



INGENIERÍA EN CONTROL Y AUTOMATIZACIÓN
MANUAL PRACTICAS MAQUINA CONTEO

Presenta:

SALDARRIAGA HORTA DAVID ALEXANDER
PINZÓN MALAVER CRISTIAN DAVID

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLOGICA
BOGOTÁ D.C COLOMBIA
2021
TABLA DE CONTENIDO

Contenido

PC000. Reconocimiento de la maquina	10
1. Reconocimiento de la maquina	11
2. Reconocimiento del tablero eléctrico	12
3. Verificación de las fuentes de alimentación.....	13
PC001. Conexión entre Intouch-Codesys OPCUA.....	14
1. Crear configuración de Símbolos en Codesys.....	15
2. Configuración KEPServerEX con SOFTPLC.....	17
3. Crear grupo Acces Name en Intouch.....	21
PC002. Test de la planta.....	26
1. Verificación de sensores.....	28
2. Verificación de actuadores	29
PC003. Importar/Exportar SCADA.....	30
1. Como exportar un Programa de Intouch.....	30
2. Como importar un Programa de Intouch.....	33
3. Cambiar variables luego de importar.....	36
3.1 Método 1:.....	37
3.2 Método 2:	39
PC004. Crear una base de Datos SQL Server	43
1. Creación Base de datos en SSMS	45
2. Conexión ODBC	51
3. Intouch Crear BindList	54
4. Crear Script de Conexión	55
5. Comprobar el almacenamiento de datos On-Premise	57
PC005. Base de datos SQL on premise sincronizada con Azure a través de Data Sync	59
1. Requisitos del sistema.....	61
2. Crear Grupo de recursos y servidor SQL en Azure	62
3. Crear regla en Firewall y base de datos en Azure	67
4. Crear un grupo de Sincronización	72
5. Configurar Agente.....	75
6. Probar envío de datos desde base On-Premise a Azure.....	82
7. Posibles Fallos.....	83
8. Referencia	83
PC006. Identificación y diseño del controlador discreto en Matlab	84

1.	Creación señales de identificación y almacenamiento de Datos	86
1.1.	Señal paso	87
1.2.	Señal paso cuadrada	88
1.3.	Señal paso modificada	89
1.4.	Señal pseudoaleatoria.....	90
1.5.	Adquisición de datos	92
2.	Diseño de la estructura Continua.....	95
2.1.	Cálculo de parámetros	101
2.2.	Validación experimental	103
3.	Diseño de la estructura Discreta.....	104
4.	Selección del mejor modelo (¿Continuo o Discreto?).....	106
5.	Diseño del controlador.....	108
5.1.	SISOTOOL.....	108
6.	Validación del controlador (Simulink).....	119
PC007. Comparación entre controlador PID y Bloque PID en Codesys	121
1.	Creación del bloque PID en Codesys	122
1.1.	Bloque en Codesys y creaciones variables	122
1.2.	Rutina de conversiones	123
1.3.	Sintonización PID (Ejemplo)	124
2.	Creación del controlador discreto PID en Codesys	125
3.	Programación proceso en Codesys.....	125
4.	Comparación intuitiva entre ambos PIDs.....	128
Anexos	131
1.	Recopilaciones variables Intouch/Codesys[PC003].....	131
2.	Código creación Tabla por SQL[PC004].....	137
3.	Código Data Change Script en Intouch[PC004]	138
4.	Código señales de identificación en Codesys[PC006].....	139
5.	Pseudocódigo Controlador PID[PC006]	142
6.	Conversiones previas al desarrollo del PID [PC007]	144
7.	Código Regulación en Codesys [PC007].....	145

LISTA DE FIGURAS

Figura 1: Esquema básico de la actividad de reconocimiento. [Autores]	10
Figura 2: Diagrama P&ID de la máquina de conteo. [Autores]	11
Figura 3: Esquema de conexión de alimentación en el tablero eléctrico. [Autores]	12
Figura 4: Fuentes de alimentación de la máquina de conteo. [Autores]	13
Figura 5: Borneras de alimentación del tablero eléctrico. [Autores]	13
Figura 6. KEPServerEX	14
Figura 7. Esquema básico de la actividad a realizar [Autores]	15
Figura 8. Agregar un objeto [Autores]	15
Figura 9. Configuración de símbolos [Autores]	16
Figura 10. Seleccionar variables [Autores]	16
Figura 11. KepServerEX [Autores]	17
Figura 12. Agregar canal en KepServerEX [Autores]	17
Figura 13. Canal Codesys. [Autores]	18
Figura 14. Agregar dispositivo. [Autores]	18
Figura 15. Device Discovery. [Autores]	19
Figura 16. Buscar dispositivo. [Autores]	19
Figura 17. Seleccionar Raspberry. [Autores]	20
Figura 18. Propiedades dispositivo. [Autores]	20
Figura 19. Importar Tags [Autores]	21
Figura 20. Agregar los tags de la configuración de símbolos. [Autores]	21
Figura 21. Quick Client. [Autores]	21
Figura 22. Crear Alias. [Autores]	22
Figura 23. Propiedades generales del alias. [Autores]	22
Figura 24. Mapeo de los tags importados [Autores]	23
Figura 25. Configuración final del alias. [Autores]	23
Figura 26. Intouch.[Autores]	23
Figura 27. Crear Access Names. [Autores]	24
Figura 28. Configuración del Acces Name. [Autores]	24
Figura 29. KepServerEX propiedades del proyecto. [Autores]	25
Figura 30. Propiedades. [Autores]	25
Figura 31. Obtener el application name. [Autores]	25
Figura 32: Esquema básico de la actividad de testeo. [Autores]	26
Figura 33: Verificación de la comunicación entre el SoftPLC y el SCADA. [Autores]	27
Figura 34: Selección del panel de control para el proceso de testeo. [Autores]	27
Figura 35: Habilitación del modo testeo en el panel de control del SCADA. [Autores]	28
Figura 36: Verificación del estado de sensores. Los rojos son on/off. El Verde es variable. [Autores]	28
Figura 37: Verificación del estado de los actuadores. Los rojos son on/off. Los verdes son regulables. [Autores]	29
Figura 38. SCADA que deseamos exportar. [Autores]	31
Figura 39. Export Windows. [Autores]	31
Figura 40. Error por ventanas abiertas [Autores]	32
Figura 41. Buscar destino del archivo a exportar. [Autores]	32
Figura 42. Seleccionar las ventanas a exportar. [Autores]	33
Figura 43. Buscar el destino de la carpeta. [Autores]	33
Figura 44. Creación de un nuevo SCADA. [Autores]	34

Figura 45. Importar SCADA. [Autores]	34
Figura 46. Buscar la carpeta del SCADA a importar. [Autores]	35
Figura 47. Opciones de importación. [Autores]	35
Figura 48. Importar las ventanas. [Autores]	36
Figura 49. Importar. [Autores]	36
Figura 50. Modificar Variables. [Autores]	37
Figura 51. Variable Total_Bot. [Autores]	37
Figura 52. Variable Total_Bot Método 1. [Autores]	38
Figura 53. Variable Total_Bot Método 1, usar tabla recopilación. [Autores]	38
Figura 54. Variable Total_Bot Método 1 definir el Acces Name. [Autores]	38
Figura 55. Variable Total_Bot Metodo 1 configurar Acces Name. [Autores]	39
Figura 56. Modificar variables con Metodo 2. [Autores]	39
Figura 57. Método 2, seleccionar todo. [Autores]	40
Figura 58. Método 2, Opción sustituir tags. [Autores]	40
Figura 59. Método 2, sustituir Tags. [Autores]	41
Figura 60. Método 2, variable Reset. [Autores]	41
Figura 61. Metodo 2, variable Reset usar tabla recopilación. [Autores]	41
Figura 62. Método 2, variable Reset modificación. [Autores]	42
Figura 63. Esquema básico de la actividad a realizar. [Autores]	43
Figura 64. Diferentes orígenes de datos en analítica y ciencia de Datos.....	44
Figura 65. Sql server configuration Manager. [Autores]	45
Figura 66. Conectarse al servidor (local). [Autores]	45
Figura 67. Crear Database. [Autores]	46
Figura 68. Crear un Nuevo Login. [Autores]	47
Figura 69. Password and Login name. [Autores]	47
Figura 70. Pestaña User Mapping. [Autores]	48
Figura 71. Tabla del proceso (completo). [Autores]	49
Figura 72. Configuración llave primaria. [Autores]	50
Figura 73. Hacer una query usando el asistente. [Autores]	50
Figura 74. Comprobar la creación de la tabla DatasetProceso. [Autores]	51
Figura 75. Abrir ODBC Data Sources. [Autores]	51
Figura 76. DSN Sistema	52
Figura 77. Agregar conexión SQL Server. [Autores]	52
Figura 78. Nombre y autenticación. [Autores]	53
Figura 79. Seleccionar Database. [Autores]	53
Figura 80. Probar ODBC. [Autores]	53
Figura 81. SCADA Intouch. [Autores]	54
Figura 82. BindList. [Autores]	54
Figura 83. BindList tagname Lotes. [Autores]	55
Figura 84. Script Data change. [Autores]	55
Figura 85. Interfaz conexión SQL. [Autores]	57
Figura 86. Runtime en Intouch. [Autores]	57
Figura 87. Registro de datos on premise con SQL Server. [Autores]	58
Figura 88. Esquema básico de la actividad a realizar. [Autores]	59
Figura 89. Algunas de las nubes tradicionales. [Autores]	60
Figura 90. SQL vs NoSQL. [Autores]	60
Figura 91. Formato JSON. [Autores]	61
Figura 92. Azure Portal. [Autores]	62

Figura 93. Crear Grupo de recursos. [Autores]	62
Figura 94. Configuración Grupo de Recursos. [Autores]	63
Figura 95. Creacion del Grupo de Recursos. [Autores]	63
Figura 96. Buscar Servicio SQL Server. [Autores]	64
Figura 97. Crear servidor en Azure [Autores]	64
Figura 98. Configuración Server. [Autores]	64
Figura 99. Servidor creado. [Autores]	65
Figura 100. Conectar el servidor de Azure, parte 1. [Autores]	65
Figura 101. Conectar el servidor de Azure, parte 2. [Autores]	66
Figura 102. Creación de la regla en Azure. [Autores]	66
Figura 103. Creación de la regla “RuleHome1”. [Autores]	67
Figura 104. Acceder desde SMSS al servidor en Azure. [Autores]	67
Figura 105. Win+R para abrir la venta	68
Figura 106. Regla de salida. [Autores]	68
Figura 107. Regla de salida, paso 1. [Autores]	68
Figura 108. Regla de salida, paso 2. [Autores]	69
Figura 109. Buscar servicio SQL Database. [Autores]	69
Figura 110. Crear base de datos sqlazuredb. [Autores]	69
Figura 111. Cambiar opciones de almacenamiento DB. [Autores]	70
Figura 112. Plan Básico para bases de datos. [Autores]	70
Figura 113. Configuración resumen sqlazuredb. [Autores]	70
Figura 114. Implementación base de datos en curso. [Autores]	71
Figura 115. Conectar servidor local. [Autores]	71
Figura 116. Refrescar servidor Azure. [Autores]	71
Figura 117. Sqlazuredb propiedades [Autores]	72
Figura 118. Crear grupo de sincronización. [Autores]	72
Figura 119. Crear grupo de sincronización parte 1. [Autores]	73
Figura 120. Creación grupo de sincronización parte 1. [Autores]	73
Figura 121. Configuración grupo de sincronización. [Autores]	74
Figura 122. Primera base de datos asignada. [Autores]	74
Figura 123. Configuración previa. [Autores]	75
Figura 124. Configuración On-premise. [Autores]	75
Figura 125. Descargar SQL Data Sync. [Autores]	75
Figura 126. Instalar Microsoft SQL Data Sync Agent 2.0. [Autores]	76
Figura 127. ¿Quién soy yo?. [Autores]	76
Figura 128. Instalación exitosa de Microsoft SQL Data Sync Agent. [Autores]	77
Figura 129. Crear y generar clave. [Autores]	77
Figura 130. Abrir Microsoft SQL Data Sync Agent 2.0. [Autores]	77
Figura 131. Microsoft SQL Data Sync Agent 2.0 configurar llave. [Autores]	78
Figura 132. Registrar la Base de datos On-Premise. [Autores]	78
Figura 133. Se registro correctamente la base de datos OnPremise. [Autores]	79
Figura 134. Verificar estado del agente en Azure. [Autores]	79
Figura 135. Seleccionar Base de datos On-premise. [Autores]	80
Figura 136. Comprobar miembros de sincronización (On-premise y Base de datos Azure). [Autores]	80
Figura 137. Seleccionar Base de datos On-premise. [Autores]	80
Figura 138. Sincronizar Grupo. [Autores]	81
Figura 139. Sincronización exitosa. [Autores]	81

Figura 140. Editor de consultas en Azure. [Autores].....	82
Figura 141. Ingrese con sus credenciales. [Autores]	82
Figura 142. Haga una consulta básica para ver los registros. [Autores]	82
Figura 143. Esquema básico de la actividad a realizar. [Autores]	84
Figura 144. Metodología a seguir.	85
Figura 145. Interfaz Señales de identificación en Codesys.[Autores]	86
Figura 146. Código en Lenguaje estructurado señal paso. [Autores]	87
Figura 147. Código en Ladder para señal paso. [Autores].....	87
Figura 148. Visualización señal paso en Codesys. [Autores]	88
Figura 149. Código en Lenguaje estructurado señal paso cuadrada. [Autores]	88
Figura 150.Código en Ladder para señal paso cuadrada. [Autores].....	89
Figura 151. Temporizador en Ladder. [Autores]	89
Figura 152. Visualización señal paso cuadrada en Codesys. [Autores]	89
Figura 153. Código en Lenguaje estructurado señal paso modificada. [Autores]	90
Figura 154. Código en Ladder para señal paso modificada. [Autores]	90
Figura 155. Visualización señal paso modificada en Codesys. [Autores]	90
Figura 156. Código en Lenguaje estructurado señal pseudoaleatoria. [Autores].....	91
Figura 157. Código en Ladder para señal pseudoaleatoria. [Autores]	91
Figura 158. Visualización señal pseudoaleatoria en Codesys. [Autores].....	92
Figura 159. Bloques CFC para almacenamiento datos formato csv. [Autores].....	92
Figura 160. Definir los tiempos para recolectar datos. [Autores]	93
Figura 161. Ruta del csv en la Raspberry. [Autores]	93
Figura 162. Command Window de MATLAB. [Autores]	95
Figura 163. System Identification MATLAB.[Autores]	96
Figura 164. Datos en el dominio del tiempo en IDENT. [Autores].....	96
Figura 165. Importar Datos en Matlab. [Autores]	96
Figura 166. Filtrar Datos a usar. [Autores]	97
Figura 167. Atributos exportados al Workspace de MATLAB. [Autores].....	97
Figura 168. Configurar la señal en IDENT. [Autores].....	98
Figura 169. Observar la salida y entrada. [Autores].....	98
Figura 170. Visualización entrada/salida. [Autores]	99
Figura 171. Seleccionar Rangos de trabajo. [Autores]	99
Figura 172. Resultado señal filtrada. [Autores]	100
Figura 173. Interfaz IDENT. [Autores].....	100
Figura 174. Trabajar con la señal deseada. [Autores]	101
Figura 175. Process Models para calcular parámetros. [Autores]	102
Figura 176. Obtención del modelo continuo	102
Figura 177. Visualización de los parámetros en el modelo. [Autores]	103
Figura 178. Validación del modelo en continuo. [Autores]	104
Figura 179. Diseñar Estructura discreta. [Autores]	105
Figura 180. Configurar los parámetros. [Autores]	105
Figura 181. Resumen de la estructura creada. [Autores].....	106
Figura 182. Visualización de las dos estructuras creadas. [Autores]	106
Figura 183. Porcentaje de estimación en modelos. [Autores].....	106
Figura 184. Creación de distintos modelos. [Autores]	107
Figura 185. Distintos porcentajes de estimación. [Autores]	107
Figura 186. Orden del Modelo ARX creado. [Autores].....	108
Figura 187. Exportar el Modelo creado en tiempo continuo. [Autores]	109

Figura 188. Visualizar el modelo exportado en Command Window. [Autores]	109
Figura 189. Líneas previas a SISOTool. [Autores]	110
Figura 190. Discretizar el modelo continuo antes de SISOTool. [Autores]	110
Figura 191. Toolbox de SISOTool. [Autores]	111
Figura 192. Seleccionar estructura. [Autores]	111
Figura 193. Crear requerimientos para el diseño del controlador. [Autores]	112
Figura 194. Requerimientos del diseñador. [Autores]	112
Figura 195. Configurar la visualización de la señal. [Autores]	113
Figura 196. Espacio de trabajo para Root-Locus. [Autores]	113
Figura 197. Diseño del controlador. [Autores]	114
Figura 198. Diseño del controlador agregar polo real. [Autores]	114
Figura 199. Configuración final del controlador diseñado. [Autores]	115
Figura 200. Root Locus controlador. [Autores]	115
Figura 201. Pseudocódigo parte 1. [Autores]	118
Figura 202. Pseudocódigo parte 2. [Autores]	119
Figura 203. Validación del controlador creado en Simulink. [Autores]	120
Figura 204. Visualización de la respuesta del controlador	120
Figura 205. Esquema básico de la actividad a realizar. [Autores]	121
Figura 206. Bloque de función del PID en Codesys. [Autores]	122
Figura 207. Variables globales para el Bloque PID. [Autores]	123
Figura 208. Código Conversiones PID. [Autores]	124
Figura 209. Ejemplo de sintonización para el controlador PID, verde: Entrada RPM, azul: Salida RPM y rojo: Valor de referencia. [Autores]	125
Figura 210. Constante proporcional, integral y derivativa definida para el controlador PID. [Autores]	125
Figura 211. Graficet del proceso general. [Autores]	126
Figura 212. Código Regulación por lotes para PID. [Autores]	127
Figura 213. Código Regulación por distancia para PID	127
Figura 214. Graficet acciones secundarias. [Autores]	128
Figura 215. Sensor Inteligente comunicación MQTT. [Autores]	128
Figura 216. HMI Configuración PID en Intouch. [Autores]	129
Figura 217. Bloque PID por producción, carga	130
Figura 218. Controlador PID por producción, carga.	130
Figura 219. Bloque PID por producción, descarga.	130
Figura 220. Controlador PID por producción, descarga.	130
Figura 221. Bloque PID por distancia.	130
Figura 222. Controlador PID por distancia.	130
Figura 223. Conversiones Entrada/Salida ADC. [Autores]	130

LISTA DE TABLAS

Tabla 1: Planilla de reconocimiento. [Autores]	12
Tabla 2: Verificación del funcionamiento de los sensores. [Autores]	29
Tabla 3: verificación del funcionamiento de los actuadores. [Autores]	29
Tabla 4. Variables Data Change Script/Otras. [Autores]	56
Tabla 5. Configurar Data Change. [Autores]	56
Tabla 6. Datos tomados para una de las señales. [Autores].....	93
Tabla 7. Compilación Graficas obtenidas. [Autores]	94
Tabla 8. Asignación de memorias. [Autores]	117
Tabla 9. Tipos de regulación para aplicar el PID. [Autores]	122
Tabla 10. Parte de la estructura del programa. [Autores]	125
Tabla 11. Comparación entre Bloque PID y Controlador PID	129

	UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo ACTIVIDAD: Reconocimiento de la maquina	
Código: PC000 <input type="checkbox"/> ONLINE <input checked="" type="checkbox"/> OFFLINE		Duración: ---

PC000. Reconocimiento de la maquina

Objetivo de la práctica:

Entender la estructura y la composición de la maquina mediante un proceso de identificación y reconocimiento de elementos donde se evalúa la composición y las aplicaciones de cada una de las etapas que constituyen el sistema, para así establecer las condiciones de uso y los aspectos de su funcionamiento.

Material Necesario y requisitos para el desarrollo:

- Multímetro
- Computador
- Programas de diseño de esquema o diagramas

Esquema Grafico de la Actividad:



Figura 1: Esquema básico de la actividad de reconocimiento. [Autores]

Requisitos previos:

- Conocer el manual de funcionamiento de la maquina
- Conceptos básicos de circuitos

Resumen:

En la siguiente actividad se realizará la identificación de cada etapa de la maquina junto con cada lazo de control que la compone, para esto el estudiante deberá emplear sus conocimientos en circuitos y comprobar cada una de las conexiones que estructuran el sistema, de igual forma se debe ubicar la posición de cada elementos y describir su funcionamiento y su aplicación dentro del ciclo de producción.

Desarrollo de la Actividad:

En la práctica se debe realizar el reconocimiento de los diferentes dispositivos de la máquina y evaluar cada una de las conexiones.

1. Reconocimiento de la maquina

Lo primero es, mediante el análisis del siguiente diagrama P&ID de la llenar la planilla de reconocimiento

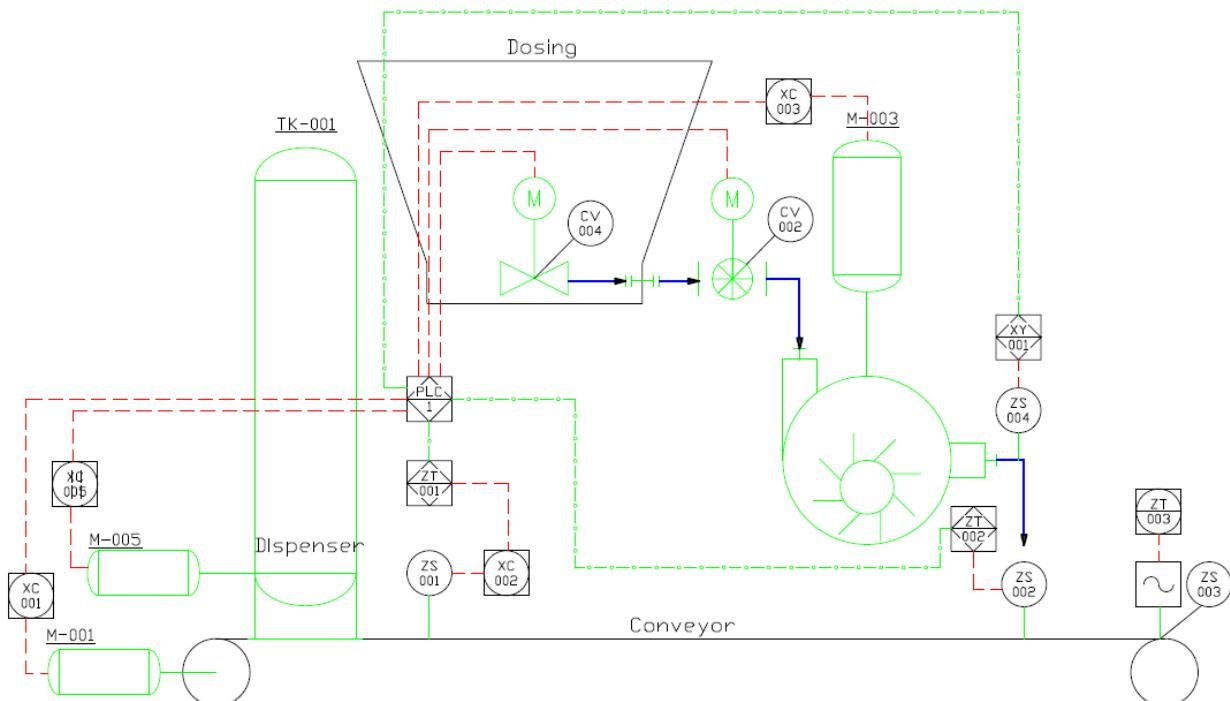


Figura 2: Diagrama P&ID de la máquina de conteo. [Autores]

Con la imagen anterior podemos determinar la ubicación de cada uno de los elementos puestos en campo, de esta forma es posible tener una idea general de cada uno de los circuitos, sensores, actuadores, elementos mecánicos y electromecánicos que hay en la máquina. En la siguiente planilla se deben llenar cada uno de los apartados de acuerdo a la estructura propuesta:

Nombre del elemento	Descripción de funcionamiento	Tipo de elemento y etapa de producción	Representación
Sensor Banda	Se describe de forma resumida la aplicación y el funcionamiento que tienen el elemento en la maquina	Los tipos de elementos que hay son: <ul style="list-style-type: none"> • Alimentación • Sensores • Actuadores • Estructurales • Mecánicos • Electromecánico • Circuito de acondicionamiento 	Foto del elemento descrito

		<p>Los tipos de etapas que componen el sistema son:</p> <ul style="list-style-type: none"> • Dispensado • Centrifugado • Dosificado • Producido 	
Banda Transportadora	<p>Este mecanismo se encarga de transportar las botellas desde el dispensador hasta la centrifuga, permite que la producción se realice en cadena y permite optimizar los procesos de envasado de capsulas.</p>	<p>Tipo</p> <ul style="list-style-type: none"> • Estructura mecánica <p>Etapa</p> <ul style="list-style-type: none"> • Forma parte de todas las etapas de producción 	

Tabla 1: Planilla de reconocimiento. [Autores]

2. Reconocimiento del tablero eléctrico

Luego de llenar la tabla anterior y conocer los elementos de la maquina es necesario realizar un diagrama que permite verificar las conexiones del tablero eléctrico con la máquina. Es decir se debe diseñar un esquema que indique cuales son las conexiones entre el soft PLC y los dispositivos descritos en la planta. De igual forma se debe realizar esto con las conexiones de alimentación. En la siguiente imagen se realiza un ejemplo de esto:

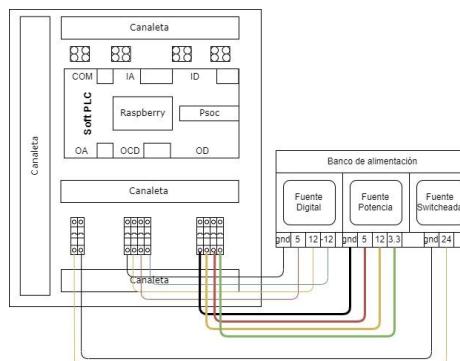


Figura 3: Esquema de conexión de alimentación en el tablero eléctrico. [Autores]

Para realizar este proceso se puede emplear el multímetro en continuidad y algún programa de diseño de esquemas como EdrawMax o Drawio, de esta forma se puede verificar fácilmente las conexiones del tablero y los circuitos, cabe aclarar que no a todos los elementos se les

puede medir la continuidad por lo tanto se debe revisar el manual de funcionamiento de la máquina para conocer la conexión del elemento respectivo.

3. Verificación de las fuentes de alimentación

Por último, para finalizar la práctica se deben comprobar las fuentes de alimentación.



Figura 4: Fuentes de alimentación de la máquina de conteo. [Autores]

El sistema maneja tres fuentes de alimentación y un adaptador, tal como se ve en la figura anterior, cada uno de ellas se debe conectar a una toma y con el multímetro se debe comprobar si el voltaje que se suministra al tablero eléctrico es el correcto.



Figura 5: Borneras de alimentación del tablero eléctrico. [Autores]

En la Figura 5 se ve la sección de fuentes del tablero, se debe comprobar que los niveles de voltaje son los que se indican y que la continuidad entre la fuente digital y la fuente swicheada existe y entre la fuente digital y la de potencia sea nula.

Como aspecto adicional, se puede comprobar el funcionamiento del adaptador de 5v que se emplea para la alimentación del servomotor, este puede verificar su consumo con una pinza amperimétrica o seguir su conexión y ver si el servo tiene el suministro de energía correspondiente.

	UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo ACTIVIDAD: Conexión entre Intouch-Codesys OPCUA	
Código: PC001	<input checked="" type="checkbox"/> ONLINE <input type="checkbox"/> OFFLINE	Duración: ---

PC001. Conexión entre Intouch-Codesys OPCUA

Objetivo de la práctica:

Aplicar el protocolo OPC UA el cual simplifica la comunicación de maquina a máquina (M2M), su versatilidad en la industria ha generado un gran uso del mismo por su versatilidad y eficiencia. Protocolos como estos aportan distintas alternativas a los estudiantes de Ingeniería en control y automatización que desean desarrollar aplicaciones específicamente en el área de la Industria 4.0 y el IoT.

