrinseca/glosario/termino/cyber-physical-system/#:~:text=Un%20sistema%20ciberf%C3%ADsico%20\(Cyber%2Dphysical,de%20partes%20mec%C3%A1nicas%20o%20electr%C3%B3nicas.&text=Los%20sistemas%20ciberf%C3%ADsicos%20son%20una,s](https://www.ecom-ex.com/es/seguridad-intrinseca/glosario/termino/cyber-physical-system/#:~:text=Un%20sistema%20ciberf%C3%ADsico%20(Cyber%2Dphysical,de%20partes%20mec%C3%A1nicas%20o%20electr%C3%B3nicas.&text=Los%20sistemas%20ciberf%C3%ADsicos%20son%20una,s)

A, G. (11 de Diciembre de 2020). *La industria 4.0 en la sociedad digital.* Obtenido de

https://books.google.com/books?id=YnSIDwAAQBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

seuba, L. i. (2019). *Internet de las cosas. La transformación digital de la sociedad.* Madrid: Rama.

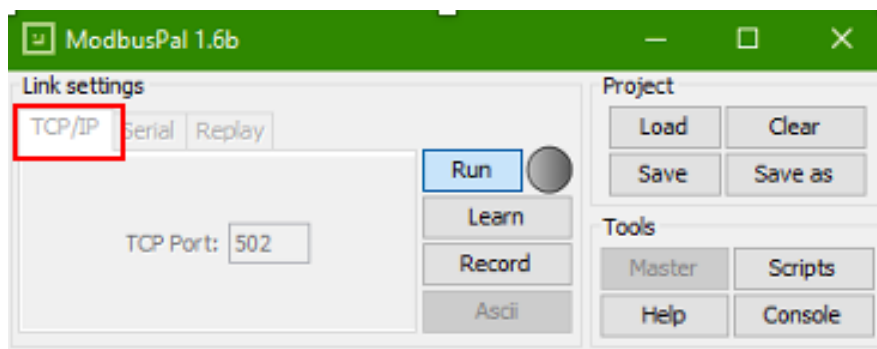
www.oracle.com. (2019). Obtenido de Que es una base de datos relacional?:

<https://www.oracle.com/ar/database/what-is-a-relational->

10. Anexos

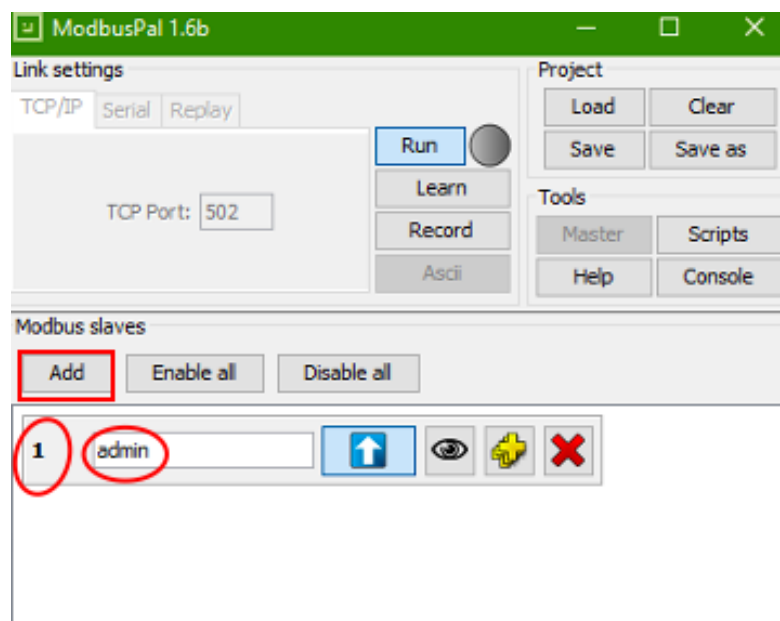
10.1. Simulación Modbus TCP

- Descargar ModbusPal y abrir la aplicación y seleccionar TCP/IP



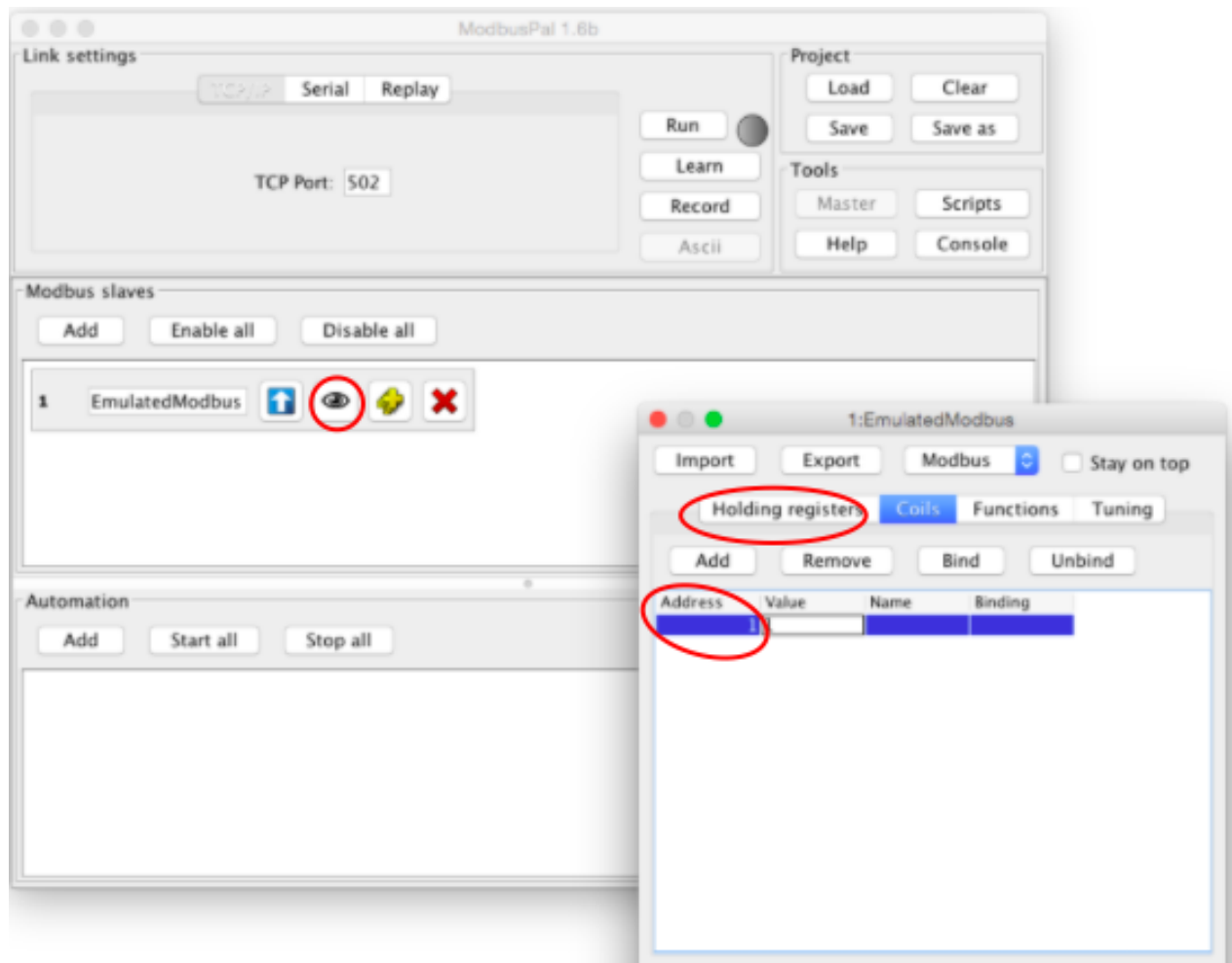
Fuente: Elaboración propia

- Clic en el botón agregar para crear un esclavo Modbus, se selecciona la dirección y se pone un nombre