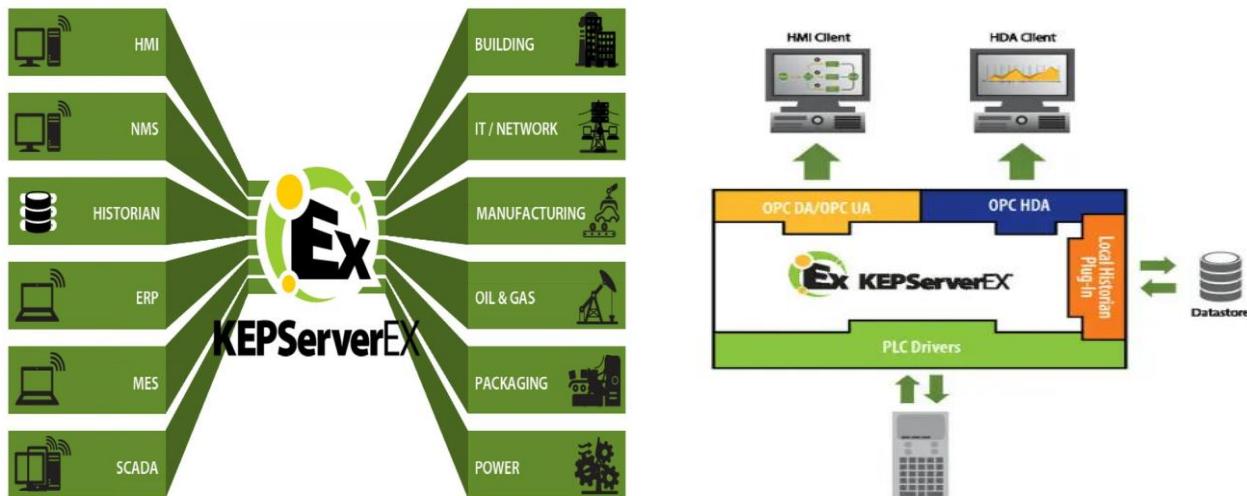


Figura 6. KEPServerEX

Material Necesario y requisitos para el desarrollo:

- Codesys V3.5 SP16 o superior
- Intouch Versión 10.0
- KEPServerEX 6

Esquema Grafico de la Actividad:



Figura 7. Esquema básico de la actividad a realizar [Autores]

Requisitos previos:

-Reconocimiento Maquina de Conteo y SoftPLC

Resumen:

En Codesys se deben crear las variables que van a ser enviadas por OPC UA al SCADA Intouch, esto se hace por medio de una “Configuración de Símbolos”, en KepserverEX crearemos un nuevo canal con el dispositivo de la Raspberry para así asociar las variables que envió de Codesys a Kepserverex. Luego de tener una conexión exitosa entre estos dos programas, se usará Intouch para crear las interfaces del proyecto, en Intouch se crea un Access Name y se asocia KEPServerEX, con este grupo ya garantizamos que las variables de Codesys se envíen síncronamente al SCADA en Intouch.

Desarrollo de la Actividad:

1. Crear configuración de Símbolos en Codesys

En primer lugar se debe crear la configuración de símbolos en Codesys, observe la siguiente Figura 8 donde se muestran los pasos para la creación.

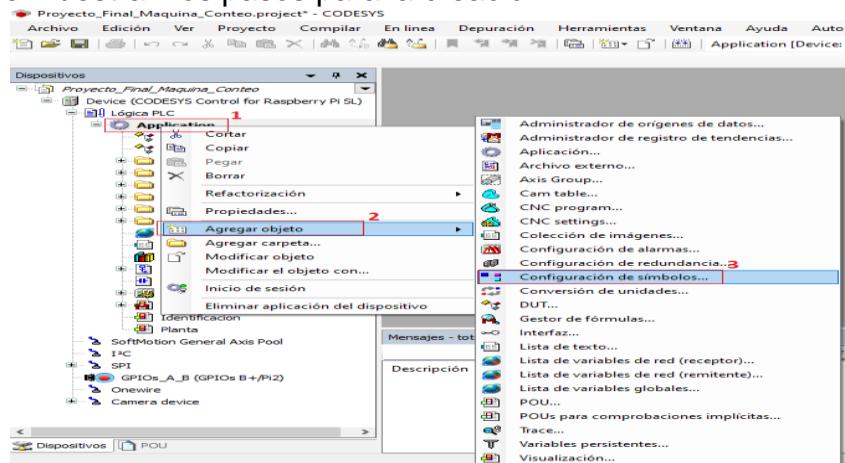


Figura 8. Agregar un objeto [Autores]

Escriba un nombre para su configuración de símbolos, recomendamos que las variables que asociamos a esta configuración de símbolos sean variables globales.

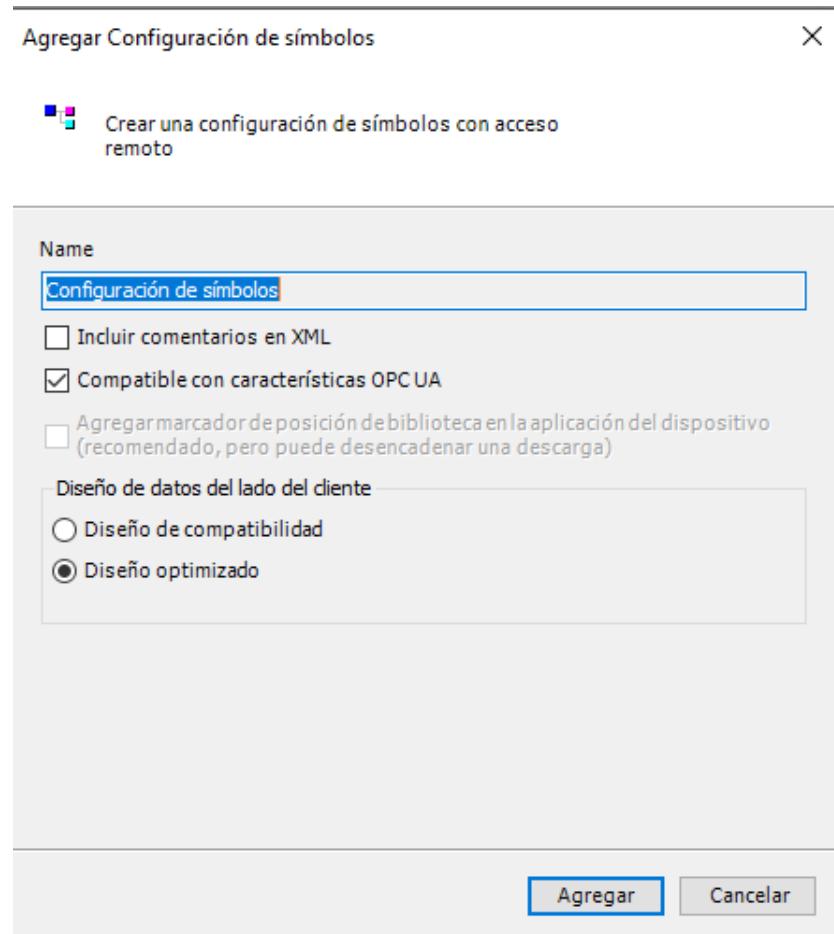


Figura 9. Configuración de símbolos [Autores]

Como se muestra en la figura 5, debe oprimir la opción “Crear” y luego seleccione todas las variables que desea comunicar por OPC UA con el SCADA (Intouch).

Símbolos	Derechos de acceso	Máximo	Attributo	Tipo	Variables de miembro	Comentario
Acciones_Secundarias						
Constants						
ExceptionFlags						
GVL						
IoConfig_Globals						
MQTT_comunication						
Programa_Central						
SPI_Principal						
Test_Alternador						
Test_Contador						
Al_01						
AO_01						
Amarillo						
B_Paso_Cuadrada						
B_Paso_Modificada						
B_Pseudoselector						
Bet_A1						
Bet_AO						
Botella						
Conectado						
Distancia						
Escrir_Data						
Etapa						
Guardar						
Kd						
Ki						
Kp						

Figura 10. Seleccionar variables [Autores]

Una aclaración importante es que no puede crear la configuración de símbolos si su programa tiene errores en la compilación, por tal motivo usualmente la configuración de Símbolos se realiza al culminar la programación, aunque la configuración de símbolos trabaja en cualquier momento, su creación debe hacerse cuando no hay errores.

2. Configuración KEPServerEX con SOFTPLC

En primer lugar ejecuta el administrador para arrancar algunas configuraciones del programa en segundo plano, luego abra la aplicación “KEPServerEX Configuration” en el cual crearemos la comunicación OPC UA entre los clientes y el servidor

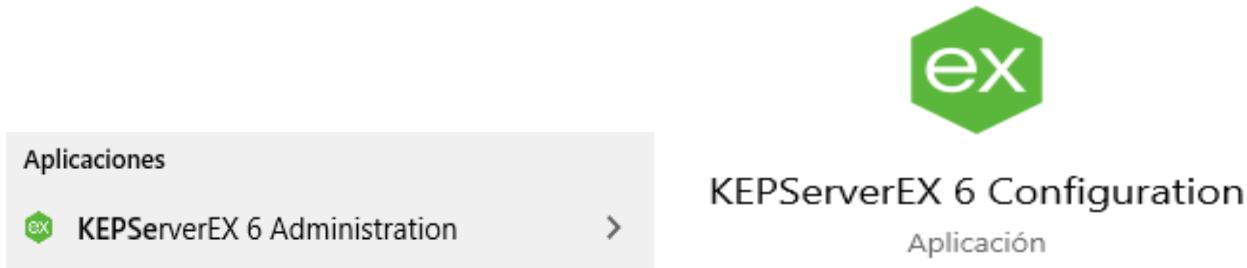


Figura 11. KepServerEX [Autores]

A continuación, asegúrese que **el proyecto en Codesys este en línea y ejecutándose**, debe agregar un nuevo canal en la raíz del Proyecto ver Figura 12, es necesario que seleccione CODESYS en la configuración del canal y en el resto de opciones déjelas como están configuradas por defecto.

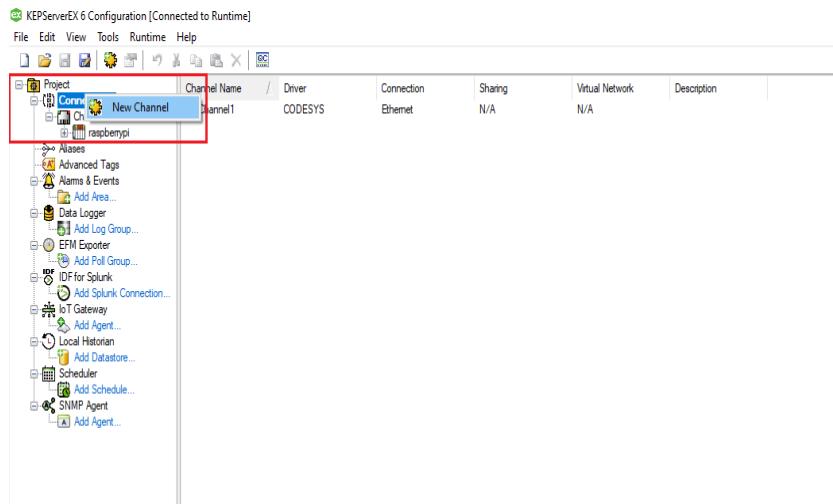


Figura 12. Agregar canal en KepServerEX [Autores]

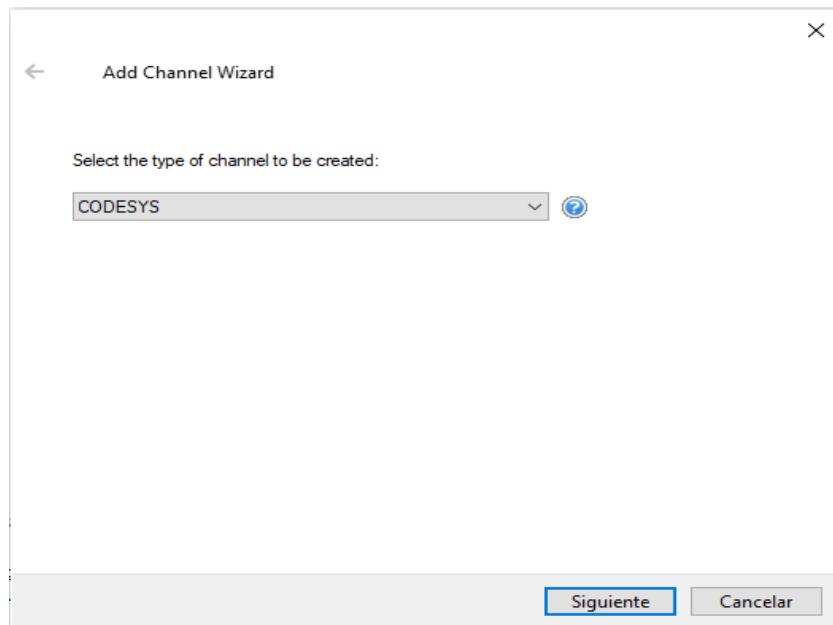


Figura 13. Canal Codesys. [Autores]

Luego de crear el canal, debe seleccionarlo y escoger la opción “Device Discovery”

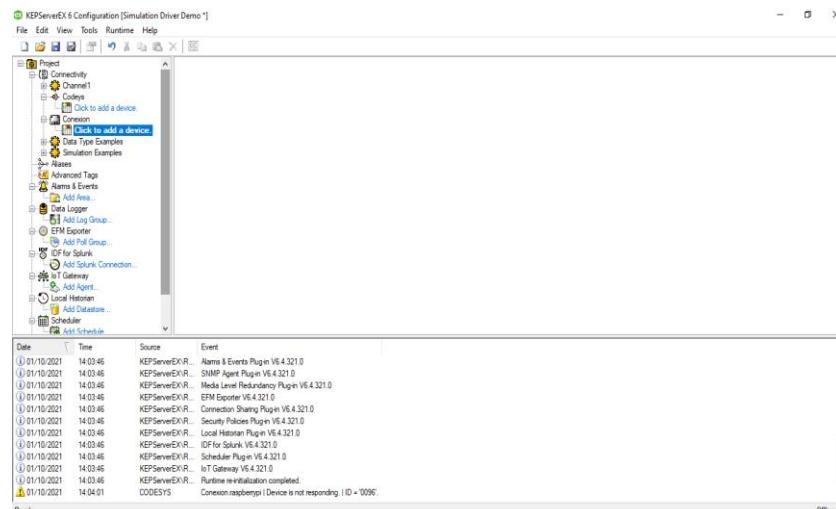


Figura 14. Agregar dispositivo. [Autores]

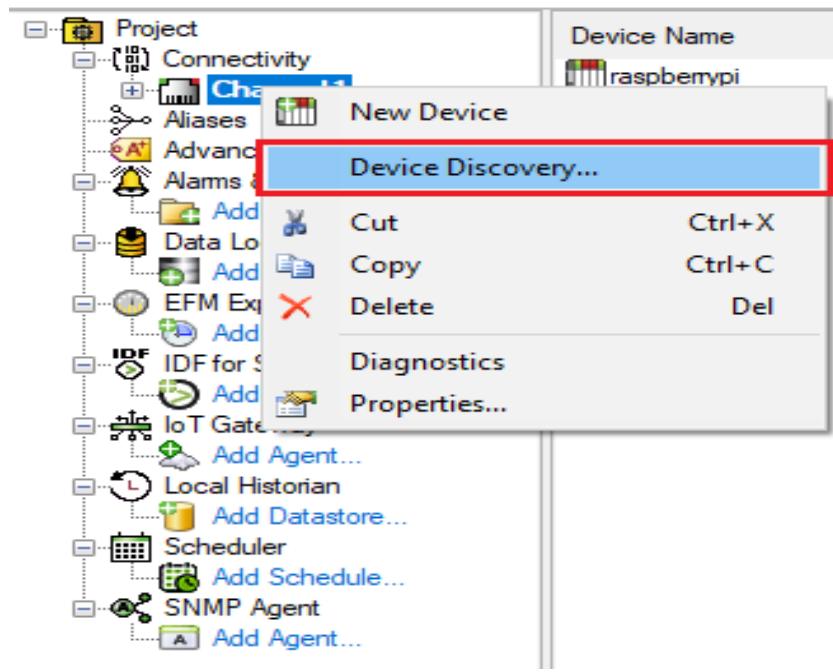


Figura 15. Device Discovery. [Autores]

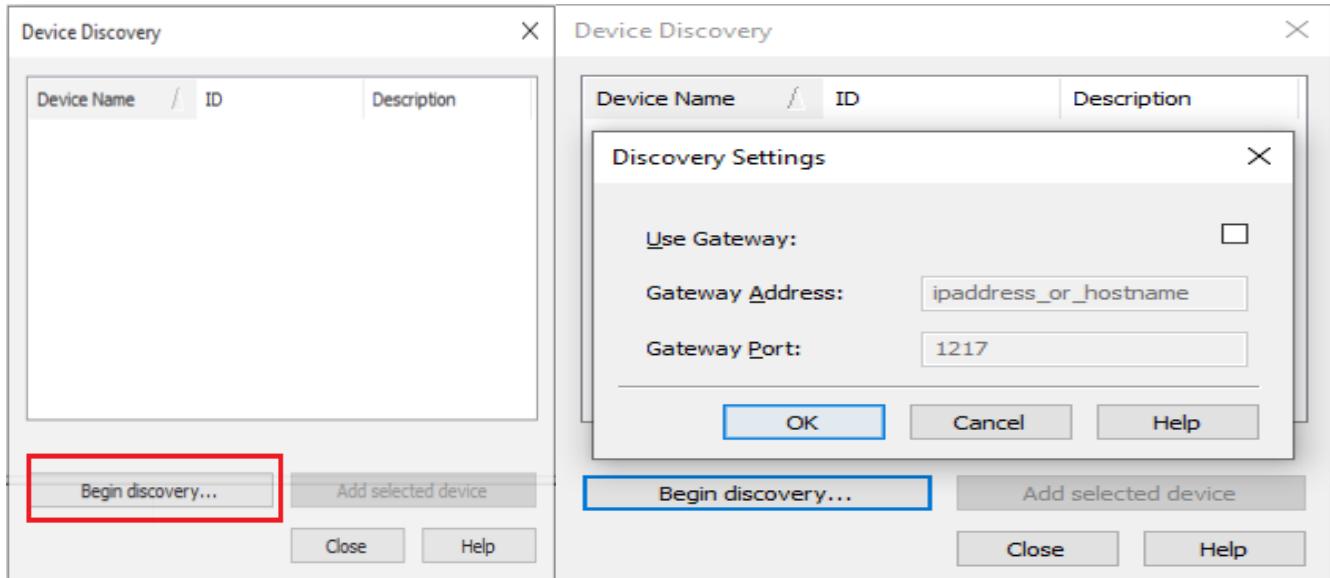


Figura 16. Buscar dispositivo. [Autores]

Observe que en la figura anterior se debe comenzar la búsqueda del dispositivo y automáticamente debe aparecer nuestro SOFTPLC, tenga en cuenta que **el proyecto en Codesys debe estar en línea y ejecutándose**

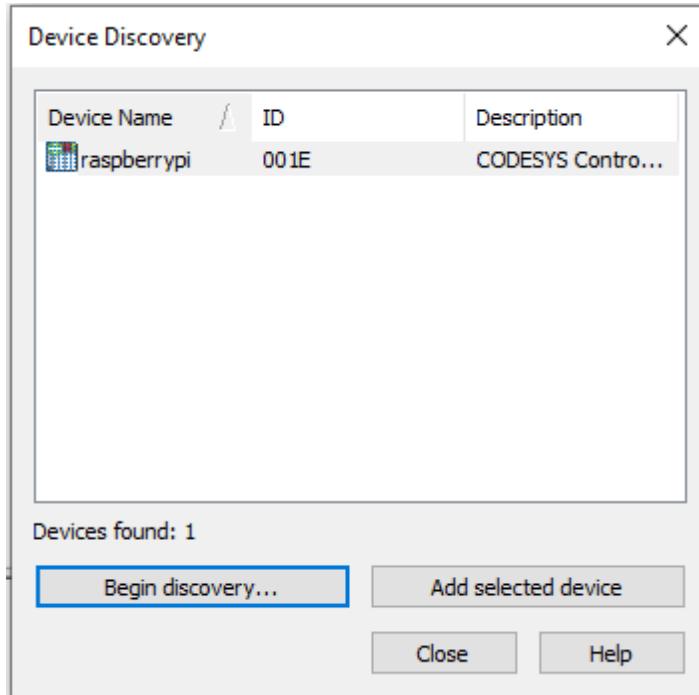


Figura 17. Seleccionar Raspberry. [Autores]

Cuando seleccione el dispositivo, parece sobre él y diríjase a las propiedades del mismo, ver Figura 18

Date	Time	Source
01/10/2021	14:06:05	KEPServerEX.R...
01/10/2021	14:06:11	KEPServerEX.C...
01/10/2021	14:06:25	CODESYS
01/10/2021	14:06:37	KEPServerEX.C...
01/10/2021	14:07:11	KEPServerEX.C...
01/10/2021	14:07:21	KEPServerEX.C...
01/10/2021	14:07:31	KEPServerEX.C...
01/10/2021	14:07:31	KEPServerEX.C...
01/10/2021	14:07:43	KEPServerEX.C...

Figura 18. Propiedades dispositivo. [Autores]

Diríjase a “Tag Import Settings”, un tag en KEPServerEX es la forma en que se llama a la variable o valor a comunicar por OPC UA.

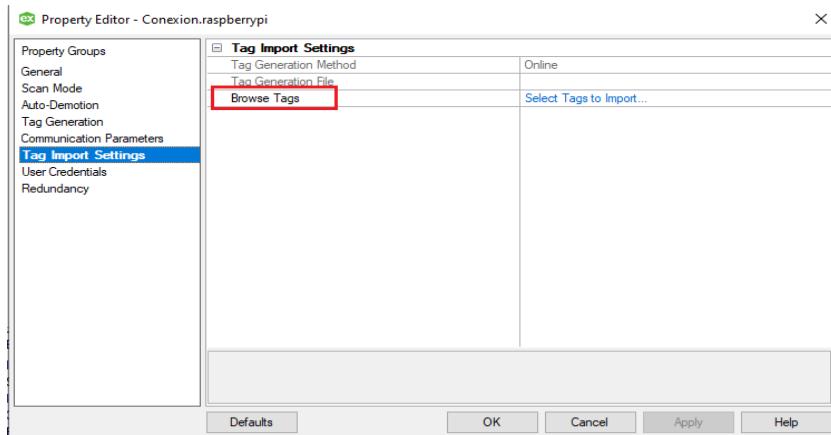


Figura 19. Importar Tags [Autores]

En la Figura 20, se importan las variables que se comunicaran hacia el sistema SCADA

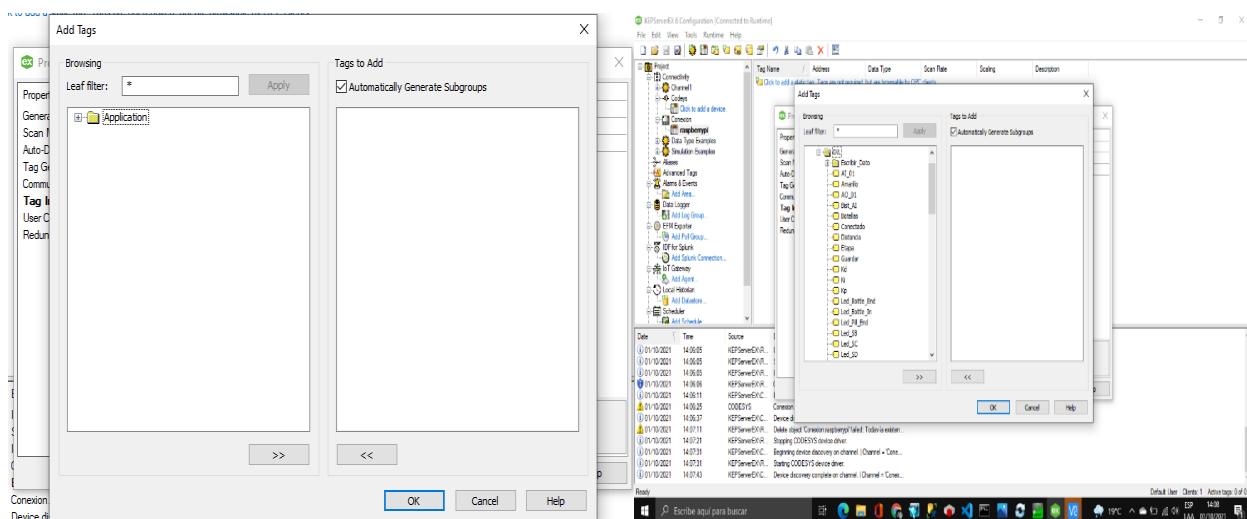


Figura 20. Agregar los tags de la configuración de símbolos. [Autores]

Luego de presionar OK, Oprima el recuadro llamado “Quick Client” y busque la opción “<nombrecanal>.raspberrypi_application”

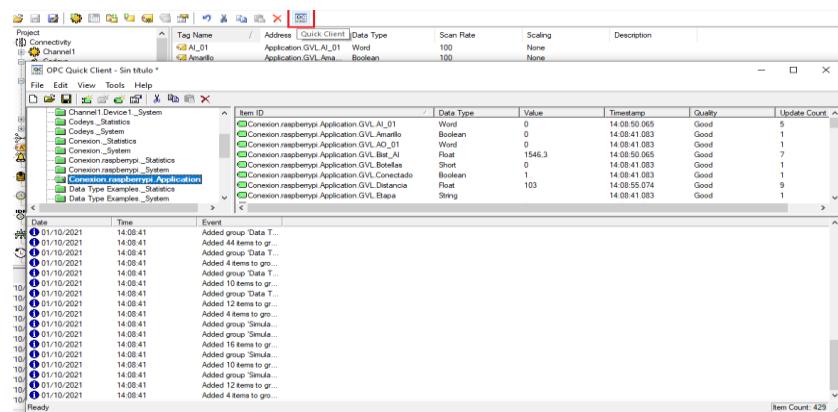


Figura 21. Quick Client. [Autores]

3. Crear grupo Acces Name en Intouch

Hasta el momento en el apartado 1 y 2 creamos el programa del proceso, se envían las variables por OPC UA usando KEPServerEX y observamos que el protocolo de comunicación está funcionando, Figura 21.

Ahora procederemos a crear un grupo de tags por medio de un alias ya que los tags no se pueden llamar directamente uno a uno (Intouch suele llamar a un grupo), Intouch llamará al alias que es un banco de variables en vez de llamar una a una por medio de su dirección.

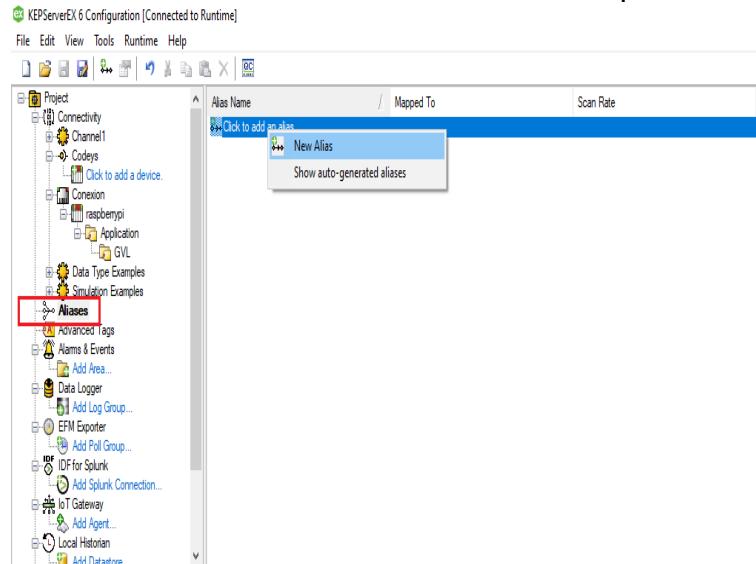


Figura 22. Crear Alias. [Autores]

Asignar un nombre, tenga en cuenta que este mismo nombre se usará en las configuraciones del Acces Name en Intouch (figura 23).

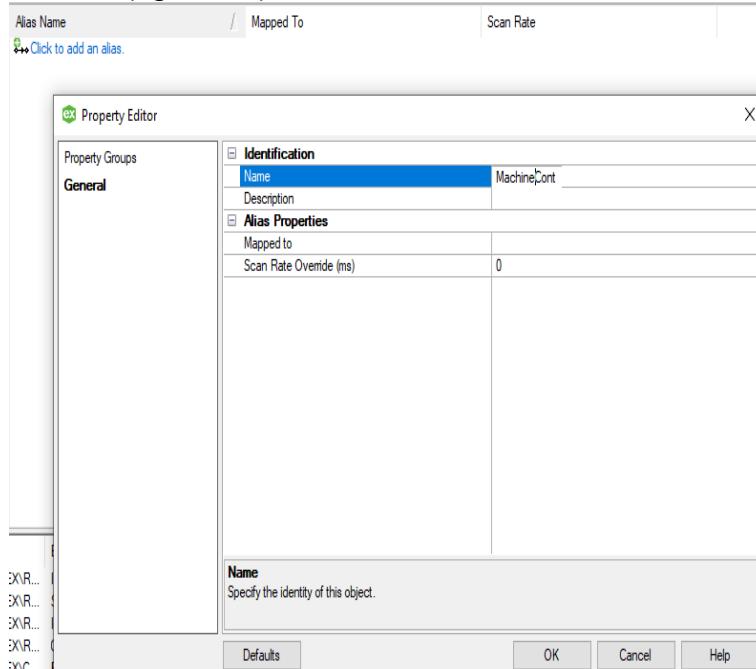


Figura 23. Propiedades generales del alias. [Autores]

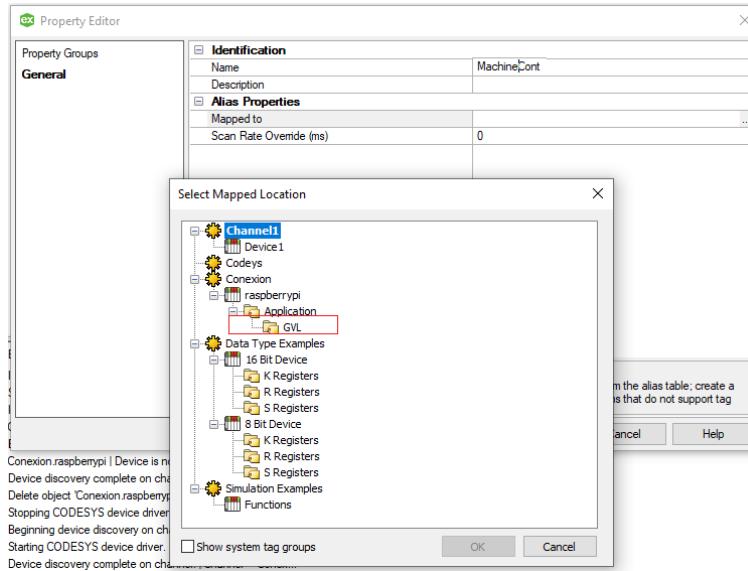


Figura 24. Mapeo de los tags importados [Autores]

Exporte los tags de la carpeta GVL, incluso podría crear unos nuevos tags en una carpeta y almacenar estos nuevos tags en el alias. En la figura anterior se exportan todas las variables del proceso.