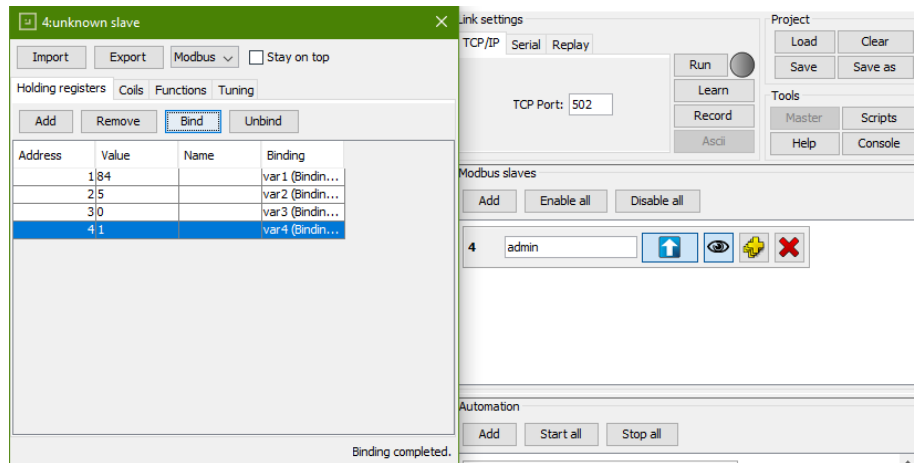
Fuente: Elaboración propia

- Clic en el botón en forma de ojo, holding registers y se asigna la dirección del esclavo que se creó anteriormente



Fuente: <https://esf.eurotech.com/docs/modbus-application-in-kura-wires>

- En la sección de Automation se hace clic en Add, clic en random generator, se establece un rango de valores y se le asigna un nombre, esto para generar valores aleatorios
- Clic en el botón del ojo nuevamente y a cada registro que se agregó se hace clic en Bind y se asocia con el generador de valores aleatorios creado.

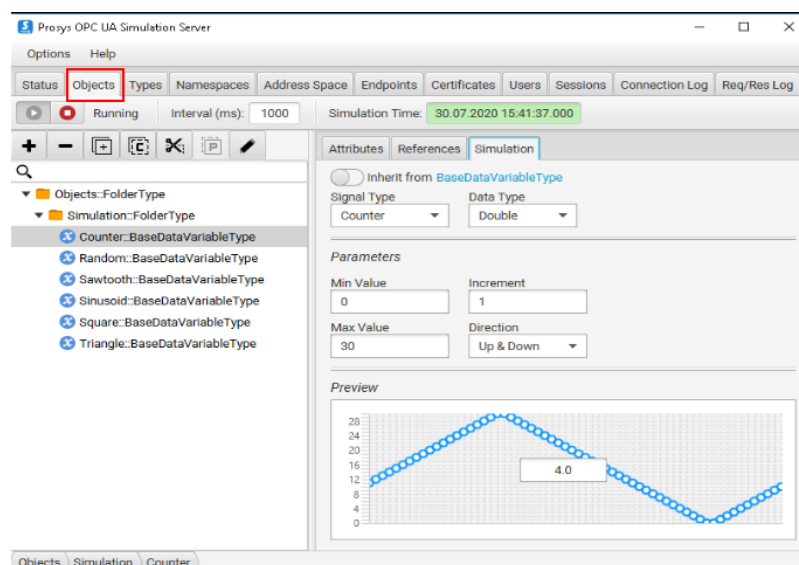


Fuente: Elaboración propia

- Se establece el puerto TCP, por defecto es el 502 y se ejecuta la aplicación haciendo clic en Run