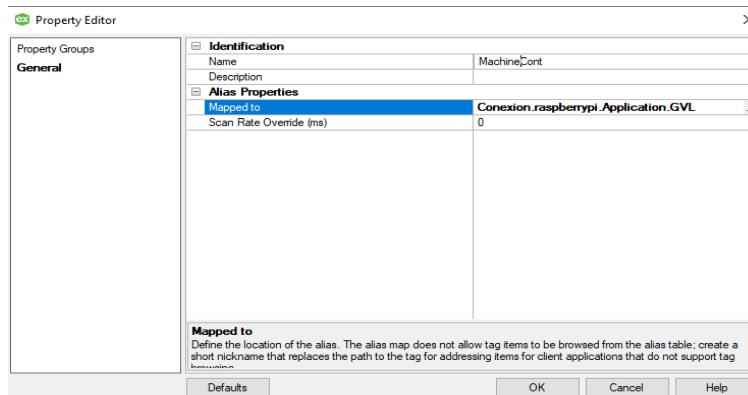


Figura 25. Configuración final del alias. [Autores]

Abra Intouch.

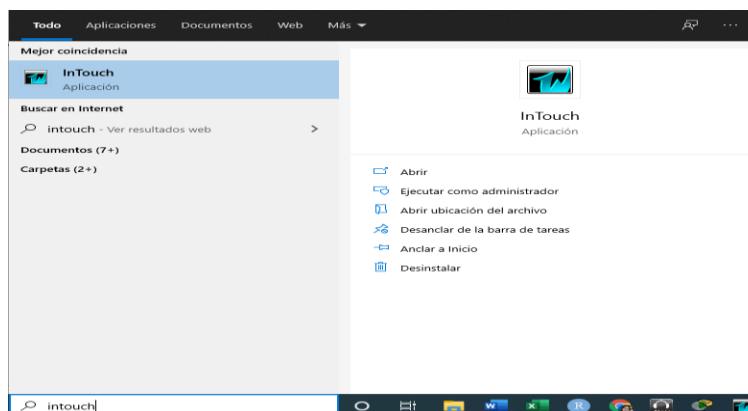


Figura 26. Intouch.[Autores]

Proceda a crear el access name

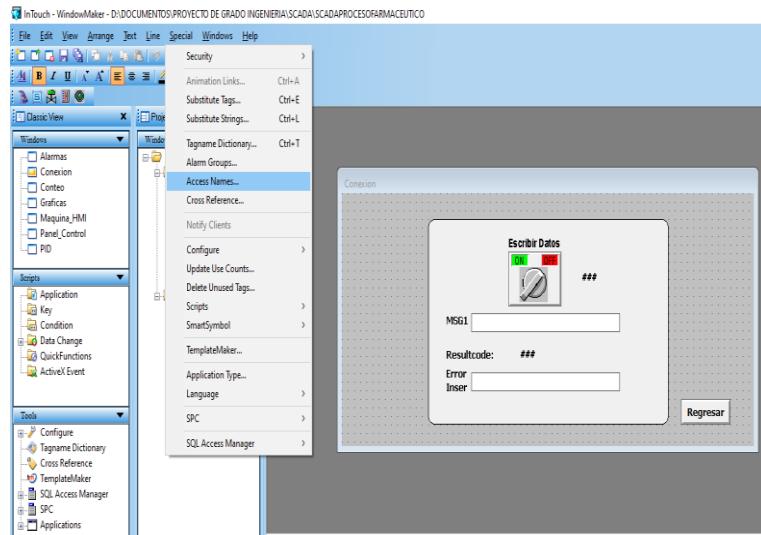


Figura 27. Crear Access Names. [Autores]

Como se observa en la figura siguiente, ya está creado, pero debes configurarlo como se muestra a continuación:

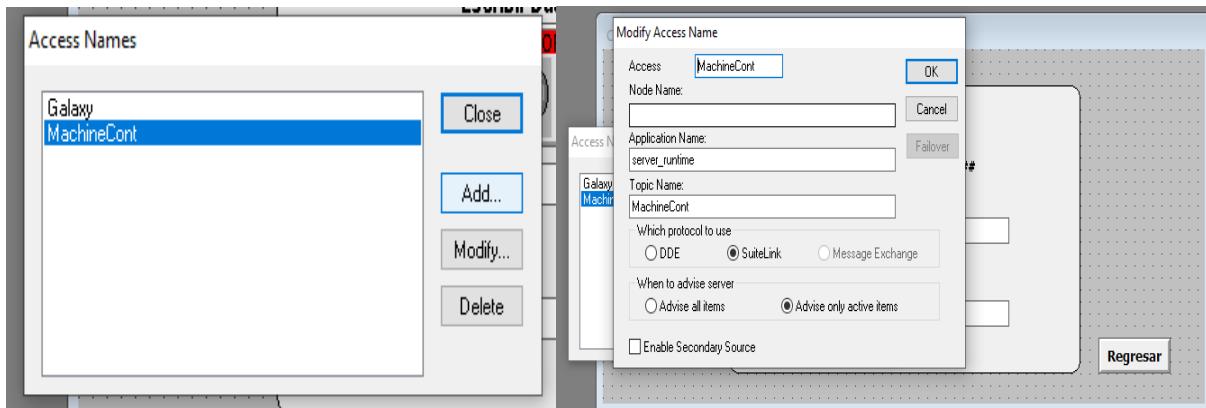


Figura 28. Configuración del Acces Name. [Autores]

Colocamos un nombre cualquiera a este nuevo Nombre de acceso.

- Node Name: No es utilizado puesto que no usamos datos de computadores remotos en la red.
- Application Name: Como el programa que nos suministra los datos es OPCLink, bajo este renglón se debe colocar el nombre de la aplicación de KEPServerEX. observe las figuras 24-26 para conocer esta variable
- Topic Name: Aquí debe colocarse el nombre exacto del Topic que se configuró previamente en KEPServer.

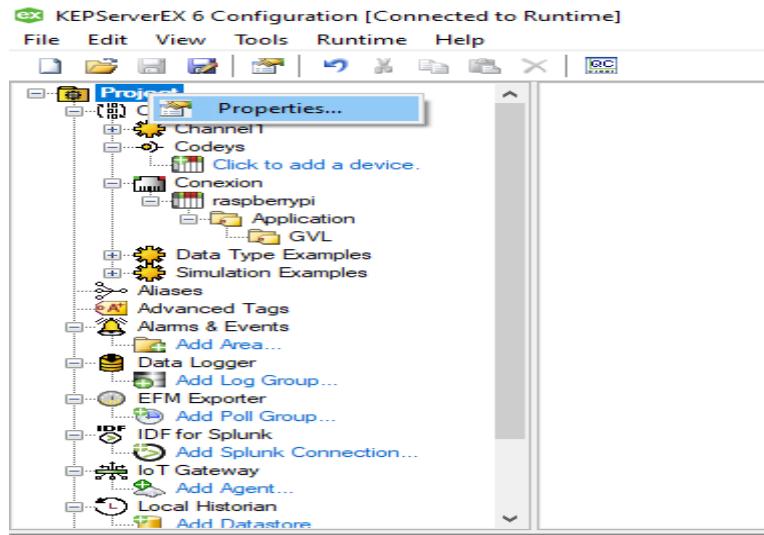


Figura 29. KepServerEX propiedades del proyecto. [Autores]

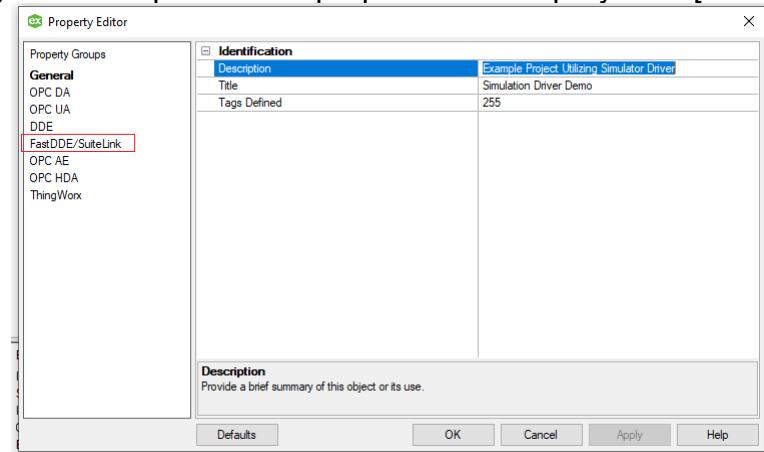


Figura 30. Propiedades. [Autores]

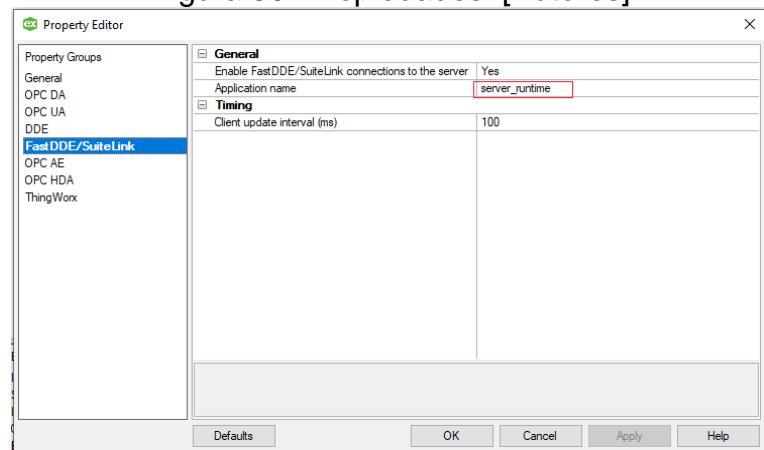


Figura 31. Obtener el application name. [Autores]

Para más información de cómo definir variables y usar el Topic Name creado en esta práctica diríjase a la “Importar/Exportar SCADA” donde se enseña a configurar variables en Intouch.

	UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo ACTIVIDAD: Test de la planta	
Código: PC003 <input checked="" type="checkbox"/> ONLINE <input type="checkbox"/> OFFLINE		Duración: ---

PC002. Test de la planta

Objetivo de la práctica:

Comprobar y evaluar el funcionamiento de los lazos de control de la máquina, por medio de su conexión con el sistema SCADA.

Material Necesario y requisitos para el desarrollo:

- Computador
- KEVServerEX
- Intouch Versión 10.0
- Codesys V3.5 SP16 o superior

Esquema Grafico de la Actividad:

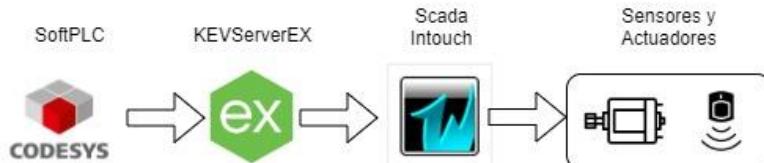


Figura 32: Esquema básico de la actividad de testeo. [Autores]

Requisitos previos:

- Reconocimiento Maquina de Conteo y SoftPLC.
- Conexión entre Intouch-Codesys OPCUA

Resumen:

La práctica de testeo se realiza para verificar el comportamiento de cada uno de los dispositivos de la planta, en este proceso se comprueba el funcionamiento de los sensores y se ejecuta de forma individual cada uno de los actuadores de la planta para así poder ver si existe algún fallo. Este proceso se realiza a través del SCADA que permite evaluar en la pestaña de panel de control los diferentes dispositivos de la máquina de conteo.

Desarrollo de la práctica

Antes de ejecutar las tareas que se designaran en la práctica es necesario establecer la comunicación con el softPLC desde codesys, tal como se explica en el manual de funcionamiento y luego emplear KEVServerEX para enviar la información al SCADA de igual forma como se realiza en la práctica PC001. Al haber completado estas actividades ya tendremos la maquina lista para poner a prueba su funcionamiento.

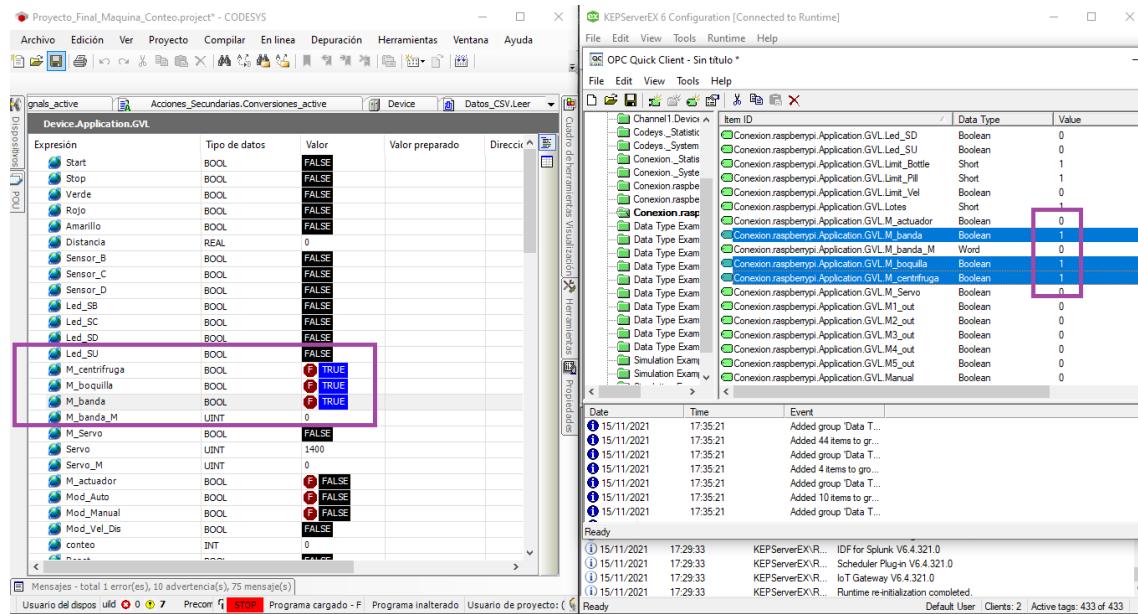


Figura 33: Verificación de la comunicación entre el SoftPLC y el SCADA. [Autores]

La imagen anterior muestra al sistema SCADA conectado al SoftPLC, desde codesys forzamos las variables y vemos como en el *quick client* de KEVServerEX aparecen, cuando se tenga esto nos dirigiremos a la pestaña de panel en la barra de desplazamiento de la interface principal, tal como se muestra a continuación:



Figura 34: Selección del panel de control para el proceso de testeo. [Autores]

Cuando estemos en esta pestaña de panel de control habilitaremos el modo de testeo (ver Figura 35) y realizaremos los puntos 1 y 2 de la práctica.

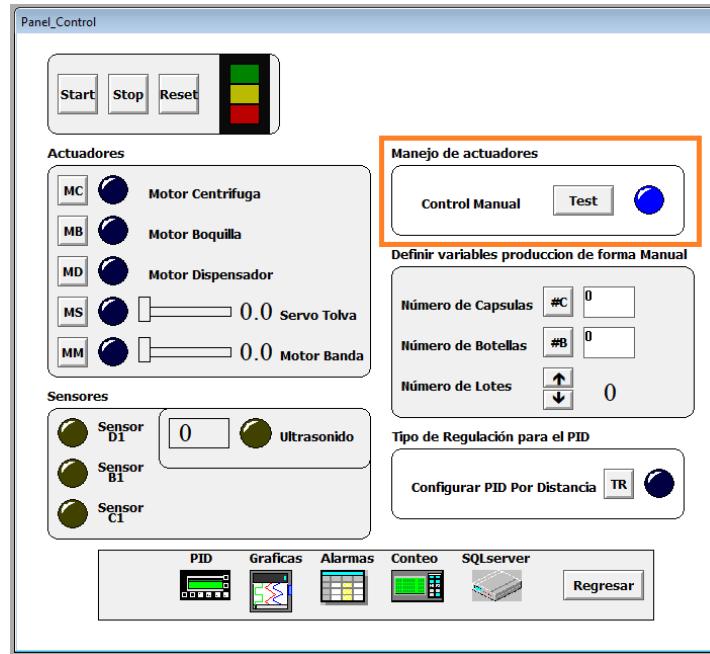


Figura 35: Habilitación del modo testeo en el panel de control del SCADA. [Autores]

1. Verificación de sensores

La pestaña del panel de control posee una sección donde podemos verificar si los sensores han sido activados o desactivados lo que se debe hacer es ubicar cada sensor y generar un cambio en él para comprobar su funcionamiento.

Es importante tener en cuenta que la planta maneja cuatro sensores, tres de ellos son on/off y uno es análogo. De acuerdo con esto en la siguiente imagen se observa un ejemplo de los cambios que se producen al comprobar los sensores de la maquina:

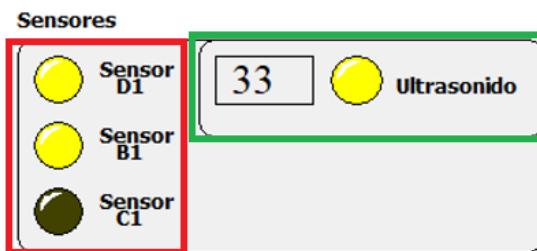


Figura 36: Verificación del estado de sensores. Los rojos son on//off. El Verde es variable. [Autores]

Cuando se compruebe cada sensor se debe emplear la tabla para anotar si su funcionamiento es correcto o hay alguna falla. Si se presenta algún daño realizar la anotación, de lo contrario chulear la casilla.

Sensor Banda	✓
Sensor Centrífuga	X
Sensor Dispensador	✓
Sensor Ultrasonido	✓

Tabla 2: Verificación del funcionamiento de los sensores. [Autores]

2. Verificación de actuadores

De igual forma la ventana del panel maneja una sección para comprobar el funcionamiento de los motores, estos se pueden verificar simplemente activando los botones. Al igual que los sensores, los actuadores se manejan de diferentes formas, tres de ellos son de uso fijo es decir son on/off y los otros dos restantes son regulables, cada uno se debe manejar con su respectiva forma. En la siguiente imagen se observa cómo se activa cada motor luego de habilitarlo con los botones del panel:

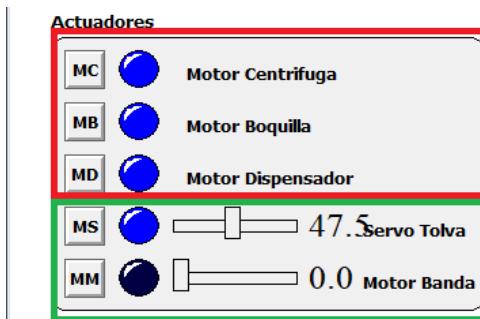


Figura 37: Verificación del estado de los actuadores. Los rojos son on/off. Los verdes son regulables. [Autores]

Después de verificar el funcionamiento de cada motor, se debe completar la siguiente tabla de la forma en se mencionó en el punto número 1 para la verificación de sensores.

Elemento	Funcionamiento
Motor Dispensador	
Motor Boquilla	
Motor Centrífuga	
Motor Servo	
Motor Banda	

Tabla 3: verificación del funcionamiento de los actuadores. [Autores]

Nota: Es importante tener en cuenta que tanto los sensores como los actuadores no solo tienen indicadores en el SCADA que le permiten al operario conocer el estado en el que se encuentra, también es necesario verificar si los indicadores físicos que tiene cada elemento está funcionando.

	UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo	
	ACTIVIDAD: Importar/Exportar SCADA	
Código: PC003	<input type="checkbox"/> ONLINE <input checked="" type="checkbox"/> OFFLINE	Duración: ---

PC003. Importar/Exportar SCADA

Objetivo de la práctica:

Exportar y cargar distintos programas en Intouch para así poder intercambiar información y trabajar más eficientemente en los grupos de trabajo.

Material Necesario y requisitos para el desarrollo:

-Intouch Versión 10.0

Requisitos previos:

- Reconocimiento Maquina de Conteo y SoftPLC.
- Conexión entre Intouch-Codesys OPCUA , no es obligatorio, pero en el apartado 3 se usan configuraciones definidas en esta práctica.

Resumen:

El documento pretende aclarar algunas inquietudes que pueden surgir a lo largo de la elaboración del Sistema SCADA, lastimosamente la versión de Intouch 10.0 no tiene una forma tradicional de importar programas provenientes de otros equipos, por lo cual debemos importar y exportar los SCADA de cierta forma.

Desarrollo de la práctica

1. Como exportar un Programa de Intouch

Para exportar un programa, haga lo siguiente, la siguiente figura es el SCADA completo de la máquina, no es necesario que importe todas las pantallas por cuestión de tiempos. El SCADA siguiente aún no tiene la conexión SQL y tenga en cuenta que, siempre que exporte un SCADA debe realizar de nuevo el BindList y la base de datos OnPremise en su equipo local para más información vea la práctica “Conexión entre Intouch-SQL”.

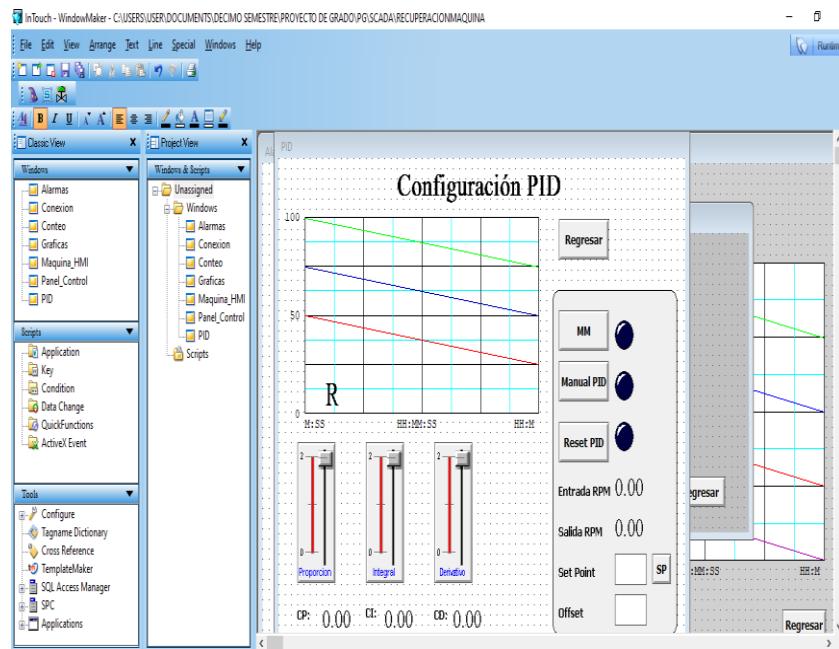


Figura 38. SCADA que deseamos exportar. [Autores]

Diríjase a File/Export Window

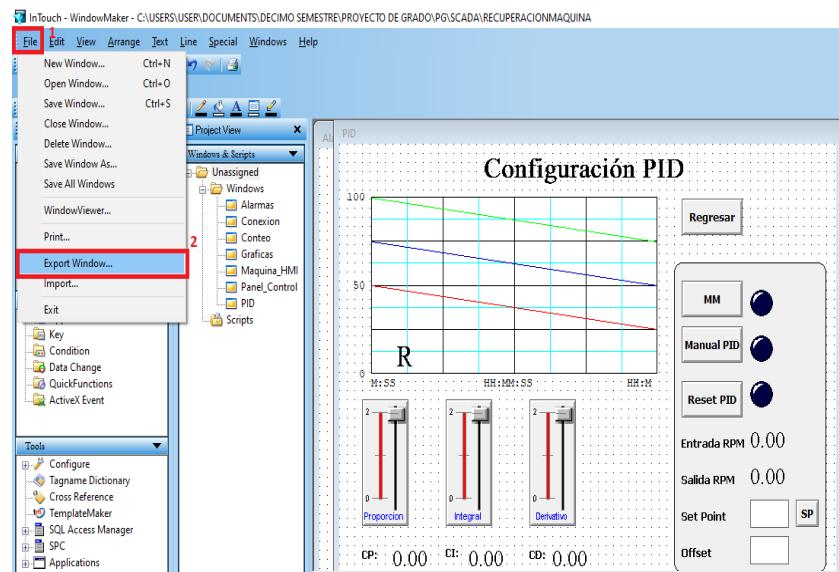


Figura 39. Export Windows. [Autores]

Note que en la Figura 40 se observa un error porque las ventanas están abiertas (Figura 39); por tal motivo de anticlick en cada ventana y “Close”

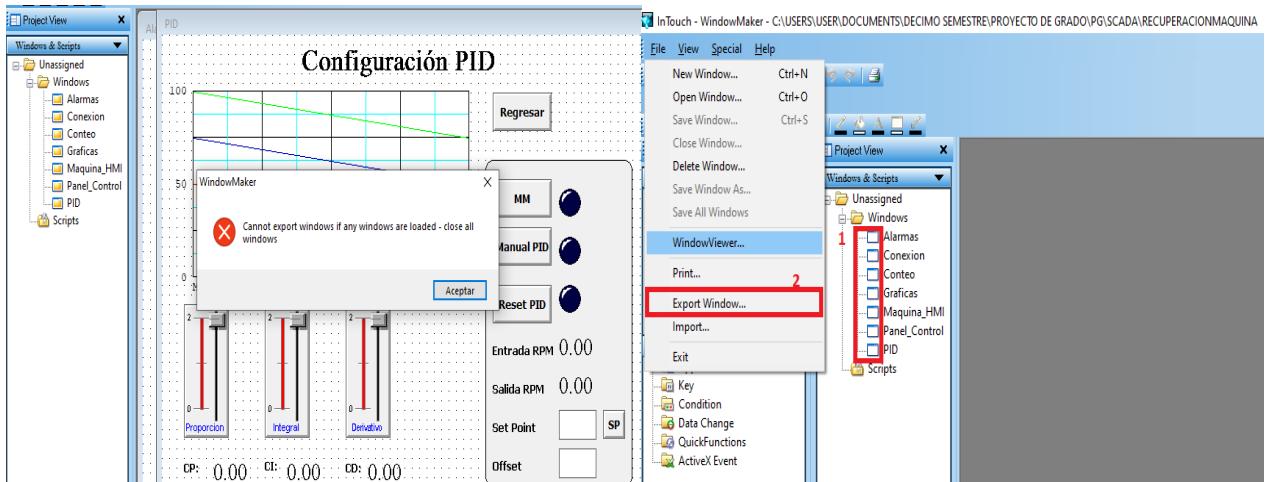


Figura 40. Error por ventanas abiertas [Autores]

Buscar la ruta donde quiere guardar el archivo

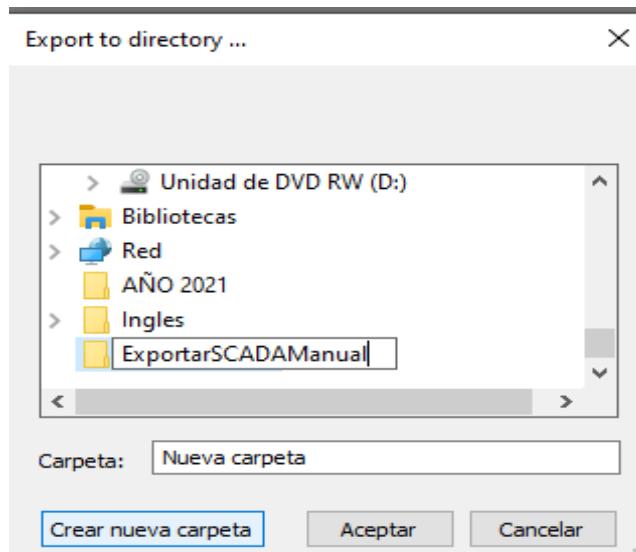


Figura 41. Buscar destino del archivo a exportar. [Autores]

Por último seleccione las ventanas que desea exportar como se ve en la Figura 42 y verifica en la Figura 43 que el archivo este exportado en la ruta seleccionada

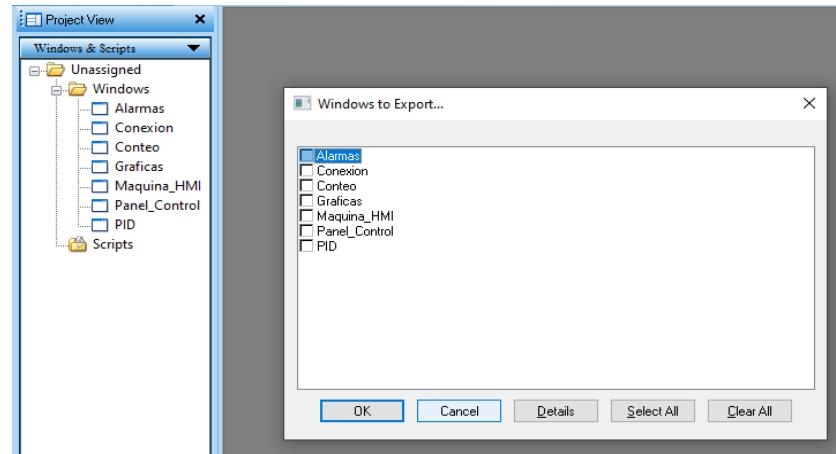


Figura 42. Seleccionar las ventanas a exportar. [Autores]

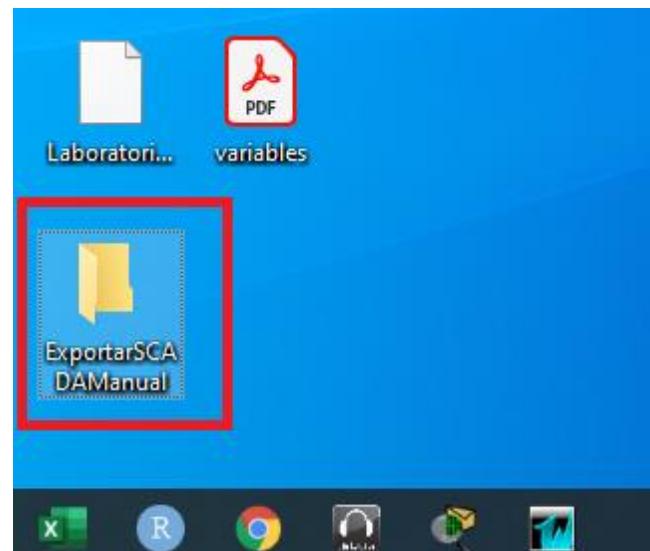


Figura 43. Buscar el destino de la carpeta. [Autores]

2. Como importar un Programa de Intouch

Para importar un programa primero debe crear una nueva aplicación en Intouch y luego importar la carpeta de la Figura 43.

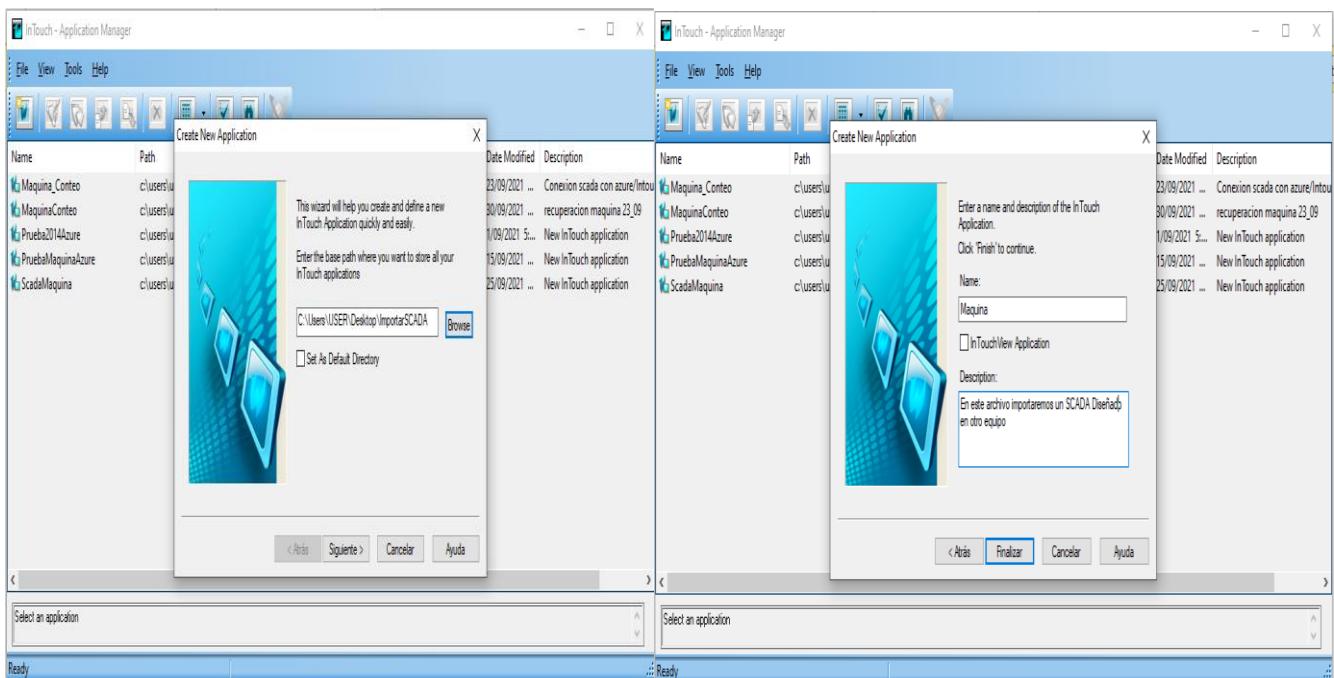


Figura 44. Creación de un nuevo SCADA. [Autores]

Seleccionar la opción de Importar y buscar el programa que se desea importar

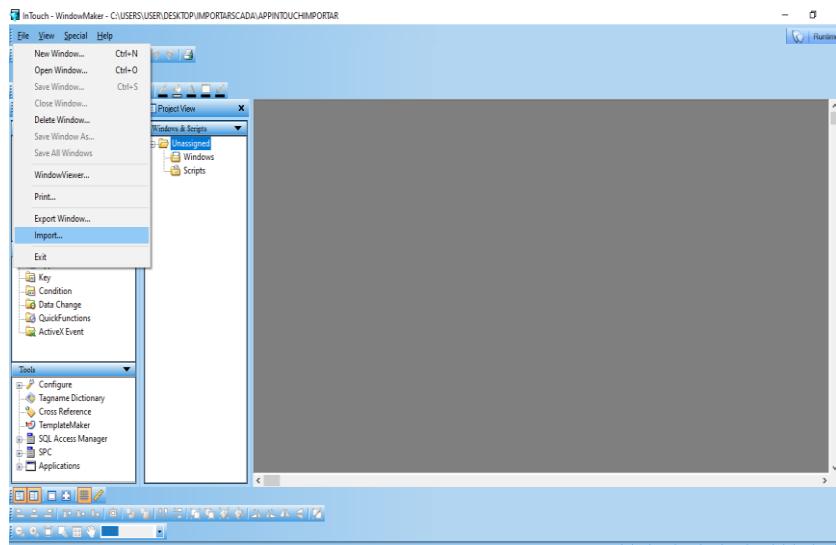


Figura 45. Importar SCADA. [Autores]

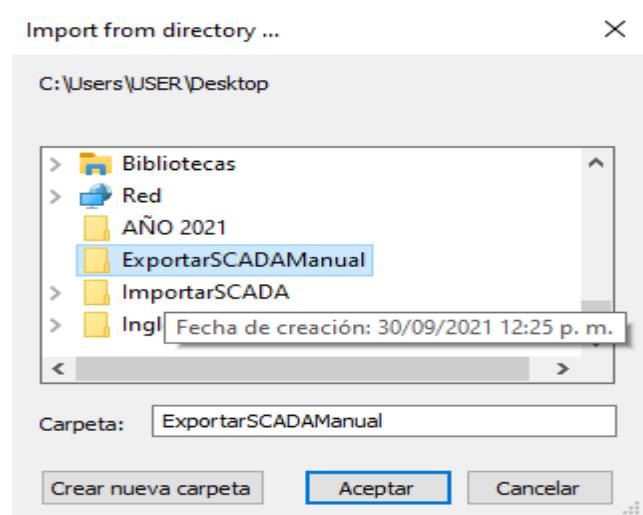


Figura 46. Buscar la carpeta del SCADA a importar. [Autores]

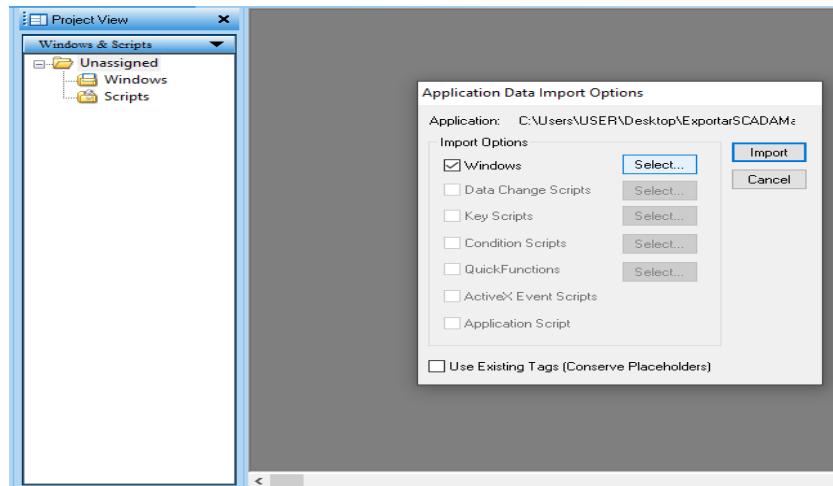


Figura 47. Opciones de importación. [Autores]

Como observa en la figura anterior Figura 47, se pueden importar varias configuraciones como Data Change Scripts o condicionales que se han creado; en el caso del Data Change creado en la práctica “Conexión entre Intouch-SQL”.

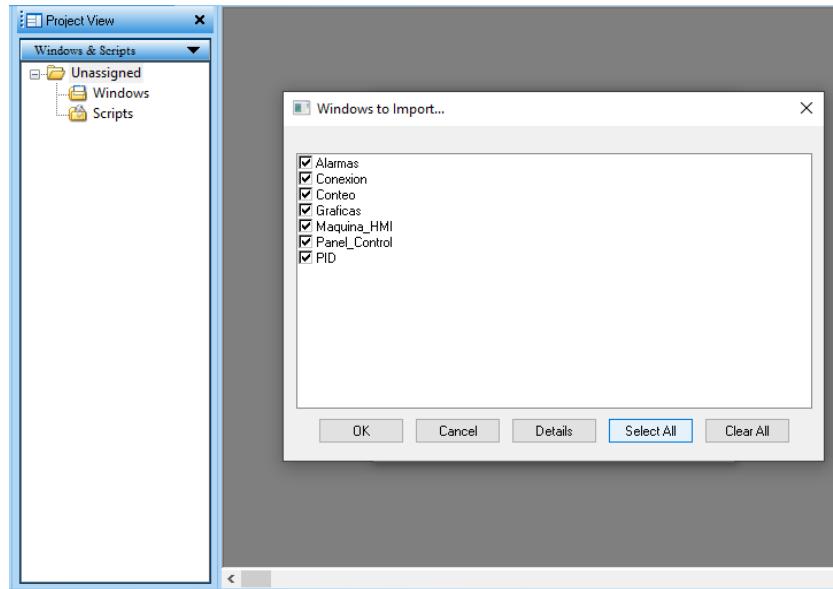


Figura 48. Importar las ventanas. [Autores]

Luego de seleccionar las ventanas que desea importar, confirme la elección como en la figura siguiente.

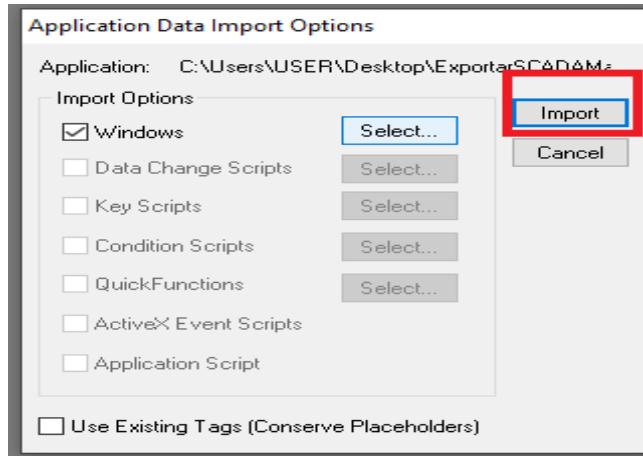


Figura 49. Importar. [Autores]

3. Cambiar variables luego de importar.

Al importar un SCADA lastimosamente toca volver a configurar muchas de las herramientas que ya estaban, observe en la Figura 50 procederemos a modificar la variable Total Botellas.

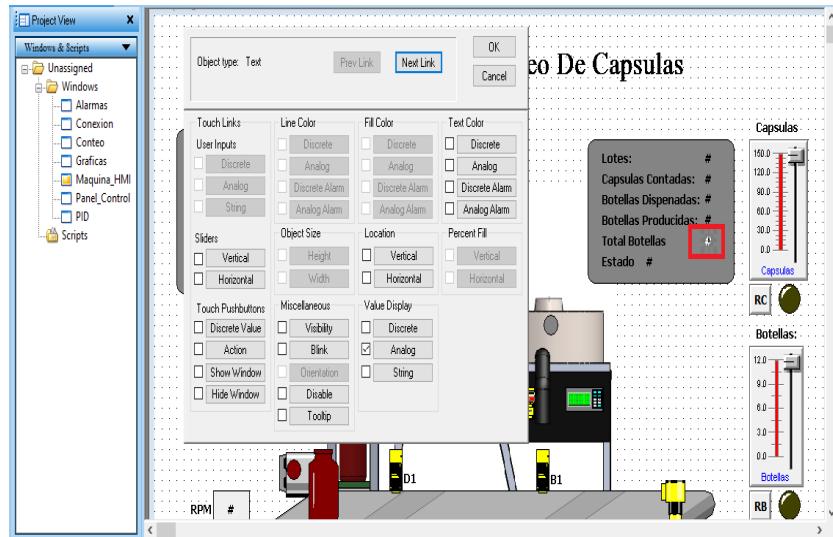


Figura 50. Modificar Variables. [Autores]

Se harán dos métodos, el segundo es más rápido que el primero, pero requiere cierto entendimiento en la creación de variables, tag name de Intouch pero no se preocupe ya que esta lo más explícito posible.

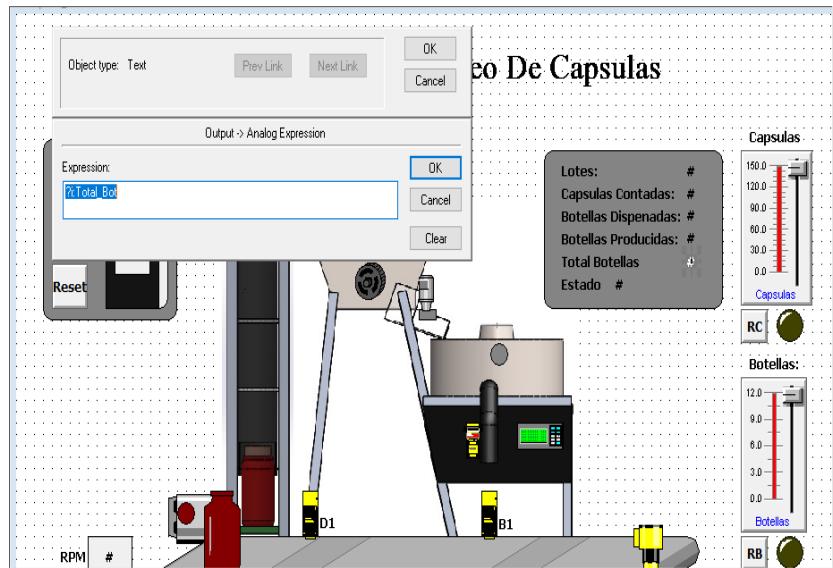


Figura 51. Variable Total_Bot. [Autores]

3.1 Método 1:

Como se observa en la figura anterior la variable esta definida como “?:iTTotal_Bot”, cuando se importa un SCADA, todas las variables aparecerán de esa forma, en el caso de la variable anterior, los símbolos “?:i” significa que la variable es entera(int) pero recuerde que hay variables asignadas a Keepserver y otras variables locales en el SCADA; por ende es necesario que entienda la configuración del Acces Name que se creo en la práctica “Comunicar Codesys Scada (Intouch)”

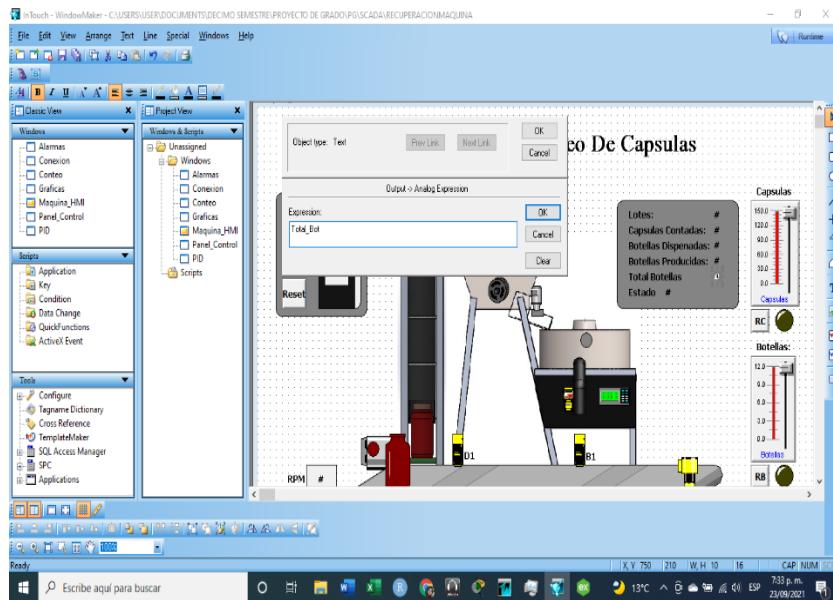


Figura 52. Variable Total_Bot Método 1. [Autores]

Use la tabla que se encuentra en el Anexo 1 y en el documento llamado “TagNameNames.docx”

	Discrete			
Test_Maquina	I/O	Mod_Manual		X
Total_Bot	I/O Integer	Total_Botellas		X
true	Memory Integer	---	¿WS.Conteo?	
Ultrasonido	I/O Real	Distancia		X
Velocidad	I/O Integer	RPM		X
Velocidad_Out	I/O Real	Out_RPM		X
Y_Manual_PID	I/O Discrete	Y_Manual_PID		X

Figura 53. Variable Total_Bot Método 1, usar tabla recopilación. [Autores]

En la imagen anterior se muestra una recopilación de todas las variables usadas en el SCADA, en la primera columna está el nombre de la variable en Intouch, la segunda columna es el tipo de variable, la tercera es el nombre de la variable en Codesys.

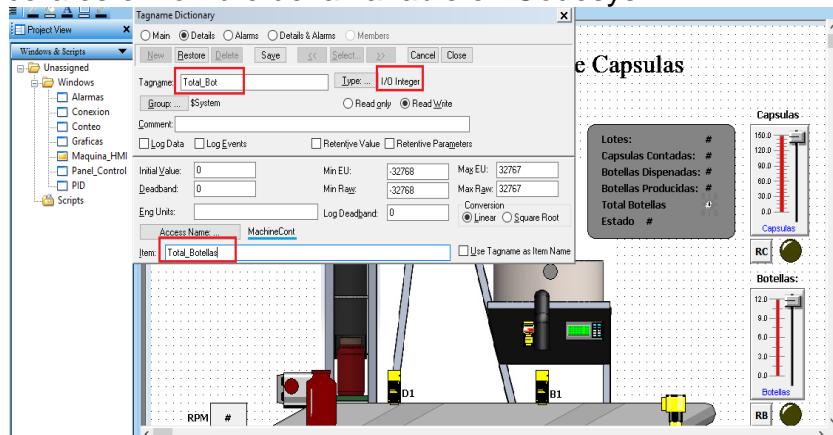


Figura 54. Variable Total_Bot Método 1 definir el Acces Name. [Autores]

En azul se observa la creación del “Acces Name”, esta configuración esta explicada a mayor detalle en la práctica “Conexión entre Intouch-SQL”.

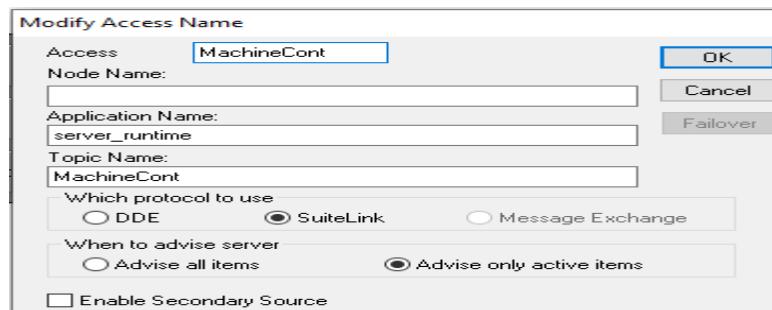


Figura 55. Variable Total_Bot Metodo 1 configurar Acces Name. [Autores]

Ya con esto hemos terminado de modificar una de las variables del SCADA, recuerde que usted debe configurar las que crea necesitar para su proceso

3.2 Método 2:

Vaya a la interfaz de anticlick y seleccione todo

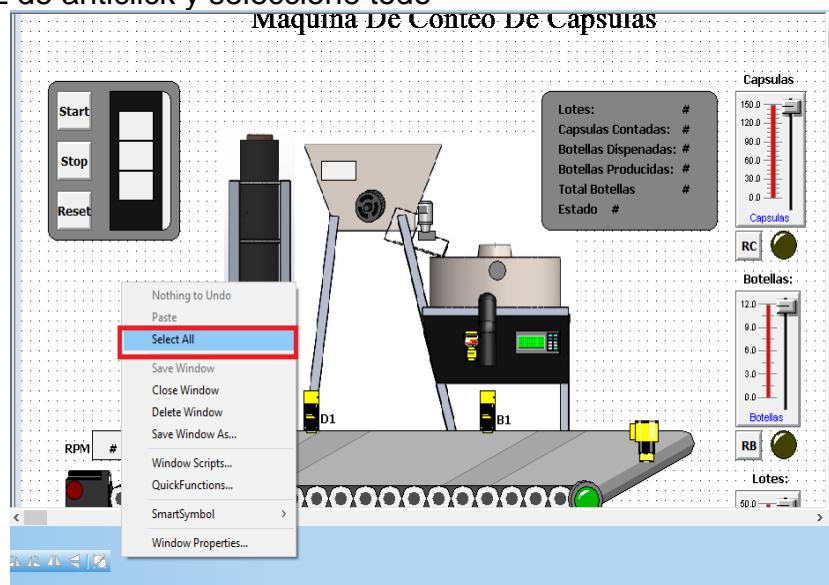


Figura 56. Modificar variables con Metodo 2. [Autores]

Después de haber seleccionado todo como se muestra en la Figura 57, debe ir a las opciones especiales de Intouch, Figura 58

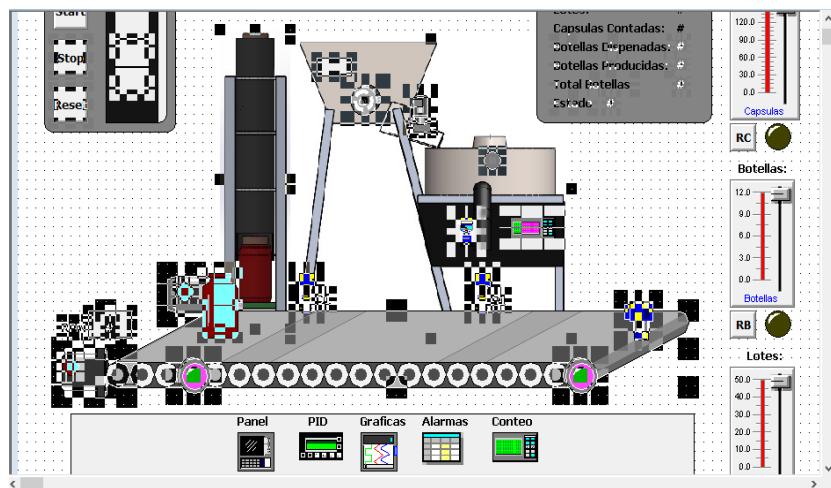


Figura 57. Método 2, seleccionar todo. [Autores]

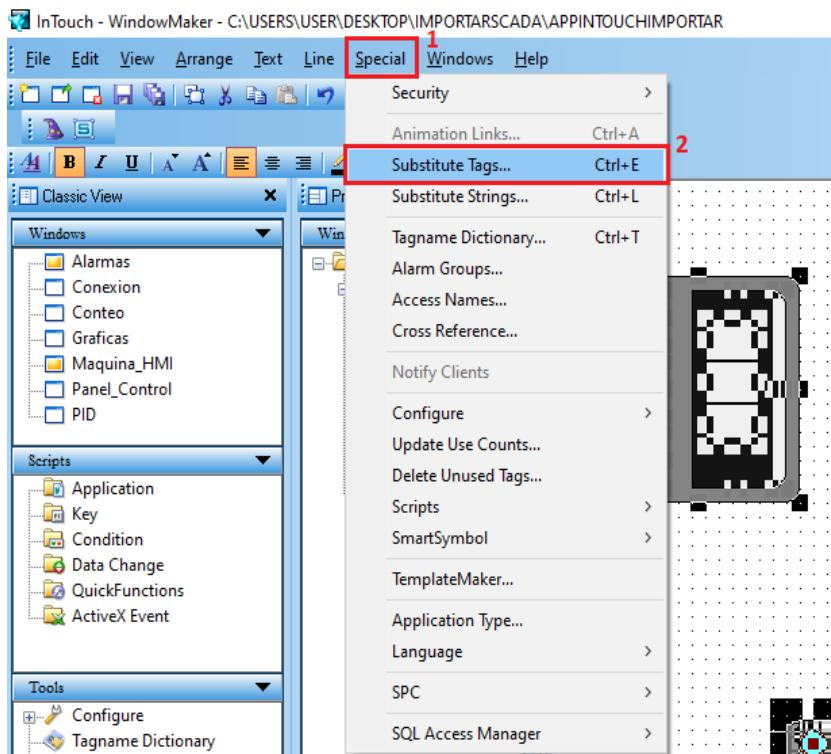


Figura 58. Método 2, Opción sustituir tags. [Autores]

Ahora se procede a sustituir los tags, observe que ya se modificaron “Salida_Amarilla” y “Salida_Rojo”, debes eliminar “?:d” y aparece el mensaje de la Figura 60.

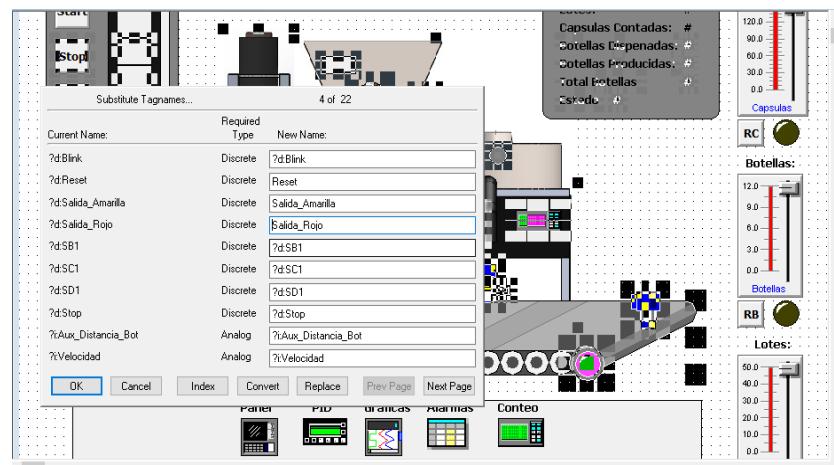


Figura 59. Método 2, sustituir Tags. [Autores]

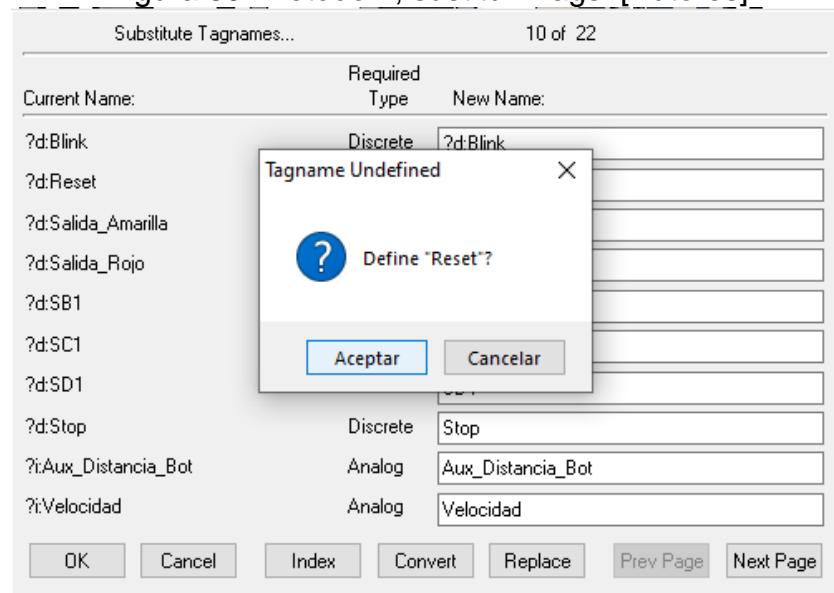


Figura 60. Método 2, variable Reset. [Autores]

Diríjase de nuevo a la tabla de recopilación de variables y configura la variable según la información siguiente:

Num Botellas	I/O Integer	Num_Bottle			X	X	
Num Pastas	I/O Integer	Num_Pill		X	X	X	
Offset_Text	Memory Message						X
Out RPM	I/O Real	Out_RPM					X
Reset	I/O Discrete	Reset		X	X		
Reset_Botellas	I/O Discrete	Reset_Cont_Bottle					
Reset_Capsulas	I/O Discrete	Reset_Cont_Pill					

Figura 61. Metodo 2, variable Reset usar tabla recopilación. [Autores]

Cuando termine de configurar Reset, puede seguir con todas las variables.