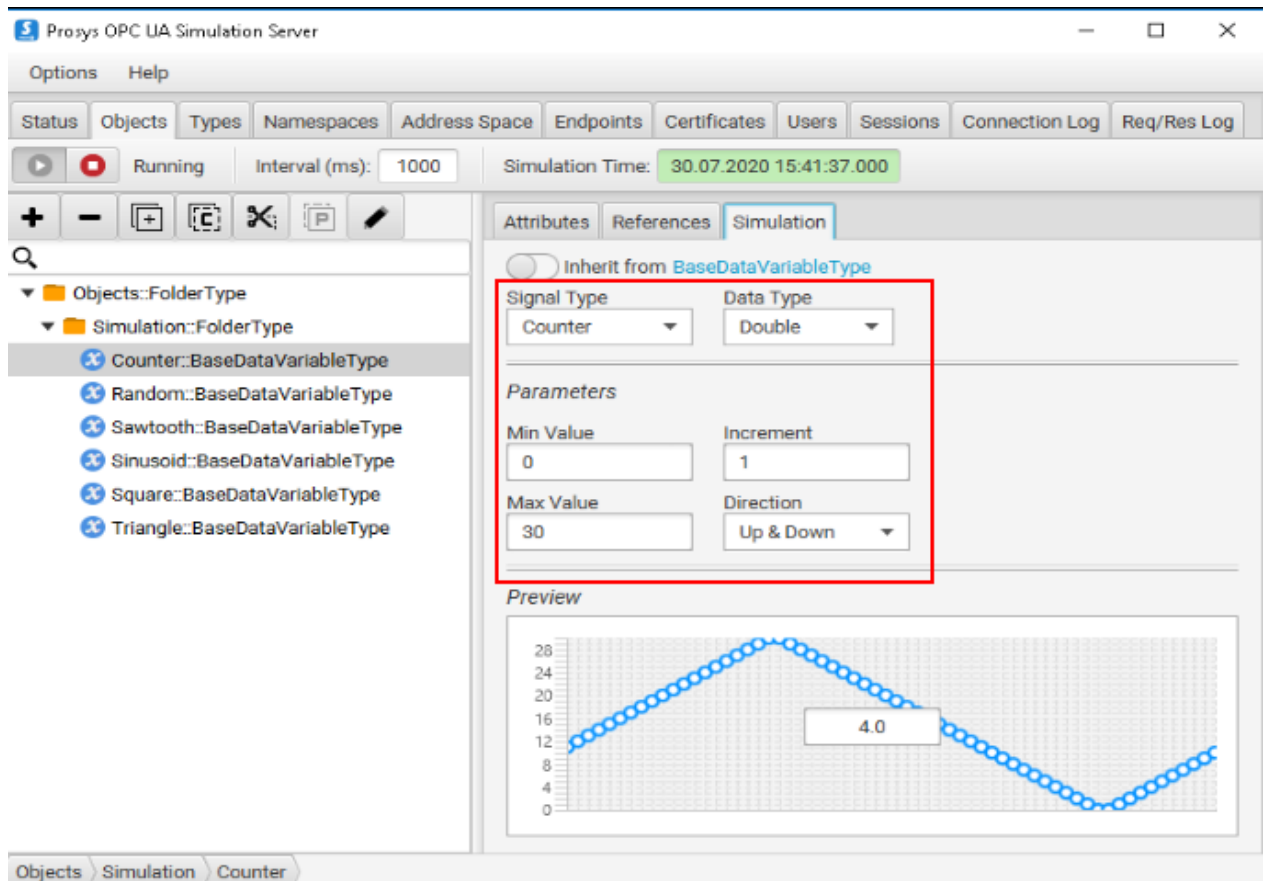
10.2. Simulación OPC UA

- Se descarga la aplicación Prosys OPC UA Simulation Server
- Clic en Objects, ahí se crean la cantidad de nodos que se vayan a simular



Fuente: Elaboración propia

- A cada nodo se le asigna el tipo de señal, tipo de dato, valores mínimo y máximo

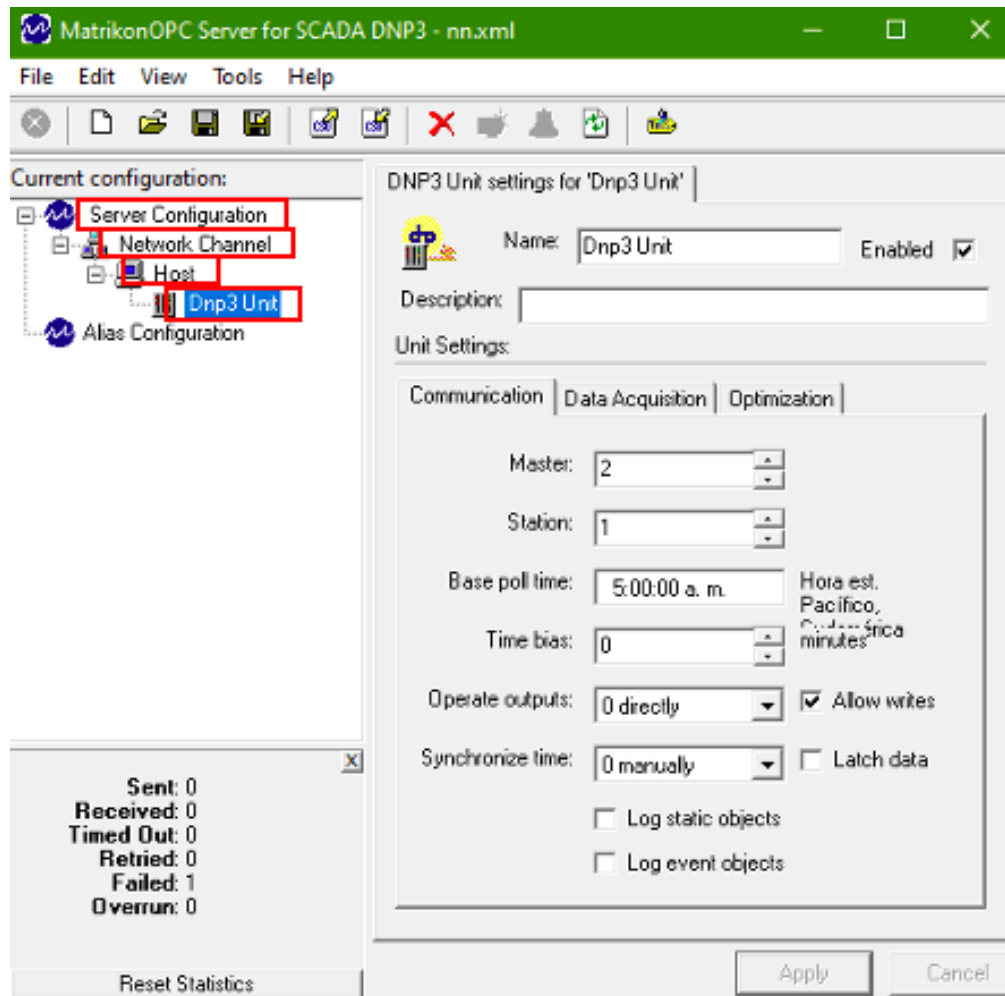


Fuente: Elaboración propia

- Clic en Run para ejecutar la simulación

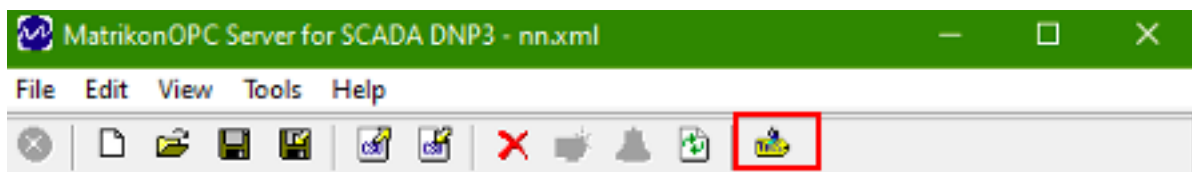
10.3. Simulación DNP3

- Descargar Matrikon OPC Server for SCADA DNP3
- Clic en Server Configuration, Network Channel, Host, Dnp3 Unit



Fuente: Elaboración propia

- Se asigna un nombre a la unidad de DNP3 creada y el número del maestro y la estación que se va a utilizar
- Una vez creada la unidad DNP3 se hace clic en el botón de las tags



Fuente: Elaboración propia

- Se crean las tags de las variables que se desean simular haciendo clic en agregar tag



Fuente: Elaboración propia

- Una vez creadas las tags se crea un grupo y se seleccionan las tags creadas para ser monitoreadas

The image shows two screenshots from an OPC UA simulation environment. The top screenshot displays a tree view of the 'Matrikon.OPC.Simulation.1' server. Under the 'Matrikon.OPC.Simulation.1' node, a folder named 'Group0' is highlighted with a red rectangle. The bottom screenshot shows the 'Contents of 'Group0'' table, which lists three data items: 'Random.Boolean', 'Random.DoubleFloat', and 'Random.Real8'. Each item has a value, quality, timestamp, and status.