La ventaja del método 2 con respecto al 1 es que no se nos va a pasar ninguna variable porque al seleccionar todos los elementos, Intouch automatiza la tarea de reemplazar las variables.

En la Figura 59 en el botón “Next Page” se puede seguir configurando las distintas variables seleccionadas. Debes hacer esto para cada variable que desee visualizar en Intouch y recuerde que en algunas interfaces hay código en el “Windows Script”

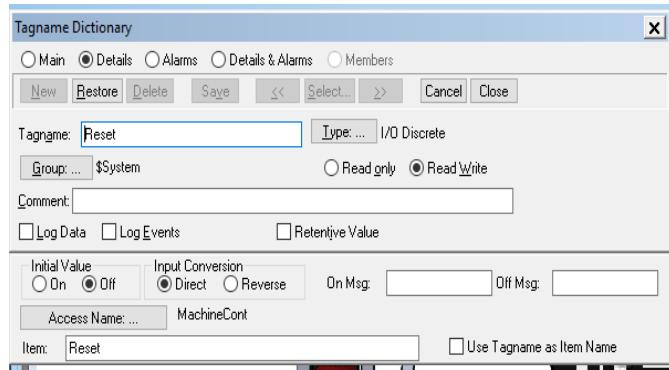


Figura 62. Método 2, variable Reset modificación. [Autores]

	<p>UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo</p> <p>ACTIVIDAD: Crear una base de Datos SQL Server</p>	
Código: PC004 <input checked="" type="checkbox"/> ONLINE <input type="checkbox"/> OFFLINE		Duración: ---

PC004. Crear una base de Datos SQL Server

Objetivo de la práctica:

Se busca que el estudiante aplique sus conocimientos y demuestre sus habilidades en los temas de bases de datos a través de una conexión generada entre Intouch (SCADA) y una base de datos On-premise (local).

Material Necesario y requisitos para el desarrollo:

- SQL Server Management Studio 2014 en adelante (Se usará v. 18).
- Intouch Versión 10.0

Esquema Grafico de la Actividad:

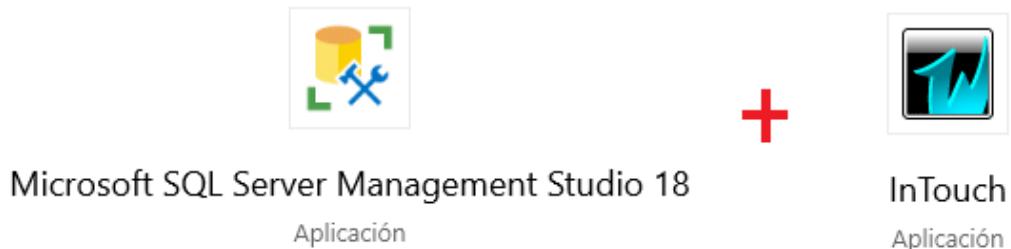


Figura 63. Esquema básico de la actividad a realizar. [Autores]

Requisitos previos:

- Reconocimiento Maquina de Conteo y SoftPLC
- Conexión entre Intouch-Codesys OPCUA (Practica #1)
- Importar/Exportar SCADA (Practica #2)

Resumen:

Un SCADA, permite supervisar y controlar el proceso; se caracteriza por: Alarmas, Trends, Interfaces (HMIs), registros de datos. Los sistemas SCADA poseen historiadores que

recolectan datos y permite registrar con eficiencia los eventos y datos de una planta; permitiendo que los pisos superiores de la pirámide de automatización hagan controles de calidad y optimización de procesos. Debe tener en cuenta que hay software de historiadores como Canary, Osisoft PI, etc

Una ventaja de SQL es su integralidad con otros sistemas; además recuerde que una de las competencias de la analítica es la capacidad de extraer información de distintas fuentes:



Figura 64. Diferentes orígenes de datos en analítica y ciencia de Datos.

Por ultimo recuerde las distintas alternativas para almacenar datos:

- PLC a SQL
- PLC a Historian (Canary, Osisoft PI)
- PLC a Nube (AW, Plataformas IoT)

Al trazar un esquema general de lo que debemos hacer, se debe comenzar por crear las variables y elementos que poseerá nuestro SCADA por ende es necesario que haya elaborado la práctica “Conexión entre Intouch-Codesys OPCUA” porque de esa forma ya esta recibiendo datos en el SCADA los cuales serán almacenados en SQL.

Luego de crear las variables en el SCADA, se procede a crear la base de datos en SSMS (SQL Server Management Studio), se crea la conexión ODBC que permite acceder a los archivos de diferentes bases de datos como Acces, Excel, SQL, entre otros. Luego procederemos a crear un BindList en Intouch, generar una interface sencilla de conexión y generar el Script en Intouch para usar las funciones:

- SQLInsert, SQLConnect, entre otros

Por último se comprueba la conexión entre el SCADA y SQL. Como trabajo posterior se recomienda aprender sobre extracción de datos y tratamiento de los mismos con SQL server, en el siguiente enlace hay información libre y otras pagas con certificado:

<https://www.codecademy.com/search?query=sql>

https://youtube.com/playlist?list=PLFNWVPtjBMju_UZdG3PLige5PBNAEQDmR

Al finalizar piense en los beneficios que nos ofrece SQL Server en escalabilidad, por tal razón realice la práctica llamada “Base de datos SQL on premise sincronizada con Azure a través de Data Sync”

Desarrollo de la Actividad:

1. Creación Base de datos en SSMS

Diríjase al buscador de su equipo y escriba “**SQL Server Configuration Manager**”, verifique que los servicios estén en estado “Running”

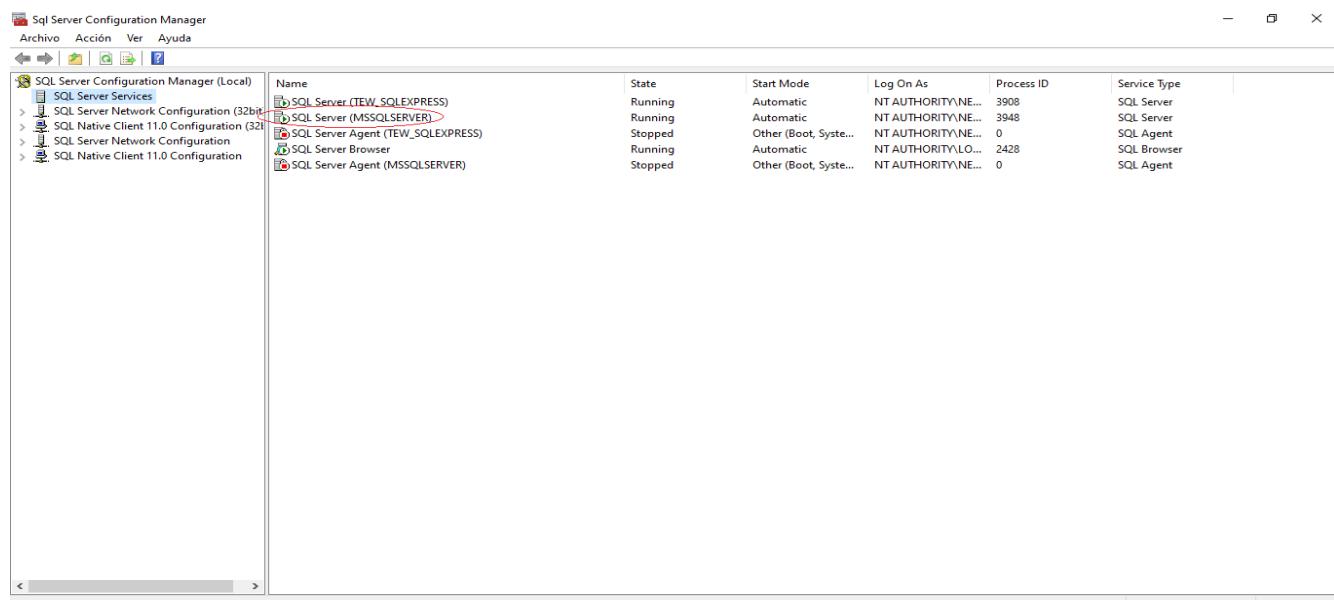


Figura 65. Sql server configuration Manager. [Autores]

Al abrir “Microsoft SQL Server Management Studio 18”, Deberá primero conectarse a su servidor local por tal motivo

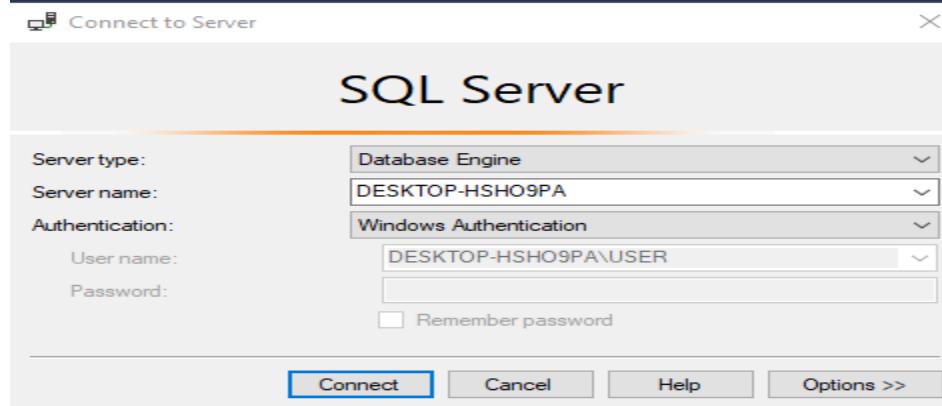


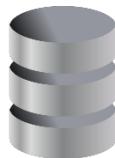
Figura 66. Conectarse al servidor (local). [Autores]

De aquí en Adelante al referirnos a la base de datos SQL en el proyecto se hablará en términos de “on-premise”, esto se hace porque en la transición de digitalización de las compañías

siempre se habla de on premise (on-site) y la migración de estos servicios a la nube (Cloud Computing)

En la Figura 66 si usted cuenta con un super usuario el cual definió al instalar SSMS debe seleccionar la opción de “SQL Server Authentication” y debe escribir el username y password respectivos; en nuestro caso no hay un super usuario pero no se preocupe más adelante crearemos un usuario con permisos para manejar la base de datos y así tener seguridad en el envío y manejo de datos.

Fíjese que en la Figura 67, se observa la conexión con el server local, algunas bases de datos on premise con su símbolo característico de “Database”



Es probable que en su servidor local SQL no tenga databases, en primer lugar, se debe crear la base de datos on premise, en la cual crearemos posteriormente un login y una tabla. Solo debe configurar la opción generica como muestra la Figura 67.

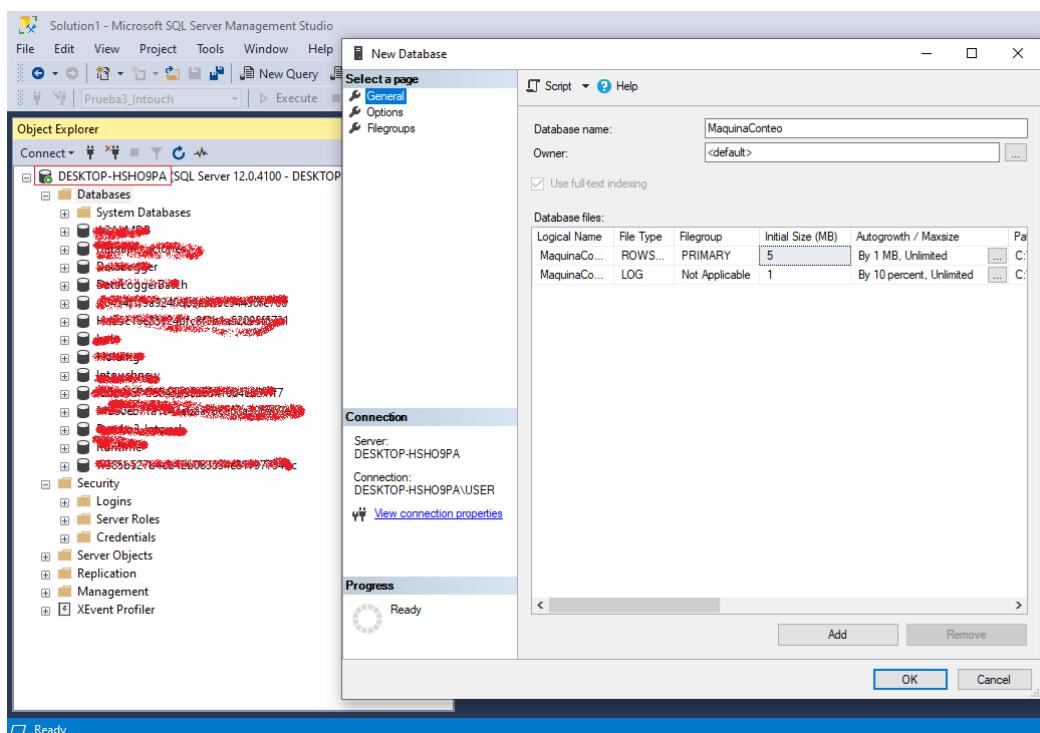


Figura 67. Crear Database. [Autores]

Ahora debe crear un Login en la carpeta Security. Debe pararse en la carpeta Logins / click derecho/ New login

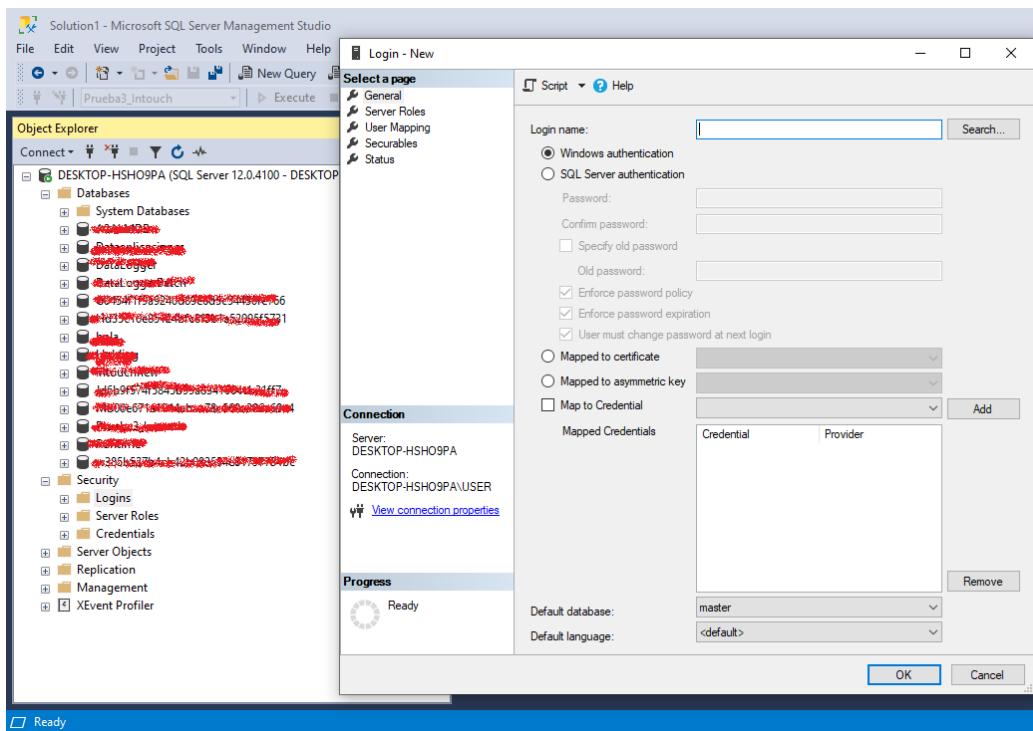


Figura 68. Crear un Nuevo Login. [Autores]

Cree un login name y seleccione una contraseña para el mismo y deje las opciones como se muestran a continuación.

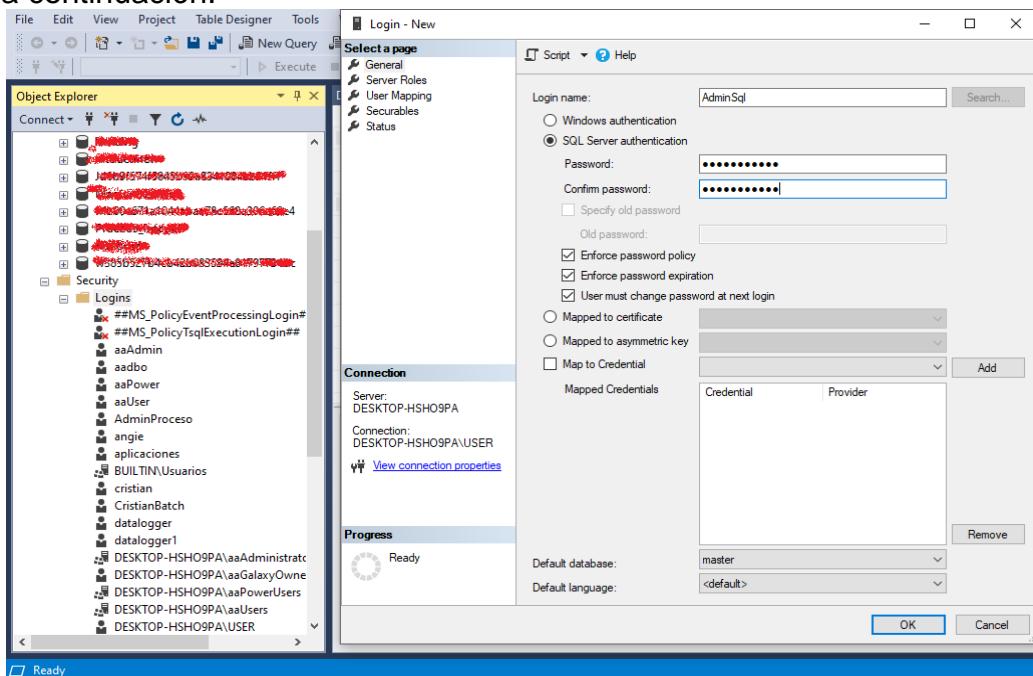


Figura 69. Password and Login name. [Autores]

En la opción “User Mapping” del Login debemos asignar nuestro login a la base de datos que creamos en la Figura 67 . Además, debe seleccionar los permisos que se muestran en la Figura 70.

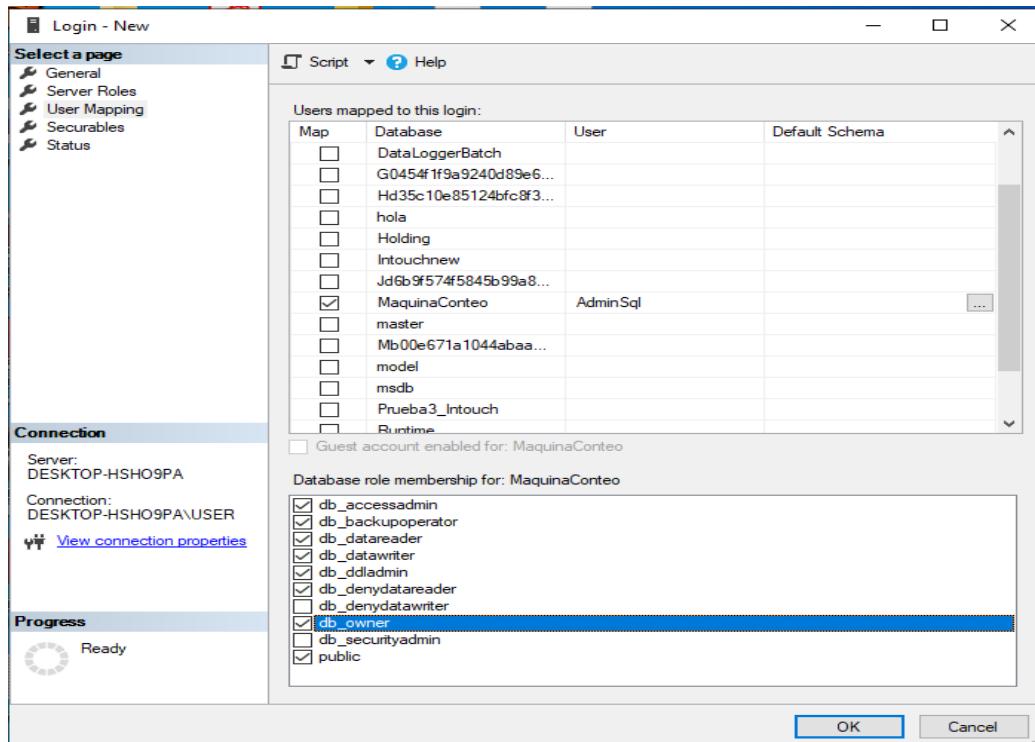


Figura 70. Pestaña User Mapping. [Autores]

Presione Ok y verifique la creación del Login como se observa en la Figura 69 donde podemos notar que hay varios usuarios y Logins creados.

Ahora debemos crear nuestra tabla que estará contenida en el database “MaquinaConteo”, para esto diríjase a su database y expándalo hasta llegar a la carpeta “tables”, click derecho y “crear tabla”

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane, 'Object Explorer', displays the database structure. The right pane shows the 'Customer' table definition. The table 'DatasetProceso' has the following columns:

Column Name	Data Type	Allow Nulls
Datold	int	<input type="checkbox"/>
Fecha	nchar(50)	<input checked="" type="checkbox"/>
Hora	nchar(50)	<input checked="" type="checkbox"/>
NoBotellas	int	<input checked="" type="checkbox"/>
LimBotellas	int	<input checked="" type="checkbox"/>
NoPildoras	int	<input checked="" type="checkbox"/>
LimPildoras	int	<input checked="" type="checkbox"/>
Lotes	int	<input checked="" type="checkbox"/>
Etapa	nchar(50)	<input checked="" type="checkbox"/>
SensorDispensador	int	<input checked="" type="checkbox"/>
SensorBanda	int	<input checked="" type="checkbox"/>
SensorCentrifuga	int	<input checked="" type="checkbox"/>
SensorUltrasonido	real	<input checked="" type="checkbox"/>
MotorActuador	int	<input checked="" type="checkbox"/>
MotorBanda	int	<input checked="" type="checkbox"/>
MotorCentrifuga	int	<input checked="" type="checkbox"/>

Figura 71. Tabla del proceso (completo). [Autores]

Al crear nuestra tabla, aparecerá en blanco, pero observe que en la Figura 71 ya esta creada toda la tabla para el proceso. No es necesario que cree todas las variables ya que es un trabajo de mayor tiempo, recomendamos que cree la tabla hasta la columna “Lotes”. Observe que el “Datold” es una llave primaria que es OBLIGATORIA porque para sincronizar una base on premise a Azure la tabla debe tener una llave primaria; una llave primaria es un identificador único y que puede relacionar varias tablas. Todas las variables booleanas de nuestro SCADA se deben asignar como “int” en sql porque el arrojara ceros y unos, mas no TRUE y FALSE, cuando necesitamos almacenar strings del SCADA es necesario definirlos como nchar(caracteres), tenga cuidado con la longitud ya que al ser muy pequeña arrojara un error al almacenar el dato, recomendamos dejarla en 50.

Ahora en la Figura 72 mostraremos como crear una llave primaria para la columna Datold

The screenshot shows two windows side-by-side. The left window displays the context menu for a column named 'Datolid' in a table named 'Table_1'. The 'Set Primary Key' option is highlighted. The right window shows the table structure with 'Datolid' as the primary key.

Figura 72. Configuración llave primaria. [Autores]

Es importante en “Column properties” seleccionar Yes en el aparato de (Is Identity). Cuando termine de crear las columnas, el tipo de variable, debe guardar la tabla dando click derecho en el nombre “DESKTOP-HSHO9PA...Table_1*” y guárdela como DatasetProceso.

Al guardar la tabla, podemos realizar una Query (petición) para consultar nuestra tabla para esto puede presionar la opción “”New Query”, aunque sabemos que no todos conocemos muchos comandos sql por lo tanto usaremos la ayuda de SSMS: Seleccione la tabla, click derecho y seleccione “Select top 1000 Rows”.

The screenshot shows the Object Explorer on the left and a table named 'Table_1' in the center. The context menu for the table is open, and the 'Select Top 1000 Rows' option is highlighted.

```

SELECT TOP (1000) [DatoId]
,[Fecha]
,[Hora]
,[Start]
,[Stop]
,[NoBotellas]
,[LimBotellas]
,[NoPildoras]
,[LimPildoras]
,[Lotes]
,[Etapa]
,[SensorDispensador]
,[SensorBanda]
,[SensorCentrifuga]
,[SensorUltrasonido]
,[MotorActuador]
,[MotorBanda]
,[MotorCentrifuga]
,[MotorBoquilla]
,[ServoM_Porcen]
,[RPMBanda]
FROM [MaquinaConteo].[dbo].[DatasetProceso]
  
```

Figura 73. Hacer una query usando el asistente. [Autores]

Se abrirá un script con la línea de código:

```

SQLQuery7.sql - D...SHO9PA\USER (58) DESKTOP-HSHO9PA...o.DatasetProceso SQLQuery6.sql - D...SHO9PA\USER (52)
/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [DatId]
,[Fecha]
,[Hora]
,[NoBotellas]
,[LimBotellas]
,[NoPildoras]
,[LimPildoras]
,[Lotes]
,[Etapa]
,[SensorDispensador]
,[SensorBanda]
,[SensorCentrifuga]
,[SensorUltrasonido]
,[MotorActuador]
,[MotorBanda]
,[MotorCentrifugal]
100 %
Results Messages
DatId Fecha Hora NoBotellas LimBotellas NoPildoras LimPildoras Lotes Etapa SensorDispensador SensorBanda SensorCentrifuga SensorUltrasonido

```

Figura 74. Comprobar la creación de la tabla DatasetProceso. [Autores]

2. Conexión ODBC

En el caso de la conexión ODBC debe abrir las “Herramientas administrativas de Windows”, como se observa en la Figura 75

Al abrir ODBC Data Source tendremos ya sea la opción de 32 bits o la de 64 bits, en nuestro caso creamos la conexión ODBC en 32 bits. Cuando abra el programa note que esta “DSN Usuario” y “DSN Sistema”, si sumerse crea la conexión en DSN sistema servirá en todo su PC y no solo en el usuario presente, por tal razón seleccione DSN Sistema en la Figura 76.

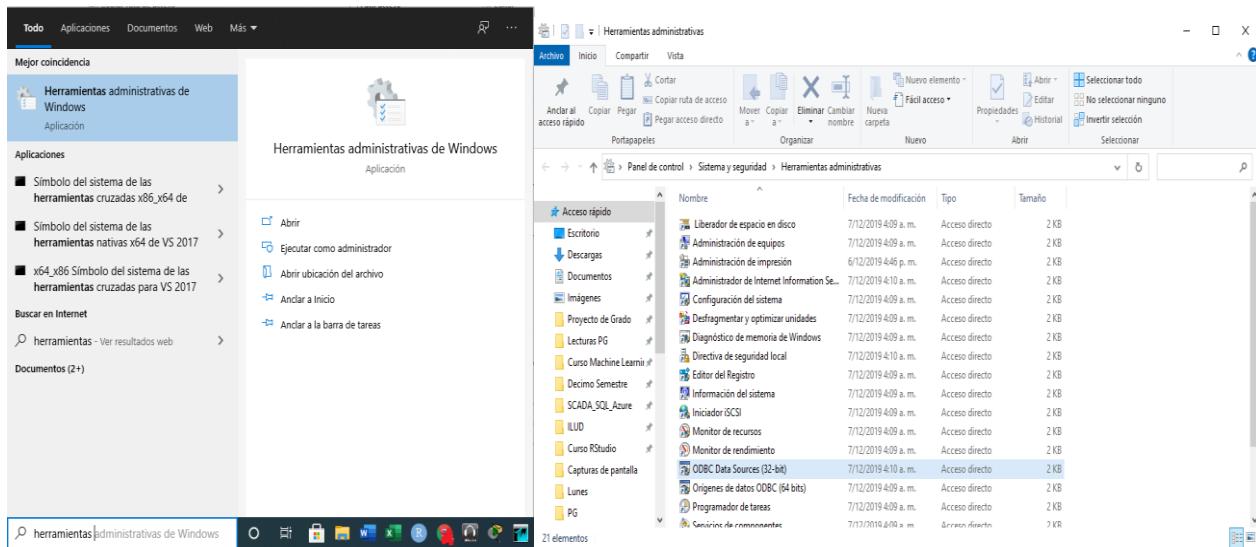


Figura 75. Abrir ODBC Data Sources. [Autores]

Cree la conexión SQL server, de click en agregar

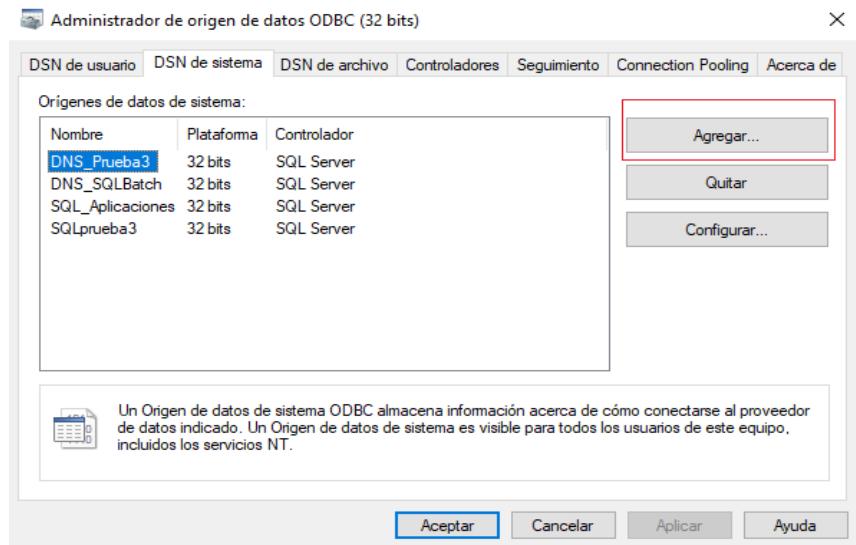


Figura 76. DSN Sistema

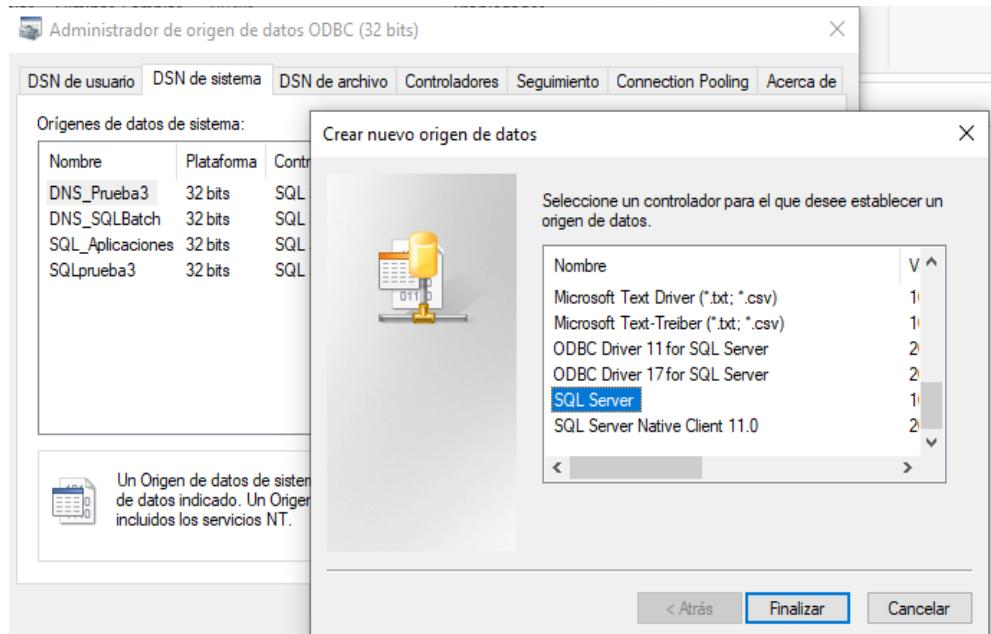


Figura 77. Agregar conexión SQL Server. [Autores]

Dele un nombre a la conexión, escoja el servidor para esto debe ir a SSMS y en la Figura 67 note que se encerró el nombre del servidor en un rectángulo rojo. Ese será su servidor, para evitar errores de transcripción deberá ir a SQL Server y da click derecho/Properties y copie el texto que aparece al frente de “NAME”. Luego de escribir el servidor en la Figura 78, dar siguiente.

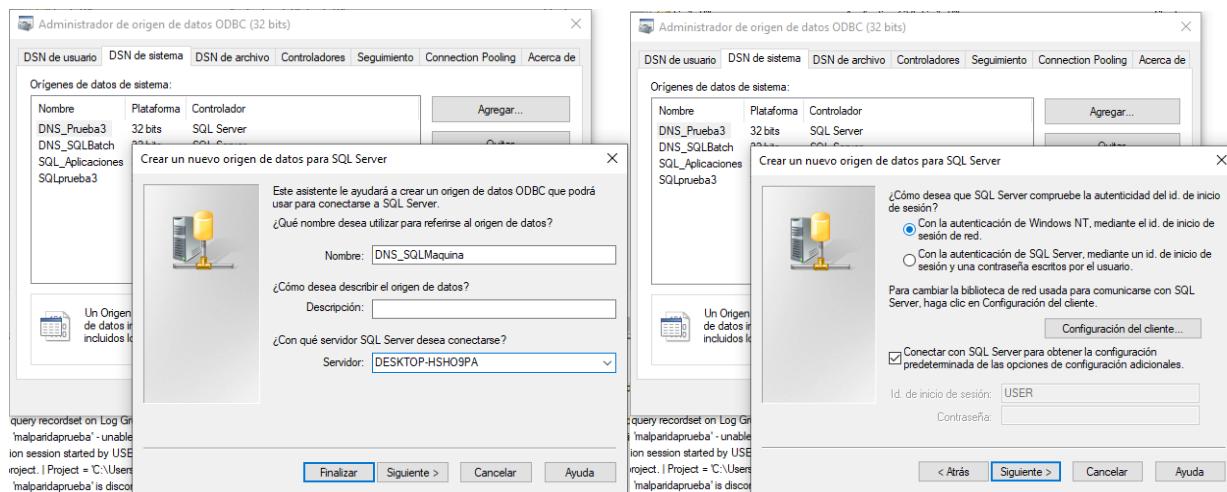


Figura 78. Nombre y autenticación. [Autores]

Como no tenemos un super usuario en autenticación seleccionamos la primera opción de la Figura 78. Ahora seleccione siguiente y establezca la Database “MaquinaConteo” por default, dar siguiente y aparecerá una ventana emergente donde “Probara el Origen de Datos” y si su conexión es correcta aparecerá un mensaje notificándolo (Figura 80).

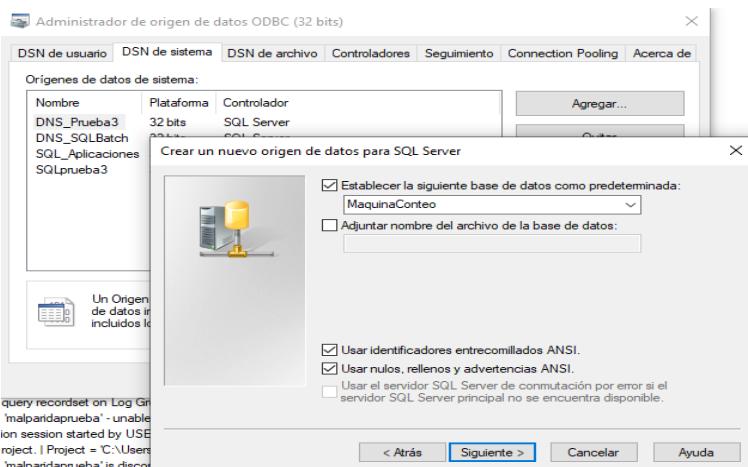


Figura 79. Seleccionar Database. [Autores]

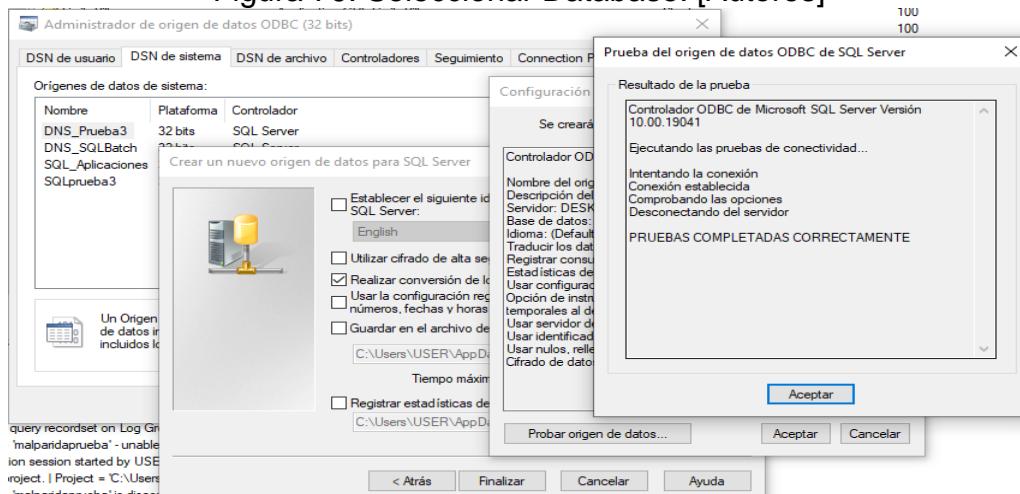


Figura 80. Probar ODBC. [Autores]

3. Intouch Crear BindList

Para la creación del BindList debe abrir Intouch, Ir a la parte izquierda inferior seleccionar “SQL Acces Manager” y crear un BindList. Debe aceptar la creación de SQL.DEF, ver Figura 81

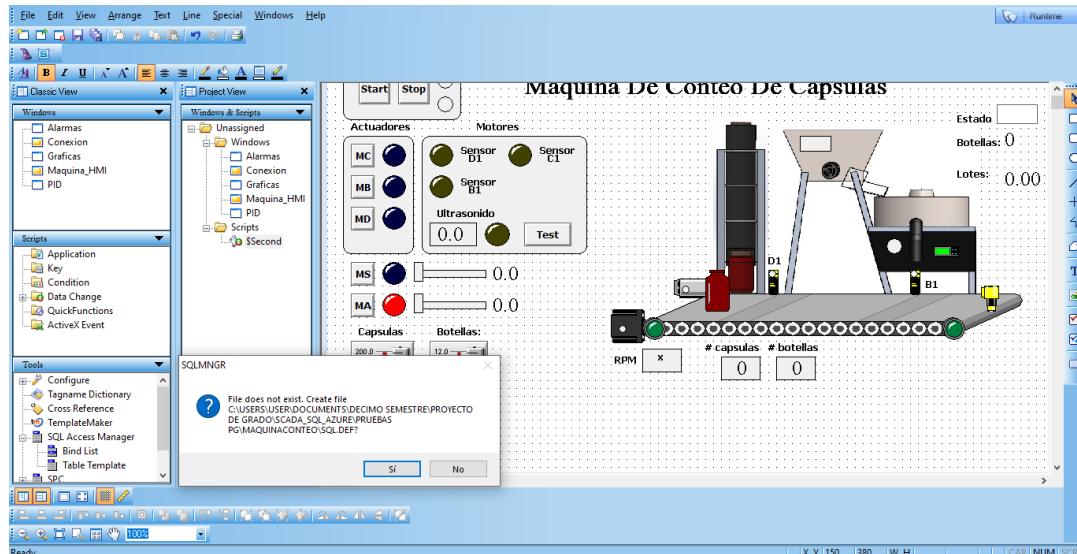


Figura 81. SCADA Intouch. [Autores]

Seleccione Nuevo BindList, dar un nombre y tenga en cuenta las variables de la Figura 73, Debe seleccionar en “Tagname” la variable que ya creo en Intouch y en ColumnName escriba el nombre de la columna de la tabla “datasetproceso”.

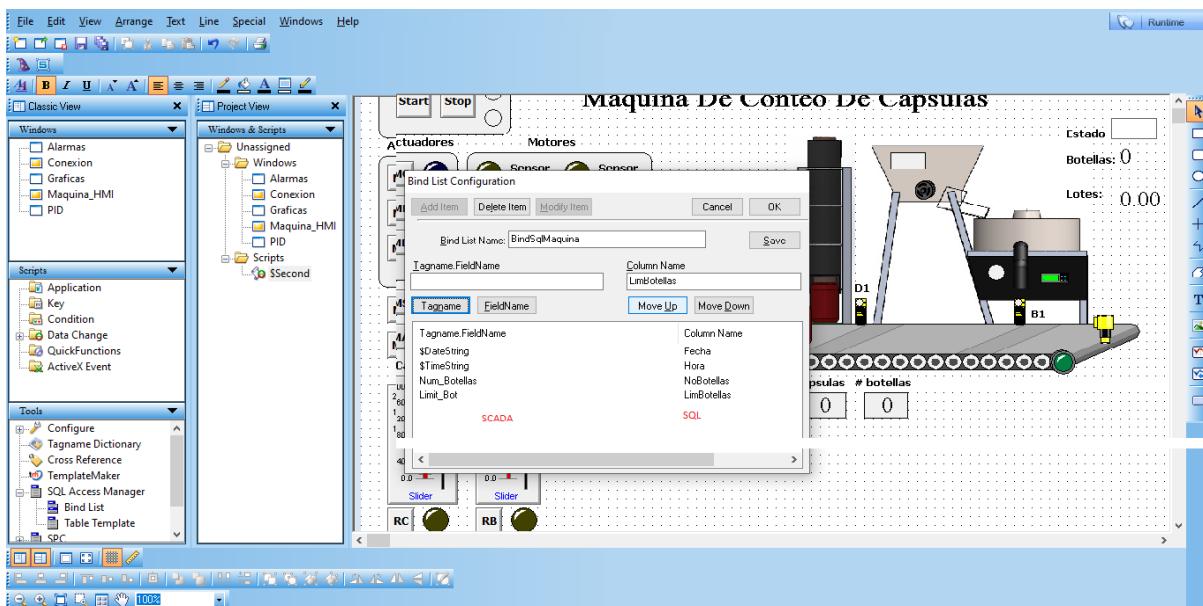


Figura 82. BindList. [Autores]

Luego de crear el BindList no puede crear más atributos por lo cuál debe premeditar bien que tipo de variables desea enviar por SQL. Por ultimo se debe crear el Script, para ello diríjase a la carpeta Scripts en Intouch de click derecho y seleccione “Data Change”, vea Figura 84 para mayor detalle.

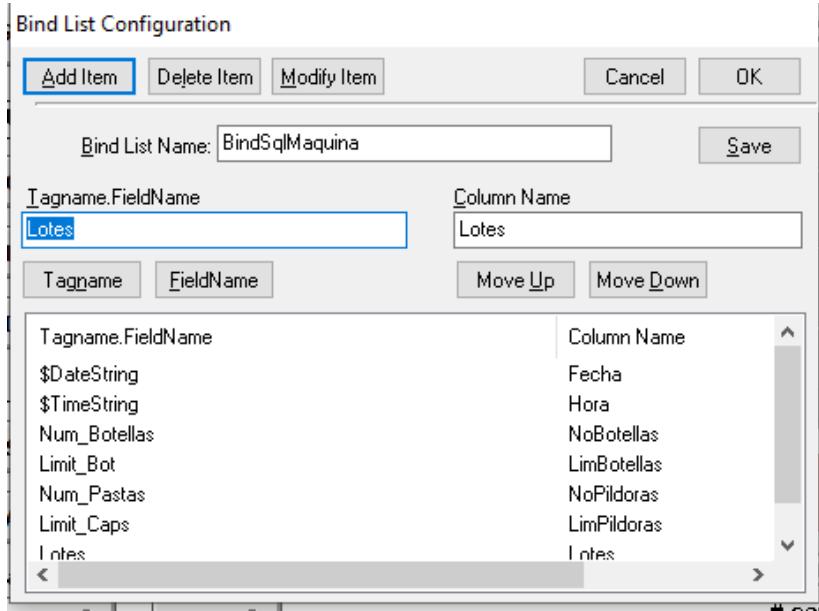


Figura 83. BindList tagname Lotes. [Autores]

4. Crear Script de Conexión

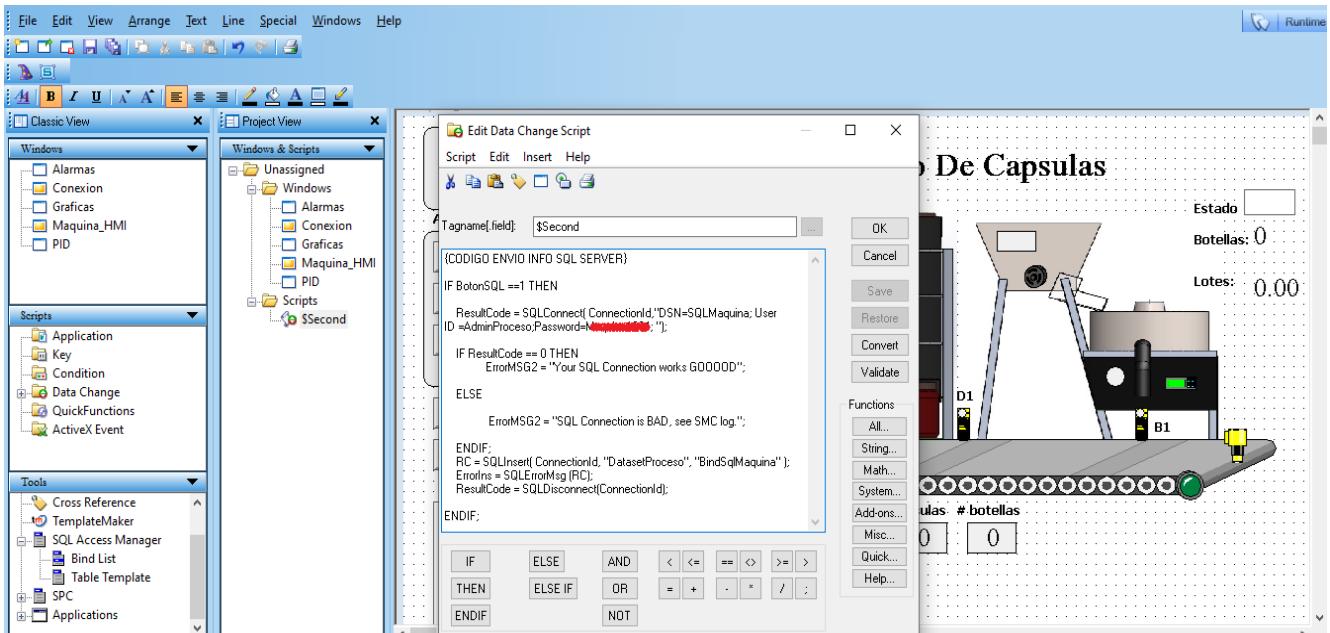


Figura 84. Script Data change. [Autores]

En la figura anterior se muestra el Script para enviar datos cada segundo, tenemos un botón que al habilitarse se conecta y necesita ciertos parámetros de conexión como DSN, UserID, Password.

Observe que primero se conecta Intouch con la base de datos, luego inserta los mismos buscando la tabla en específico y usando el BindList como conexión de variables. Defina las variables de la siguiente forma:

BotonSQ L	Memory Discrete
ErrorIns	Memory Message
ResultCo de	Memory Integer
Estado2	Memory Message
RC	

Tabla 4. Variables Data Change Script/Otras. [Autores]

Para más información consulte el documento de variables donde se listan las variables usadas en el proceso o vea el Anexo 1. Observe la interface básica para enviar datos a SQL y antes configure los parámetros como lo muestra la Tabla 5.

Atributo	Nombre	Descripción	Figura
SQLConnect (DSN)	SQLMaquina	Nombre del ODBC	Figura 78
SQLConnect (UserID)	AdminProceso	Usuario login para database	Figura 69
SQLConnect (Password)	Escriba su contraseña	PWD login para database	Figura 69
SQLInsert	DatasetProceso	Nombre de la tabla	Figura 74
SQLInsert	BindSqlMaquina	Nombre del Bind en Intouch	Figura 82

Tabla 5. Configurar Data Change. [Autores]

Por último, diseñe la siguiente interfaz básica, puede consultar el SCADA del proceso para saber qué tipo de elementos son (Botones, texto, labels)

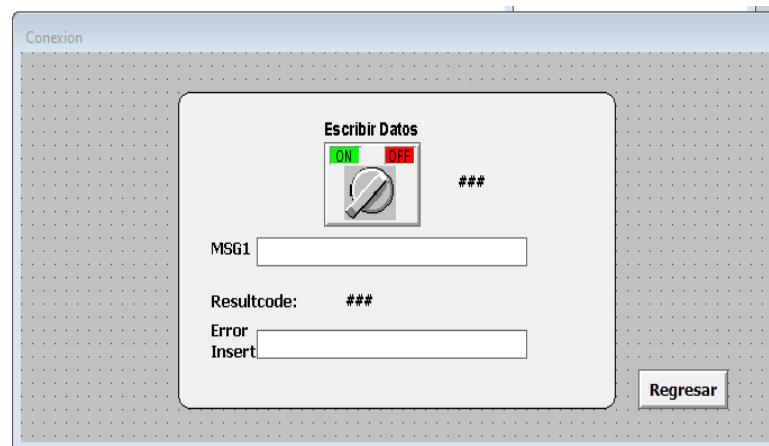


Figura 85. Interfaz conexión SQL. [Autores]

Presione Runtime para correr el SCADA, debe obturar el botón para enviar datos, para detener el envío de datos debe desobturar el botón.

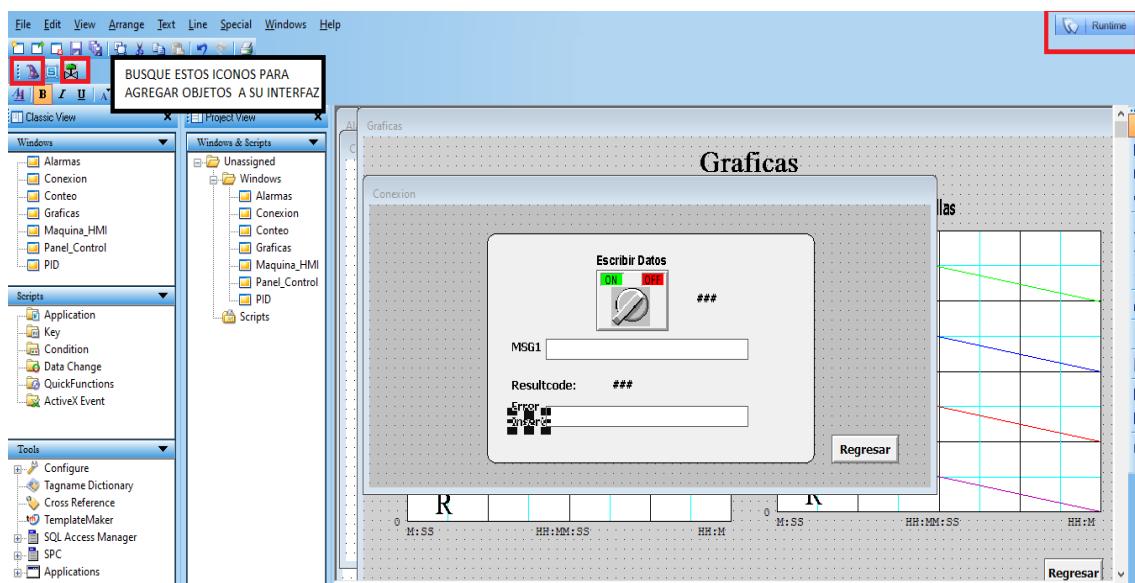


Figura 86. Runtime en Intouch. [Autores]

5. Comprobar el almacenamiento de datos On-Premise

Diríjase a SMSS busque su database, encuentre la tabla, de click derecho y haga una consulta de los registros (Select top 1000 rows), el cual mostrara las primeras 1000 instancias(filas) del proceso.

Results Messages

Datoid	Fecha	Hora	Start	Stop	NoBotellas	LimBotellas	NoPildoras	LimPildoras	Lotes	Bapa	SensorDispensador	SensorBanda	SensorCentrifuga	SensorUltrasonido	MotorAc
1	1002	21/09/2021	6:26:56 p.m	0	0	0	4	0	10	1	1	0	0	0	0
2	1003	21/09/2021	6:26:57 p.m	0	0	0	4	0	10	1	1	0	0	0	0
3	1004	21/09/2021	6:26:58 p.m	0	0	0	4	0	10	1	1	0	0	0	0
4	1005	21/09/2021	6:26:59 p.m	0	0	0	4	0	10	1	1	0	0	0	0
5	1006	21/09/2021	6:27:00 p.m	0	0	0	4	0	10	1	1	0	0	0	0
6	1007	21/09/2021	6:27:01 p.m	0	0	0	4	0	10	1	1	0	0	0	0
7	1008	21/09/2021	6:27:02 p.m	0	0	0	4	0	10	1	1	0	0	0	0
8	1009	21/09/2021	6:27:03 p.m	0	0	0	4	0	10	1	1	0	0	0	0
9	1010	21/09/2021	6:27:04 p.m	0	0	0	4	0	10	1	1	0	0	0	0
10	1011	21/09/2021	6:27:05 p.m	0	0	0	4	0	10	1	1	0	0	0	0

< > ^ v

Query executed successfully.

DESKTOP-HSH09PA (12.0 SP1) | DESKTOP-HSH09PA\USER (52) | MaquinaConteo | 00:00:02 | 211 rows

Figura 87. Registro de datos on premise con SQL Server. [Autores]

	<p>UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo</p> <p>ACTIVIDAD: Base de datos SQL on premise sincronizada con Azure a través de Data Sync</p>	
Código: PC005 <input type="checkbox"/> ONLINE <input checked="" type="checkbox"/> OFFLINE		Duración: ---

PC005. Base de datos SQL on premise sincronizada con Azure a través de Data Sync

Objetivo de la práctica:

Uno de los pilares de la Maquina contadora de pastas es la Computación en la Nube, la cual nos permite migrar la información síncrona o asíncronamente hacia nubes ya sean públicas o privadas al igual que contratar diferentes servicios. Es necesario que el estudiante comprenda la importancia de la nube y como migrar los datos en tiempo real a ella, para así dar más escalabilidad al proyecto y usar las herramientas de la industria 4.0 (conectar activos físicos con tecnologías digitales avanzadas)

Material Necesario y requisitos para el desarrollo:

- SQL Server Management Studio 2014 en adelante (Se usará v. 18)
- Cuenta en Azure Portal (Microsoft)
- Microsoft SQL Data Sync Agent 2.0

Esquema Grafico de la Actividad:

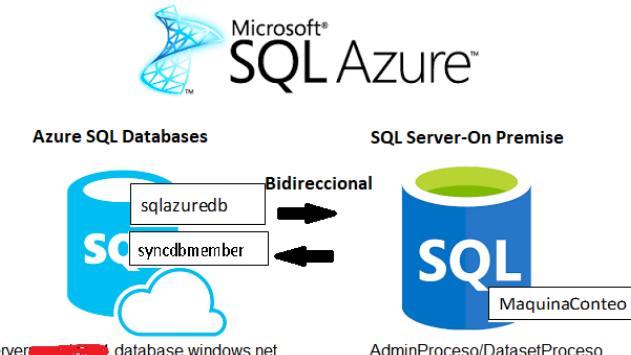


Figura 88. Esquema básico de la actividad a realizar. [Autores]

Requisitos previos:

- Reconocimiento Maquina de Conteo y SoftPLC
- Conexión entre Intouch-Codesys OPCUA (Practica #1)

- Importar/Exportar SCADA (Practica #2)
- Crear una base de Datos SQL Server (Practica 3)

Resumen:

Hay distintas opciones como Amazon, Azure, MongoDB, entre otras. Además, el Cloud Computing también trata el tema de servicios y proveedores; para mayor información lea el siguiente artículo:

<https://openwebinars.net/blog/tipos-de-cloud-computing/>



Figura 89. Algunas de las nubes tradicionales. [Autores]

Antes de escoger el tipo de base de datos que queremos es necesario entender brevemente que tipos de datos existen en la ciencia de datos:



Figura 90. SQL vs NoSQL. [Autores]

Los datos estructurados (SQL) son aquellos que se caracterizan por un identificador que relaciona los datos, usualmente se presenta en formato de tablas (donde cada columna es una variable y una fila una observación), se estima que las bases de datos relacionales mas frecuentes son las de Microsoft SQL Server.

Por otro lado, los datos semiestructurados (NoSQL), son datos que no están en campos fijos o tablas pero los datos son identificables; los formatos más comunes son:

- XML (Extensible Markup Language)
- JSON (JavaScript Object Notation)



Figura 91. Formato JSON. [Autores]

Luego de identificar que tipo de datos manejaremos, en nuestro caso Estructurados (SQL), es necesario escoger el proveedor del servicio de nube, Figura 89. Por ejemplo, MongoDB que es una nube que no es totalmente publica, pero si es versátil es usualmente trabaja para datos NoSQL.