Item ID	Access Path	Value	Quality	Timestamp	Status
Random.Boolean		False	Good, non-specific	10/15/202...	Active
Random.DoubleFloat		6915	Good, non-specific	10/15/202...	Active
Random.Real8		5917,018...	Good, non-specific	10/15/202...	Active

Fuente: Elaboración propia

10.4. Código fuente

```
const endpointUrl = "opc.tcp://LAPTOP-DGV3S6G0:53530/OPCUA/SimulationServer";
const nodeIdToMonitor = "ns=3;i=1010;s=Flow1";
const nodeIdToMonitor2 = "ns=3;i=1011;s=Flow2";
const nodeIdToMonitor3 = "ns=3;i=1007;s=Temp1";
const nodeIdToMonitor4 = "ns=3;i=1008;s=Temp2";
const nodeIdToMonitor5 = "ns=3;i=1012;s=Flow3";
const nodeIdToMonitor6 = "ns=3;i=1009;s=Pressure";
(async () => {
  try {
    const client = OPCUAClient.create({
      endpoint_must_exist: false
    });
    client.on("backoff", (retry, delay) => {
      console.log("Retrying to connect to ", endpointUrl, " attempt ", retry);
    });
    console.log("connecting to ", chalk.cyan(endpointUrl));
    await client.connect(endpointUrl);
    console.log("connected to ", chalk.cyan(endpointUrl));
  }
});
const itemToMonitor = {
```

```

    nodeId: nodeIdToMonitor,
    attributeId: AttributeIds.Value
  };

```

```

const itemToMonitor2 = {
  nodeId: nodeIdToMonitor2,
  attributeId: AttributeIds.Value
};

```

```

const itemToMonitor3 = {
  nodeId: nodeIdToMonitor3,
  attributeId: AttributeIds.Value
};

```

```

const itemToMonitor4 = {
  nodeId: nodeIdToMonitor4,
  attributeId: AttributeIds.Value
};

```

```

const itemToMonitor5 = {
  nodeId: nodeIdToMonitor5,
  attributeId: AttributeIds.Value
};

```

```

const itemToMonitor6 = {
  nodeId: nodeIdToMonitor6,
  attributeId: AttributeIds.Value
};

```

```

const parameters = {
  samplingInterval: 100,
  discardOldest: true,
  queueSize: 100
};

```

```

const monitoredItem = await subscription.monitor(itemToMonitor, parameters,
TimestampsToReturn.Both);
  const monitoredItem2 = await subscription.monitor(itemToMonitor2, parameters,
TimestampsToReturn.Both);
  const monitoredItem3 = await subscription.monitor(itemToMonitor3, parameters,
TimestampsToReturn.Both);
  const monitoredItem4 = await subscription.monitor(itemToMonitor4, parameters,
TimestampsToReturn.Both);
  const monitoredItem5 = await subscription.monitor(itemToMonitor5, parameters,
TimestampsToReturn.Both);
  const monitoredItem6 = await subscription.monitor(itemToMonitor6, parameters,
TimestampsToReturn.Both);