Después de esta introducción sobre por qué usamos **Azure** y no otros proveedores, se trazara un bosquejo superficial de la practica:

Se creará un grupo de recursos que aloja distintos tipos de servicios en su suscripción de Azure, luego es necesario crear el servidor SQL Azure (<sunombre>.database.windows.net) donde se alojara una plataforma como servicio (PaaS) totalmente administrado que se encargara de realizar algunas tareas como actualizar, aplicar revisiones, crear copias de seguridad y supervisar. Luego de esto, se creará una regla en el servidor para poder acceder a el desde la IP que tenemos en la red; se creara una regla de salida para el puerto 1433 en el firewall del equipo, mientras tanto en Azure creamos una base de datos (sqlazuredb) y en las configuraciones de esta base de datos, se creara el grupo de sincronización, donde relacionaremos una nueva base de datos (syncdbmember) y la base On-premise a través de un agente, el agente permite hacer una conexión entre la base on premise y Azure usando un programa llamado M. SQL Data Sync (es gratuito y no presenta limitaciones), por ultimo asociamos nuestra tabla on premise a la tabla en Azure por medio de una interfaz intuitiva. Para no consumir créditos y gastar recursos en modo offline, debe eliminar el agente y la base de datos (syncdbmember).

Desarrollo de la Actividad:

1. Requisitos del sistema

SQL Data Sync es un servicio de Azure que le permite sincronizar datos entre varias bases de datos de Azure SQL y bases de datos locales de SQL Server, verifique los siguientes parámetros básicos:

- Abrir puerto TCP 1433 para que el cliente de la sincronización de datos (agente) se comunique con el servidor (se realiza en el apartado 3)
- Agregar IP publica en las reglas del servidor de Azure o no podrá acceder a él.
- Permitir el acceso de los servicios de Azure (importante, se hace en el apartado 2)
- La base de datos on premise **TIENE QUE** tener una llave primaria, revise la práctica “Conexión entre Intouch-SQL”

Tenga en cuenta las limitaciones como:

- Se debe agregar o quitar las columnas agregadas de más al sincronizar las tablas (On premise - Azure)
- Asegúrese de tener una versión .Net 4.5 o superior para el agente cliente del servidor

2. Crear Grupo de recursos y servidor SQL en Azure

En la siguiente figura se observa la interfaz principal de Azure, recuerde usar su suscripción de estudiante.

Figura 92. Azure Portal. [Autores]

Si quiere primero explore las diferentes opciones que nos ofrece Azure, de paso busque el servicio “Grupos de recursos”, aquí se alojara una serie de servicios que nosotros deseemos crear.

	Nombre ↑	Suscripción ↑↓	Ubicación ↑↓
<input type="checkbox"/> cloud-shell-storage-southcentralus	Azure para estudiantes	Azure para estudiantes	Centro-Sur de EE. UU.
<input type="checkbox"/> Pruebas1	Azure para estudiantes	Azure para estudiantes	Centro-Sur de EE. UU.

Figura 93. Crear Grupo de recursos. [Autores]

La creación del grupo es sencilla, solo debe configurar la ubicación y darle un nombre, deje la Ubicación que se observa en la Figura 94. Tenga en cuenta que para hacer sincronizaciones es recomendable que los servicios estén en la *misma ubicación*

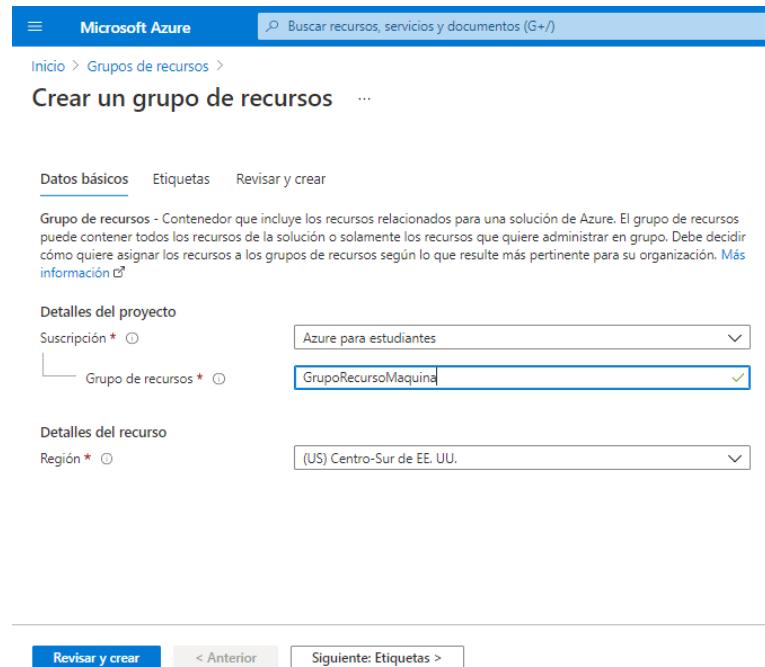


Figura 94. Configuración Grupo de Recursos. [Autores]

Observe a continuación, Azure siempre notifica cualquier cambio que realicemos en la plataforma, cuando creamos o eliminemos algún servicio.

The screenshot shows the 'Grupos de recursos' (Resource groups) page in the Microsoft Azure portal. It lists three resource groups: 'cloud-shell-storage-southcentralus', 'GrupoRecursoMaquina', and 'Pruebas1'. On the right, there is a notification message: 'Grupo de recursos creado' (Resource group created) with the note: 'La creación del grupo de recursos "GrupoRecursoMaquina" en la suscripción "Azure para estudiantes" se realizó correctamente.' (The creation of the resource group 'GrupoRecursoMaquina' in the subscription 'Azure para estudiantes' was completed successfully.) Below the notification are buttons for 'Ir al grupo de recursos' (Go to resource group) and 'Anclar al panel' (Pin to dashboard).

Figura 95.Creacion del Grupo de Recursos. [Autores]

Para alojar nuestras bases de datos es necesario crear un servidor, para ello primero debemos buscar el servicio, ver Figura 96. En la creación del servidor tenga en cuenta la Ubicación y permitir que los demás servicios de Azure accedan a él, de lo contrario la sincronización no será exitosa, ver Figura 97. Un resumen de la configuración suministrada para la creación del servidor se observa en la Figura 98.

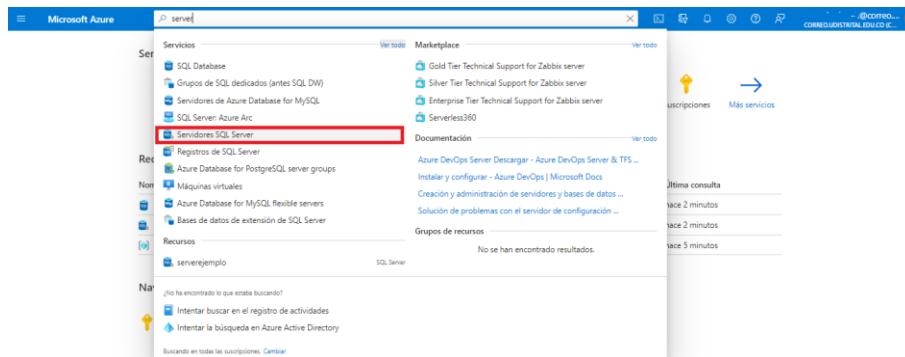


Figura 96. Buscar Servicio SQL Server. [Autores]

Observe el resumen de la creación del servidor en la Figura 98, asigne un nombre al servidor y asócielo al grupo de recursos “GrupoRecursoMaquina”

Figura 97. Crear servidor en Azure [Autores]

Recuerde el nombre y password del usuario que creo para el servidor

Figura 98. Configuración Server. [Autores]

Cuando cree el servidor, búsquelo en su Inicio y ábralos, vera la misma información, note los parámetros encerrados en cajas:

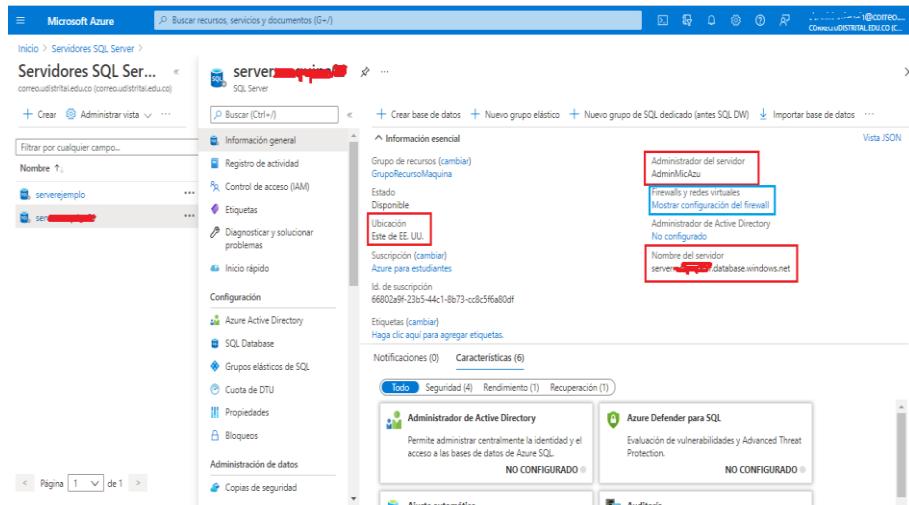


Figura 99. Servidor creado. [Autores]

De aquí en adelante usaremos el nombre del servidor “<nombre>.database.windows.net”, de aquí en adelante usaremos la ubicación Este de EE.UU, recuerde los datos que coloco para su usuario y por último en el recuadro azul podemos configurar el firewall del servidor, ¿le gustaría que cualquiera entrara a su servidor? por este motivo se debe configurar las reglas del mismo.

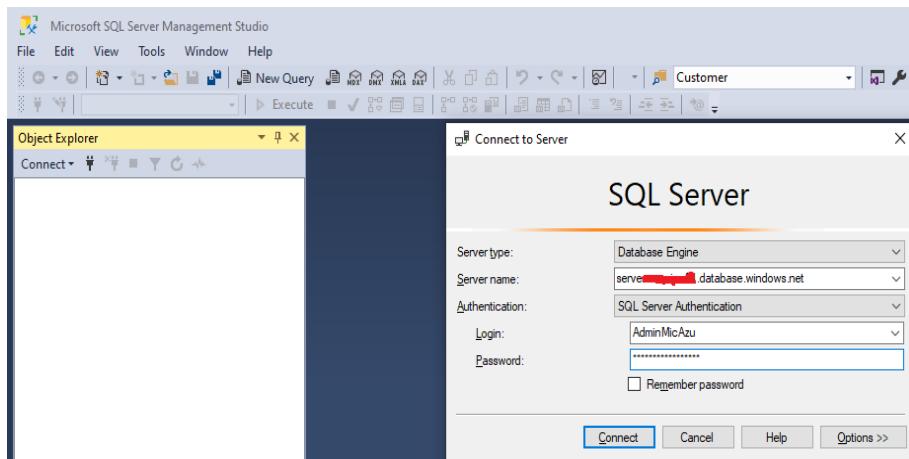


Figura 100. Conectar el servidor de Azure, parte 1. [Autores]

Al no crear la regla en el servidor aparece el siguiente mensaje y no tendremos acceso al servidor.

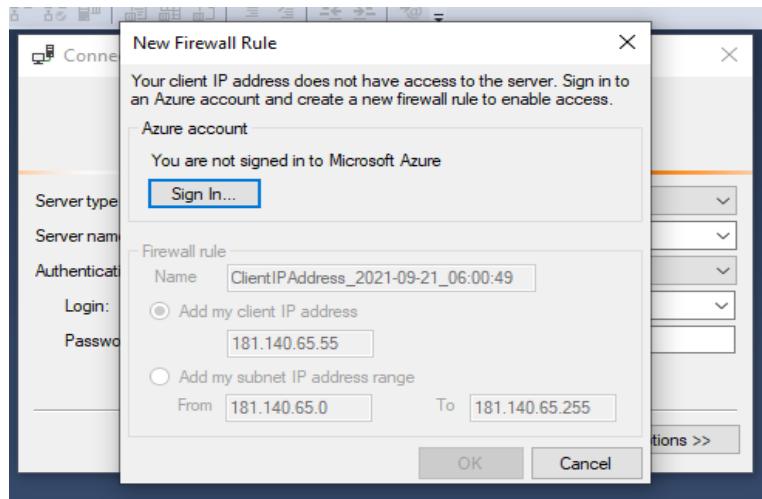


Figura 101. Conectar el servidor de Azure, parte 2. [Autores]

Regrese a Azure, vuelva a su servidor y en las configuraciones del mismo, busque la opción “Firewalls y redes virtuales”, debe crear la regla para la IP que posee su equipo.

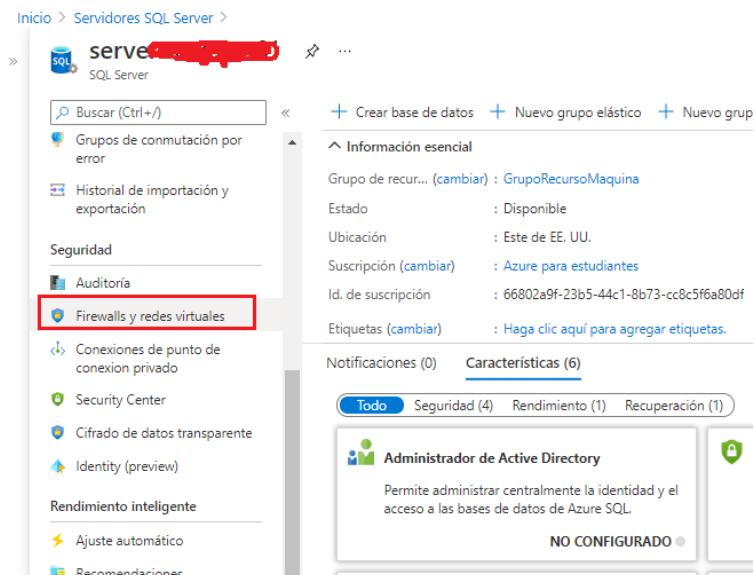


Figura 102. Creación de la regla en Azure. [Autores]

Debe consultar su dirección IP publica; ingrese a este link y obtenga su dirección IPV4 <https://whatismyipaddress.com/> y configura la regla como se muestra en la Figura 103

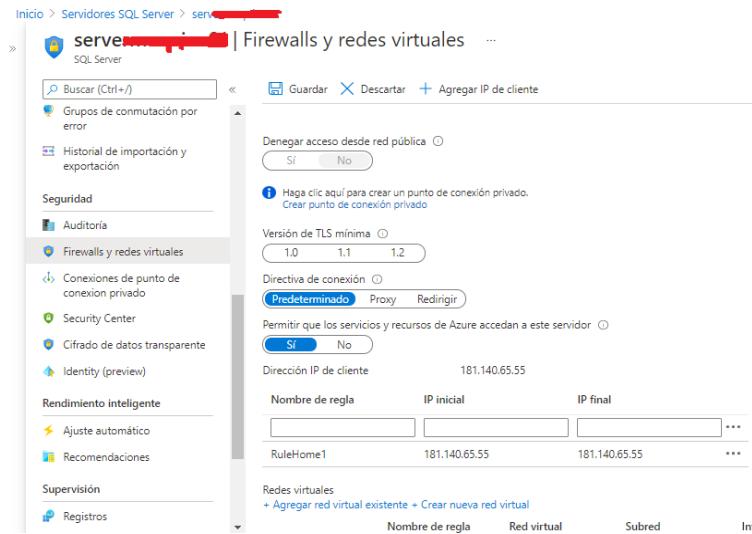


Figura 103. Creación de la regla “RuleHome1”. [Autores]

Intente de nuevo acceder al servidor desde SSMS como en la Figura 100 y si todo está bien realizado debe acceder al servidor como se muestra en la siguiente figura.

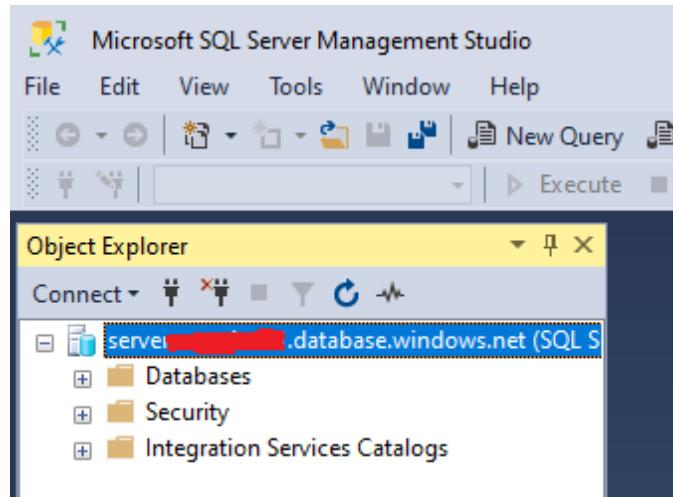


Figura 104. Acceder desde SMSS al servidor en Azure. [Autores]

3. Crear regla en Firewall y base de datos en Azure

A continuación, crearemos la regla TCP 1433 en nuestro computador para que el cliente de sincronización se pueda comunicar por dicho puerto con el servidor en Azure a través del agente, escriba el siguiente comando:

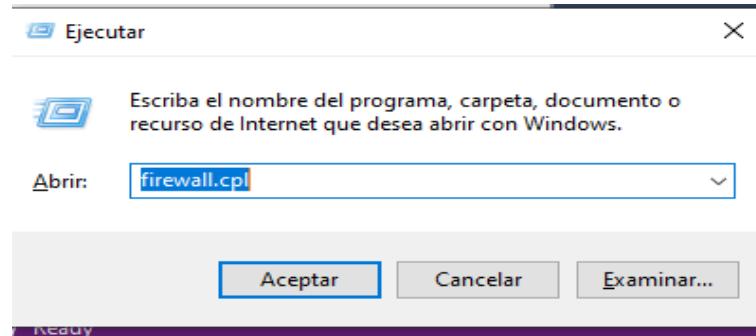


Figura 105. Win+R para abrir la venta

Crear la nueva regla, como observa en la figura siguiente la regla ya está creada, aunque en las próximas figuras se aprecian los parámetros.

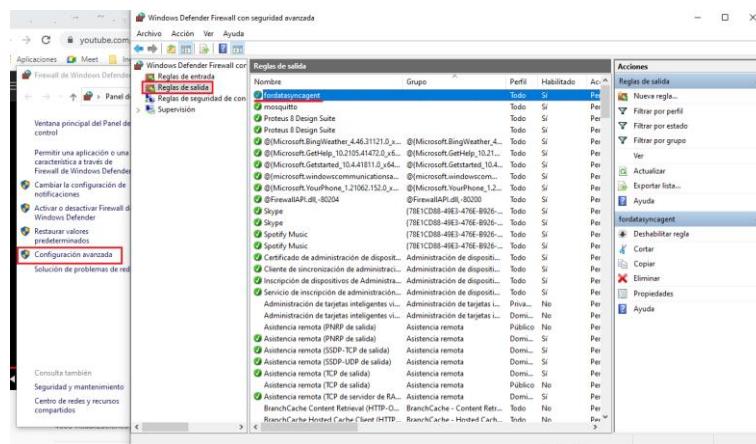


Figura 106. Regla de salida. [Autores]

Configure de la siguiente manera:

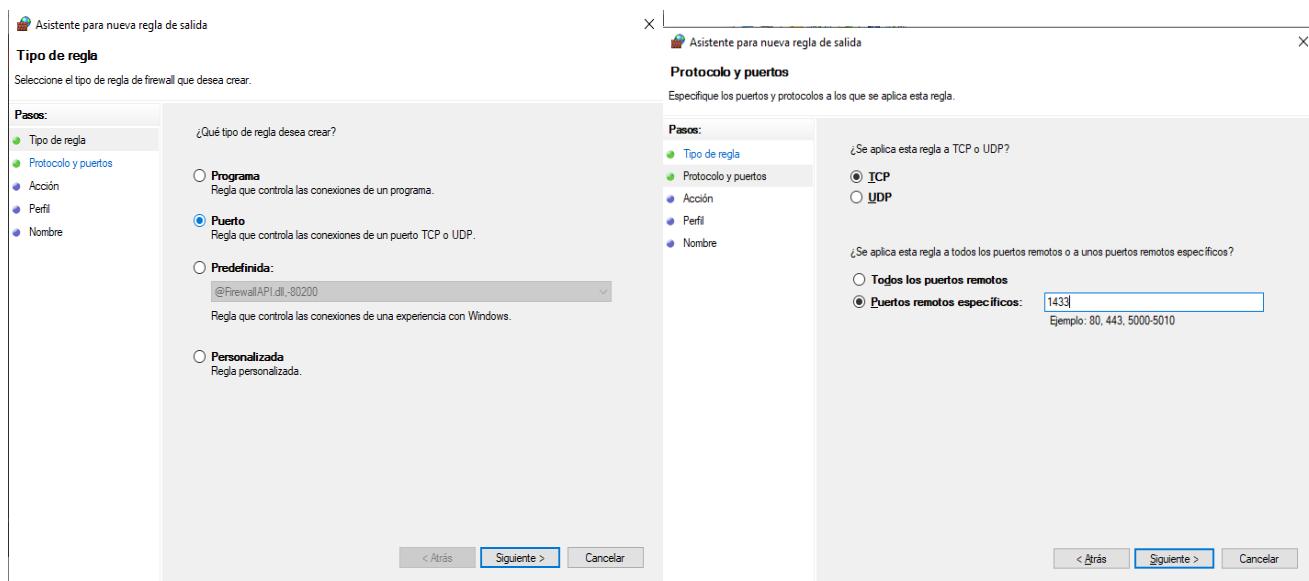


Figura 107. Regla de salida, paso 1. [Autores]

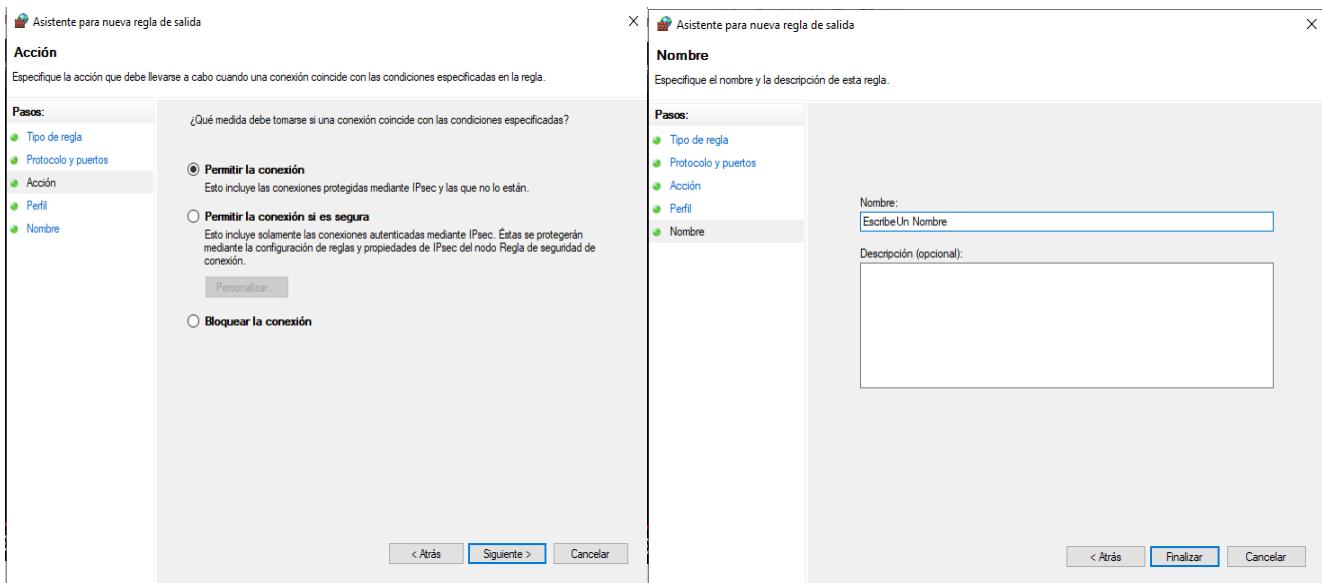


Figura 108. Regla de salida, paso 2. [Autores]

En este punto ya hemos creado el servidor en Azure y la regla del cliente de sincronización, ahora procedemos a crear la base de datos en Azure y dentro de ella crearemos el grupo de sincronización de datos.

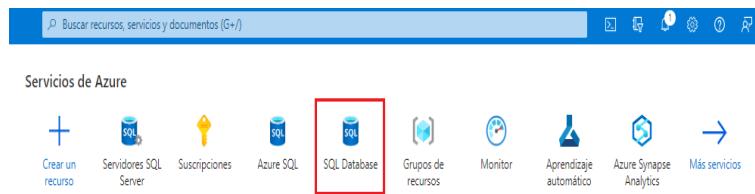


Figura 109. Buscar servicio SQL Database. [Autores]

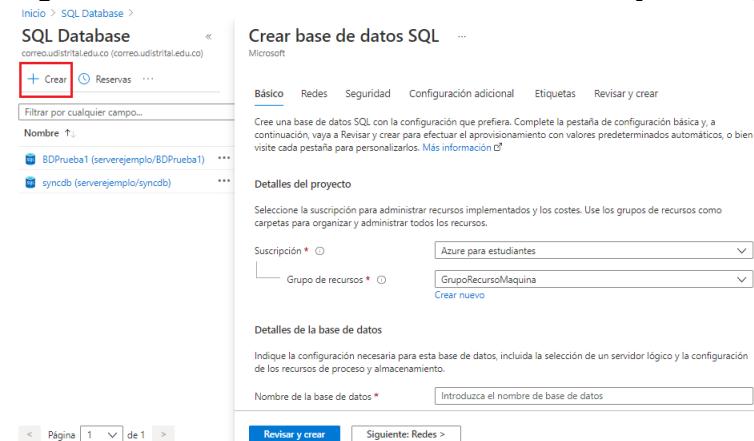


Figura 110. Crear base de datos sqlazuredb. [Autores]

En la Figura 111 se aprecia unos de los aspectos a tener en cuenta en los servicios de Azure, ¿Qué necesito hacer y estoy dispuesto a pagar?, debe modificar esta opción ya que nuestra dataset no es robusto ni complejo a diferencia de otros datasets.

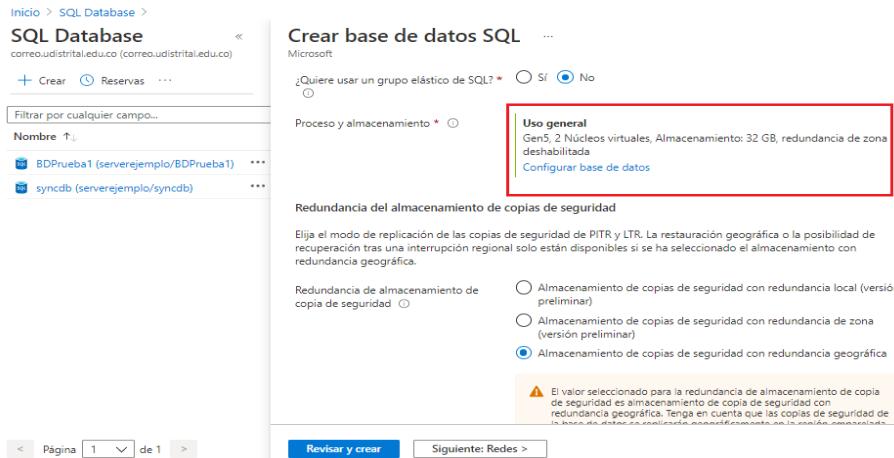


Figura 111. Cambiar opciones de almacenamiento DB. [Autores]

Seleccione el plan básico:

Resumen del costo	
Costo por DTU (en USD)	1.00
Selección DTU	x 5
COSTO MENSUAL ESTIMADO	4.99 USD

Figura 112. Plan Básico para bases de datos. [Autores]

En la Figura 113 se muestra el resumen, use el grupo de recursos y el servidor creados anteriormente

Costo estimado al mes	
4.99 USD	Ver detalles de precio

Figura 113. Configuración resumen sqlazuredb. [Autores]

Microsoft.SQLDatabase.newDatabaseExistingServer_52415972a7734284 | Información general

Nombre de implementación: Microsoft.SQLDatabase.newDatabase... Hora de inicio: 21/9/2021 13:21:27
Suscripción: Azure para estudiantes Id. de correlación: 37bc337e-7654-4dc6-acf6-4de1b738383a
Grupo de recursos: GrupoRecursoMaquina

Recurso	Tipo	Estado	Detalles de la operación
servermaquina01/sqlazuredb	Microsoft.Sql/servers/databases	Accepted	Detalles de la operación
server [REDACTED]	Microsoft.Sql/servers	OK	Detalles de la operación

Figura 114. Implementación base de datos en curso. [Autores]

Ya que creamos la base de datos sqlazuredb en el servidor <nombre>.database.windows.net; iremos a SSMS y refrescaremos el servidor(Figura 116) pero antes conéctese al server local (Figura 115)

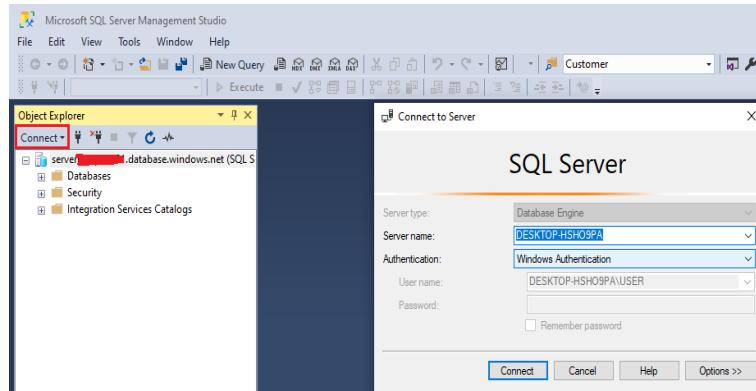


Figura 115. Conectar servidor local. [Autores]

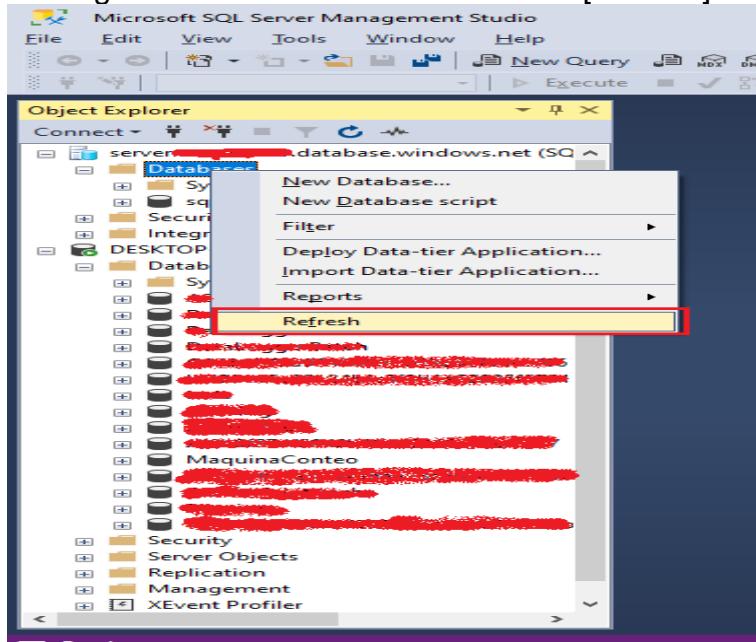


Figura 116. Refrescar servidor Azure. [Autores]

4. Crear un grupo de Sincronización

Ahora crearemos el grupo de sincronización en la base de datos del apartado anterior (sqlazuredb), vaya a Inicio y abra la BD, puede notar las diferentes opciones que tiene Azure para interactuar con cada servicio contratado. Diríjase a la sección de “Sincronizar con otras bases de Datos” y cree un nuevo grupo de sincronización.