```

```

monitoredItem.on("changed", (dataValue) => {

```

```

        console.log("Flow 1 = " + dataValue.value.toString());
    monitoredItem2.on("changed", (dataValue) => {
        console.log("Flow 2 = " + dataValue.value.toString());
    });
    monitoredItem3.on("changed", (dataValue) => {
        console.log("Temp 1 = " + dataValue.value.toString());
    });
    monitoredItem4.on("changed", (dataValue) => {
        console.log("Temp 2 = " + dataValue.value.toString());
    });
    monitoredItem5.on("changed", (dataValue) => {
        console.log("Flow 3 = " + dataValue.value.toString());
    });
    monitoredItem6.on("changed", (dataValue) => {
        console.log("Pressure = " + dataValue.value.toString());
    });

    let Modbus = require("jsmodbus");
    let socketmb = new net.Socket();
    let client = new Modbus.client.TCP(socketmb);
    let options = {
        host: "127.0.0.1",
        port: 502,};

    client.readHoldingRegisters(0,4)
        .then(function(data) {
            console.log(data.response._body.valuesAsString());
            let variable = data.response._body.valuesAsString[0].toString();
            let variable2 = data.response._body.valuesAsString[1].toString();
            let variable3 = data.response._body.valuesAsString[2].toString();
            let variable4 = data.response._body.valuesAsString[3].toString();
        })
    }

    let insDat = {Valor: variable, Nombre:'Var 1'};
    let sql = 'INSERT INTO modbus SET ? ';
    let query = db.query(sql,insDat, (err,result) =>{
        if (err) console.log (err);
    });

    let instDat2 = {Valor: variable2 , Nombre:'Var 2'}
    let sql2 = 'INSERT INTO modbus SET ? ';
    let query2 = db.query(sql2,instDat2, (err,result) =>{
        if (err) console.log (err);
    });

    let insDat3 = {Valor: variable3, Nombre:'Var 3'};
    let sql3 = 'INSERT INTO modbus SET ? ';
    let query3 = db.query(sql3,insDat3, (err,result) =>{
        if (err) console.log (err);
    });

```

```
});
```

```
let insDat4 = {Valor: variable4, Nombre:'Var 4'};
  let sql4 = 'INSERT INTO modbus SET ? ';
  let query4 = db.query(sql4,insDat4, (err,result) =>{
    if (err) console.log (err);
  });
```

```
let flujo1 = {Valor: dataValue.value.value.toFixed(2), Nombre: 'Flow 1', Fecha:
dataValue.serverTimestamp };
  let sql = 'INSERT INTO opc SET ? ';
  let query1 = db.query(sql,flujo1, (err,result) =>{
    if (err) console.log (err);
  });
```

```
let flujo2 = {Valor: dataValue.value.value.toString(), Nombre: 'Flow 2', Fecha:
dataValue.serverTimestamp };
  let sql = 'INSERT INTO opc SET ?';
  let query = db.query(sql,flujo2, (err,result) =>{
    if (err) console.log (err);
  });
```

```
let temperatura1 = {Valor: dataValue.value.value.toFixed(2), Nombre: 'Temp 1', Fecha:
dataValue.serverTimestamp };
  let sql = 'INSERT INTO opc SET ?';
  let query = db.query(sql,temperatura1, (err,result) =>{
    if (err) console.log (err);
  });
```

```
let temperatura2 = {Valor: dataValue.value.value.toFixed(2), Nombre: 'Temp 2', Fecha:
dataValue.serverTimestamp };
  let sql = 'INSERT INTO opc SET ?';
  let query = db.query(sql,temperatura2, (err,result) =>{
    if (err) console.log (err);
  });
```

```
let flujo3 = {Valor: dataValue.value.value.toString(), Nombre: 'Flow 3', Fecha:
dataValue.serverTimestamp };
  let sql = 'INSERT INTO opc SET ?';
  let query = db.query(sql,flujo3, (err,result) =>{
    if (err) console.log (err);
  });
```



```

let presion = {Valor: dataValue.value.value.toFixed(2), Nombre: 'Pressure', Fecha:
dataValue.serverTimestamp };
  let sql = 'INSERT INTO opc SET ?';
  let query = db.query(sql,presion, (err,result) =>{
    if (err) console.log (err);
  });

```

```

let DNP1 = {Valor: dataValue.value.value.toFixed(0), Nombre: 'Var1' };
  let sql = 'INSERT INTO dnp3 SET ?';
  let query = db.query(sql,DNP1, (err,result) =>{
    if (err) console.log (err);
  });

```

```

let DNP2 = {Valor: dataValue.value.value.toFixed(0), Nombre: 'Var2' };
  let sql = 'INSERT INTO dnp3 SET ?';
  let query = db.query(sql,DNP2, (err,result) =>{
    if (err) console.log (err);
  });

```

```

<div class="row">
  <div class = "col-12 col-xl-12">
    <div class="col-12 col-xl-12 stretch-card">
      <div class="row flex-grow">
        <div class="col-md-12 grid-margin stretch-card">
          <div class="card">
            <div class="card-body">
              <div id="TextCov"></div>
              <div id="TextGranjita"></div>
              <div id="TextCaucasia"></div>
              <div id="TextChiquillo"></div>
              <div id="TextVasconia"></div>
              <div id="TextBelleza"></div>
              <div id="TextPaez"></div>
              <div id="TextPorvenir"></div>
              <div id="TextMiraflores"></div>
              <div id="TextCupiagua"></div>
              <div id="TextCusiana"></div>
              
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>

```

```

    </div>
  </div>
</div>

```

```

const TextCov = document.getElementById("TextCov");
const TextGranjita = document.getElementById("TextGranjita");
const TextCaucasia = document.getElementById("TextCaucasia ");

```

```

const socket = io.connect('http://localhost:3700');

```

```

socket.on('message', function (data){

```

```

  TextCov.innerHTML = data.value + "BPH" ;
  TextGranjita.innerHTML = data.value + "BPH" ;
  TextCaucasia.innerHTML = data.value + "BPH" ;
  TextChiquillo.innerHTML = data.value + "BPH" ;
  TextVasconia.innerHTML = data.value + "BPH" ;
  TextBelleza.innerHTML = data.value + "BPH" ;
  TextPaez.innerHTML = data.value + "BPH" ;
  TextPorvenir.innerHTML = data.value + "BPH" ;
  TextCopiagua.MirafloresHTML = data.value + "BPH" ;
  TextCusiana.innerHTML = data.value + "BPH" ;

```

```

body {
font-family: "Overpass", sans-serif;
font-size: 0.875rem;
font-weight: 400;
line-height: 1.5;
}

```

```

body {
display: block;
margin-top: 8px;
}

```

```

body {
margin: 0;
color: #000;
text-align: left;
background-color: #f9fafb;
height: 100%;
}

```

```
@-moz-keyframes bounce {  
  0% {  
    -webkit-transform: translateY(0);  
    -moz-transform: translateY(0);  
    -ms-transform: translateY(0);  
    -o-transform: translateY(0);  
    transform: translateY(0); }  
  20% {  
    -webkit-transform: translateY(0);  
    -moz-transform: translateY(0);  
    -ms-transform: translateY(0);  
    -o-transform: translateY(0);  
    transform: translateY(0); }  
  40% {  
    -webkit-transform: translateY(-30px);  
    -moz-transform: translateY(-30px);  
    -ms-transform: translateY(-30px);  
    -o-transform: translateY(-30px);  
    transform: translateY(-30px); }  
  50% {  
    -webkit-transform: translateY(0);  
    -moz-transform: translateY(0);  
    -ms-transform: translateY(0);  
    -o-transform: translateY(0);  
    transform: translateY(0); }  
  60% {  
    -webkit-transform: translateY(-15px);  
    -moz-transform: translateY(-15px);  
    -ms-transform: translateY(-15px);  
    -o-transform: translateY(-15px);  
    transform: translateY(-15px); }  
  80% {  
    -webkit-transform: translateY(0);  
    -moz-transform: translateY(0);  
    -ms-transform: translateY(0);  
    -o-transform: translateY(0);  
    transform: translateY(0); }  
  100% {  
    -webkit-transform: translateY(0);  
    -moz-transform: translateY(0);  
    -ms-transform: translateY(0);  
    -o-transform: translateY(0);  
    transform: translateY(0); } }
```

```
figure {  
  margin: 0 0 1rem; }
```

```
img {  
  vertical-align: middle;  
  border-style: none; }
```

```
svg {  
  overflow: hidden;  
  vertical-align: middle; }
```

```
table {  
  border-collapse: collapse; }
```