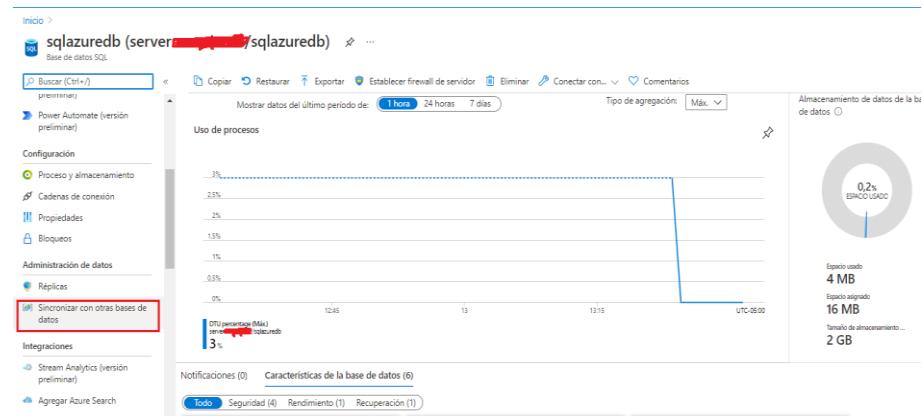


Figura 117. Sqlazuredb propiedades [Autores]

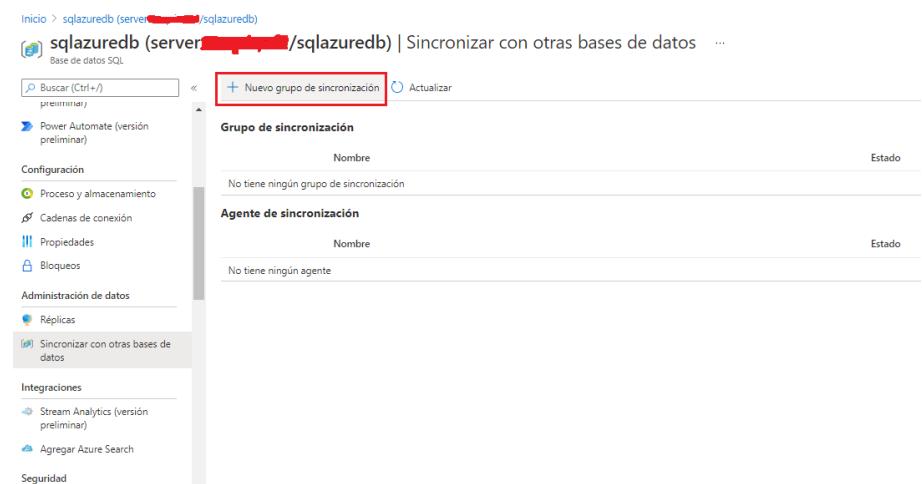


Figura 118. Crear grupo de sincronización. [Autores]

De un nombre al grupo de sincronización, seleccione “Base de datos nueva” ya que es lo recomendable porque Azure empezara a crear nuevos esquemas y configuraciones de sincronización; la frecuencia de sintonización se dejó en 5 minutos, pero eso depende de que tan frecuente quiere actualizar los datos en Azure (automáticamente), sin embargo también debe saber que en Azure hay un botón para hacer la petición y actualización de datos en cualquier momento sin tener que esperar el tiempo configurado

Crear grupo de Data Sync

Cree un grupo de sincronización en la base de datos central. Las bases de datos miembros y otros valores adicionales pueden configurarse después de la creación en la hoja de detalles del grupo de sincronización. [Más información](#)

Nombre del grupo de sincronización *

Azure_syncGroup

Base de datos de metadatos de sincronización

Usar base de datos existente Base de datos nueva

Crear nueva base de datos *

[Configuración de las opciones de la base de datos](#)

Sincronización automática *

Activado Desactivado

Frecuencia de sincronización *

5

Minutos

Resolución de conflictos *

Win de centro de conectividad

Usar Private Link

[Aceptar](#)

Inicio > sqlazuredb (server.../sqlazuredb) > Crear grupo de Data Sync >

Crear grupo de Da...

SQL Database

Nombre *

Introduzca el nombre de base de datos

Grupo de recursos *

GrupoRecursoMaquina

[Crear nuevo](#)

Servidor de destino *

server... (Este de EE. UU.)

[Seleccionar servidor](#)

Plan de tarifa *

[Configurar base de datos](#)

Intercalación *

SQL_Latin1_General_CI_AS

Redundancia de almacenamiento de copia de seguridad

Almacenamiento de copias de seguridad con redundancia local (versión preliminar)

Almacenamiento de copias de seguridad con redundancia de zona (versión

[Aceptar](#)

Figura 119. Crear grupo de sincronización parte 1. [Autores]

En la figura anterior estamos creando la nueva base de datos llamada syncdbmember, recuerde configurarla igual a la base de datos sqlazuredb, observe las siguientes figuras.

Microsoft Azure Buscar recursos, servicios y documentos (G+ /)

Inicio > sqlazuredb (server.../sqlazuredb) > Crear grupo de Data Sync >

Crear grupo de Da... SQL Database

Cree un grupo de sincronización en la base de d... otros valores adicionales pueden configurarse de... grupo de sincronización. [Más información](#)

Nombre del grupo de sincronización *

Azure_syncGroup

Base de datos de metadatos de sincronización

Usar base de datos existente Base de d...

Crear nueva base de datos *

[Configuración de las opciones de la base de da...](#)

Sincronización automática *

Activado Desactivado

Frecuencia de sincronización *

5

Resolución de conflictos *

Win de centro de conectividad

Usar Private Link

[Aceptar](#) [Aceptar](#)

Notificaciones

Más eventos en el registro de actividad → Descartar todo

- Creación de Azure.syncGroup Se creó el grupo de sincronización Azure.syncGroup hace unos segundos
- Implementación correcta La implementación "Microsoft.SqlDatabase.newDatabaseExistingServer_1734a3af0d13421e" se realizó correctamente en el grupo de recursos "GrupoRecursoMaquina". hace unos segundos

Servicios de Azure

- [Crear un recurso](#) [SQL Database](#) [Servidores SQL Server](#) [Suscripciones](#) [Azure SQL](#) [Grupos de recursos](#) [Monitor](#)

Recursos recientes

Nombre	Tipo
sqlazuredb (server.../sqlazuredb)	Base de datos SQL
GrupoRecursoMaquina	Grupo de recursos

Figura 120. Creación grupo de sincronización parte 1. [Autores]

Al terminar la creación de la nueva base de datos, diríjase de nuevo a la sincronización con otras bases de datos y cliquee en el Nombre “Azure_syncGroup”, porque vamos a proceder a asignar las dos bases de datos y la tabla.

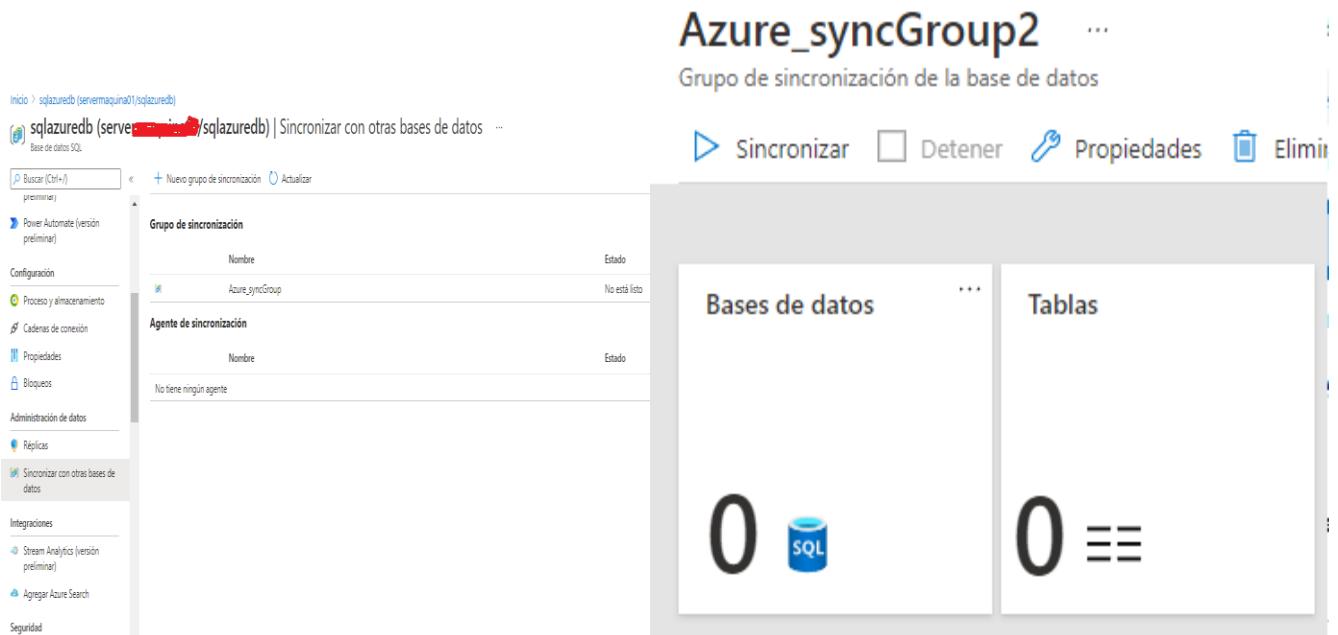


Figura 121. Configuración grupo de sincronización. [Autores]

En la Figura 122 se observa como se asignó la base de datos exitosamente, pero en la Figura 123 se aprecia como debió ser configurada. Para ello debes darle click en bases de datos (Figura 121) y de cero SQL pasara a 1 DataBase asignada.



Figura 122. Primera base de datos asignada. [Autores]

Debes escribir el servidor, tal cual aparece en Azure con el usuario y contraseña creadas en su momento (apartado 2)

Este campo es obligatorio.
La contraseña debe contener como mínimo 8 caracteres.
Su contraseña debe contener caracteres de tres de las siguientes categorías – Letras en mayúsculas del alfabeto inglés, letras en minúscula del alfabeto inglés, números (0-9), y caracteres no alfanuméricos (!, \$, #, %, etc.).

Aceptar

Figura 123. Configuración previa. [Autores]

Ahora se procederá a configurar la base de datos on premise, todo el apartado 4 fue la configuración de la base de datos de sincronización en Azure.

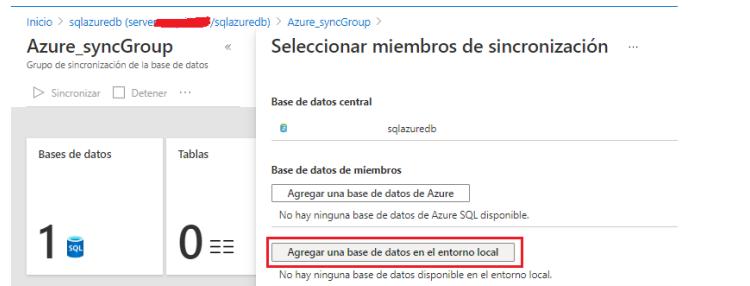


Figura 124. Configuración On-premise. [Autores]

5. Configurar Agente

Ahora se descargara y se instalara el programa del agente (Microsoft SQL Data Sync Agent 2.0)

1 Descargar agente de sincronización de cliente
Es necesario instalar el cliente del agente de sincronización para permitir que la base de datos local se conecte a la base de datos de Azure.
[Descargar](#)

2 Nombre del agente *

3 Generar una clave de agente
Use esta clave en el agente de sincronización instalado para registrar este agente.

Nombre	Fecha de modificación	Tipo	Tamaño
debug	2/09/2021 12:19 a. m.	Documento de texto	1 KB
B09fueda1	2/09/2021 12:09 a. m.	Microsoft Power BI Desktop D...	1 KB
Data Sync	1/09/2021 9:30 p. m.	Documento de Microsoft Word	73 KB
win00005.win	2/09/2021 3:07 a. m.	Archivo WIN	80 KB
pdf-clues.compress	8/09/2021 7:10 p. m.	Adobe Acrobat Document	100 KB
Consulta 2.json	2/09/2021 4:42 p. m.	Archivo de valores separados ...	106 KB
Consulta 2.json	2/09/2021 4:42 p. m.	Adobe JSON	215 KB
SyncAgent.LicenseTerms	1/09/2021 9:30 p. m.	Formato de texto enriquecido	255 KB
PalabraIdentificaciónConModelosDiscretosParaSistemasLinea-4797338	4/09/2021 12:52 a. m.	Adobe Acrobat Document	440 KB
GP-10101 (1)	2/09/2021 2:02 a. m.	Adobe Acrobat Document	440 KB
GP-10101 (2)	2/09/2021 2:02 a. m.	Adobe Acrobat Document	440 KB
GP-10101	2/09/2021 1:54 a. m.	Adobe Acrobat Document	440 KB
PDFOffice	3/09/2021 9:51 p. m.	Adobe Acrobat Document	601 KB
PDFOffice.com_control-pid-digital-para-banda-transportadora-5-pdf-free (1)	3/09/2021 6:51 p. m.	Adobe Acrobat Document	815 KB
PDFOffice.com_control-pid-digital-para-banda-transportadora-5-pdf-free (2)	3/09/2021 6:27 p. m.	Adobe Acrobat Document	815 KB
SQLDataSyncAgent-2.0-v14-ENU	1/09/2021 9:26 p. m.	Paquete de Windows Installer	10,300 KB

Figura 125. Descargar SQL Data Sync. [Autores]

En la Figura 126 se instala el programa, tenga en cuenta que debe saber el usuario actual de su PC y la contraseña; tenga cuidado porque depende de cómo usted tenga configurado ese usuario la contraseña puede ser el PIN de inicio de sesión o si tiene su usuario asociado a Outlook es la contraseña del correo. Si no sabes tu usuario actual, mirar Figura 127

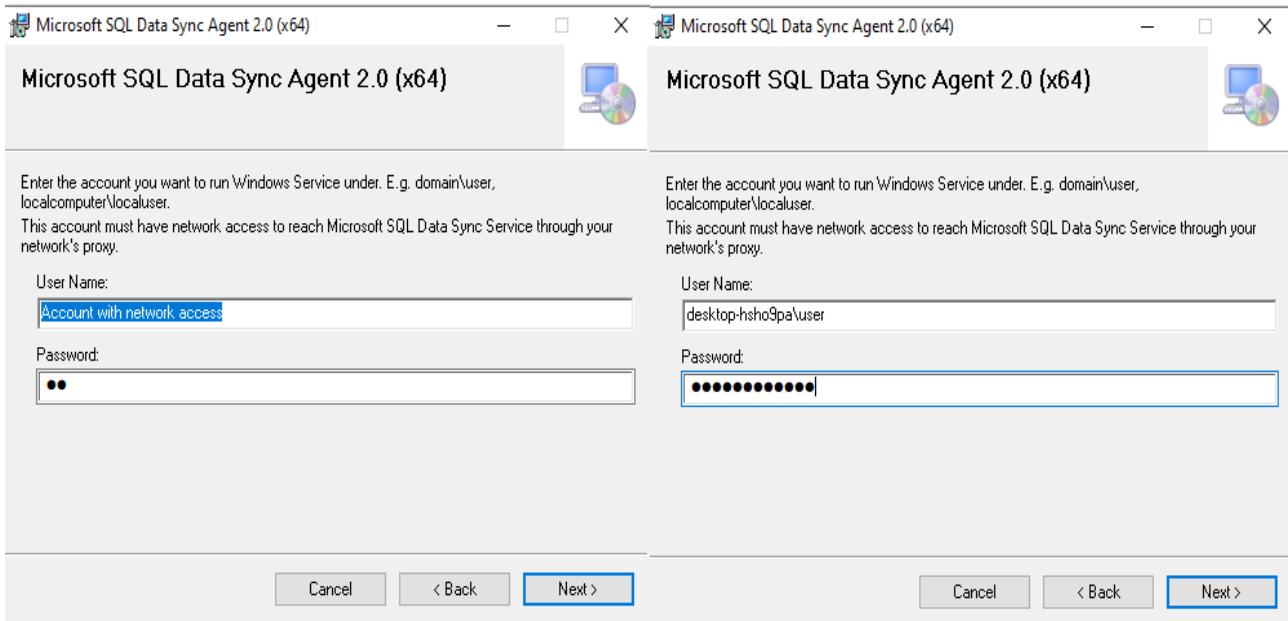


Figura 126. Instalar Microsoft SQL Data Sync Agent 2.0. [Autores]

Si todo sale bien, el programa empezara a instalar (no demora mucho) y no saldrá ningún error en medio de la instalación.

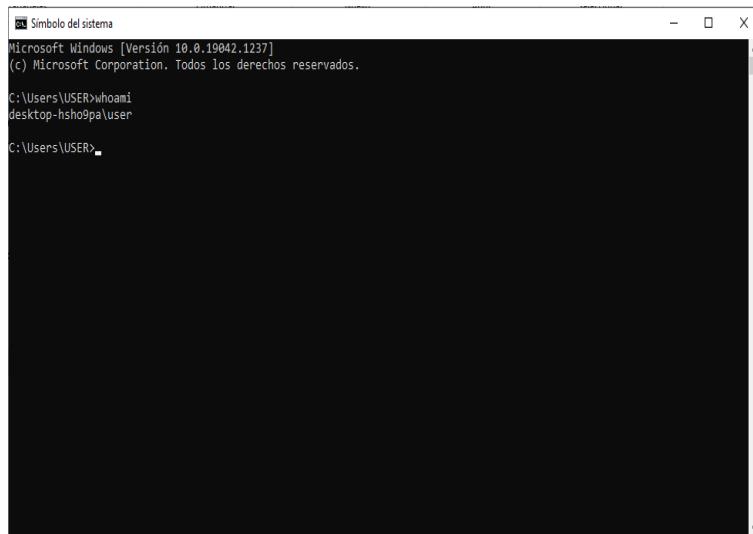


Figura 127. ¿Quién soy yo?. [Autores]

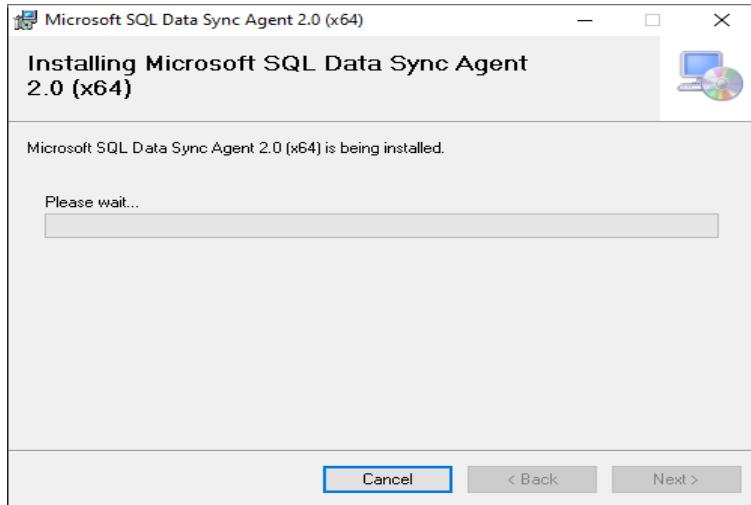


Figura 128. Instalación exitosa de Microsoft SQL Data Sync Agent. [Autores]
Luego de instalar el programa, siga en Azure y cree/genere la clave de conexión con el Agente (programa anterior)

Microsoft Azure

Buscar recursos, servicios y documentos (G+/)

Inicio > sqlazuredb (server: [REDACTED]/sqlazuredb) > Azure_syncGroup > Seleccionar miem

Seleccionar agente de sincronización

Agentes existentes Crear un agente nuevo

1 Descargar agente de sincronización de cliente
Es necesario instalar el cliente del agente de sincronización para permitir que la base de datos local se conecte a la base de datos de Azure.
[Descargar](#)

2 Nombre del agente
AgenteMaquina
[Crear y generar clave](#)

3 Generar una clave de agente
Use esta clave en el agente de sincronización instalado para registrar este agente.

Abra el programa

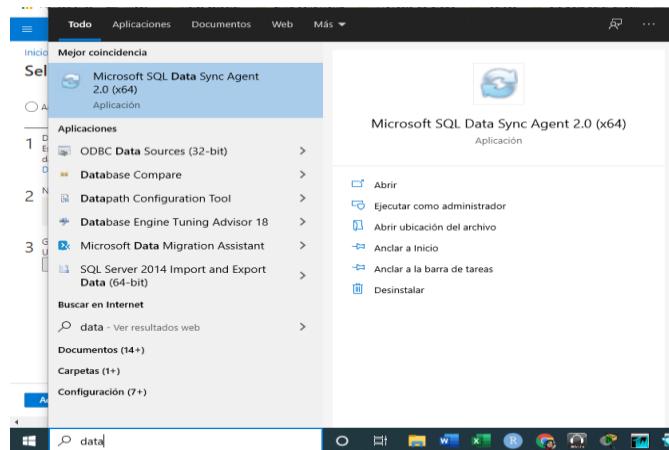


Figura 130. Abrir Microsoft SQL Data Sync Agent 2.0. [Autores]

Observe que al generar la llave en Azure (Figura 129), se debe copiar y pegar en el programa Data Sync 2.0, el Login y Password serán los que colocamos en el apartado 2 al momento de crear el servidor en Azure, al dar “Test Connection” debe aparecer el mensaje que se muestra en la Figura 131

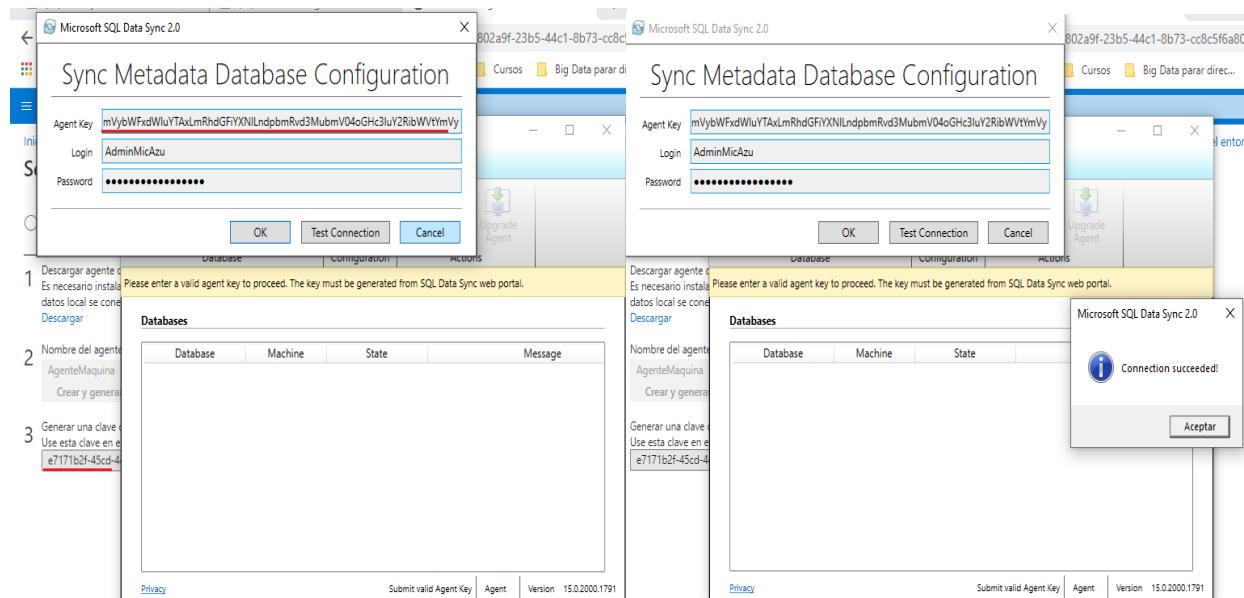


Figura 131. Microsoft SQL Data Sync Agent 2.0 configurar llave. [Autores]

En la figura siguiente se observa la creación de un nuevo registrador, en esta parte estamos asociando directamente la base de datos Onpremise (**Obligatorio** la práctica **PC004. Crear una base de Datos SQL Server**) al agente y este se encargará a su vez de conectarse con Azure.

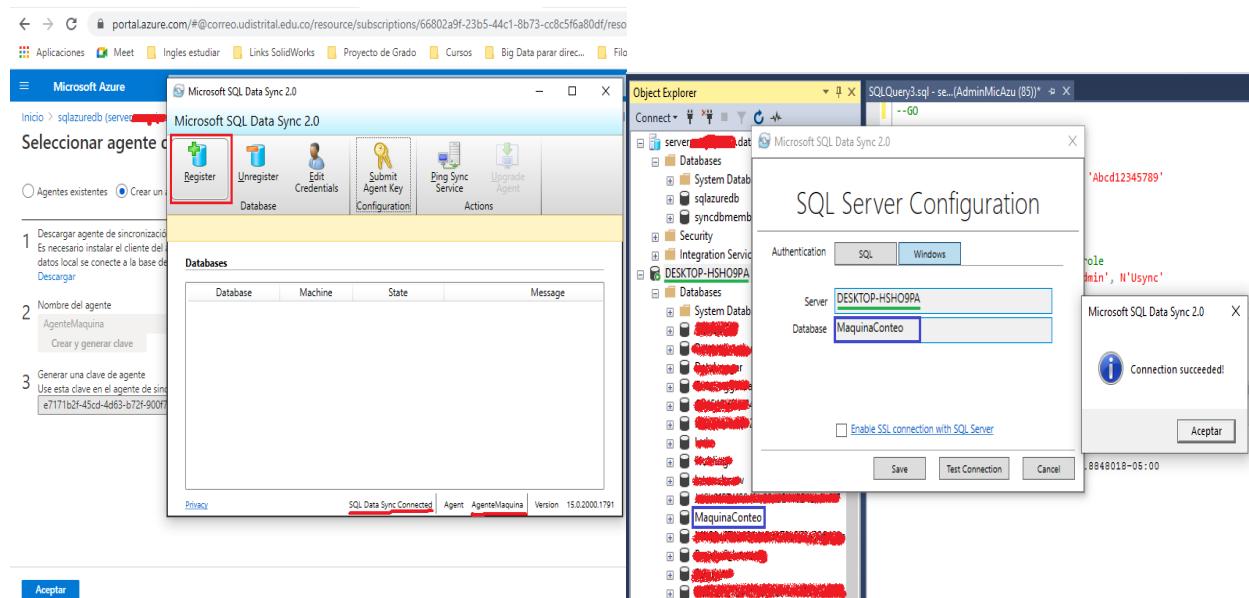


Figura 132. Registrar la Base de datos On-Premise. [Autores]

En la figura anterior nótese que la autenticación es con Windows (nuestro equipo local no tiene super usuario en SQL Server) pero si usted tiene un super usuario debe loggearlo para

que se conecte a su servidor local (OnPremise), pruebe la conexión y debe ser exitosa. Realice el Ping Sync como se muestra en la Figura 133.

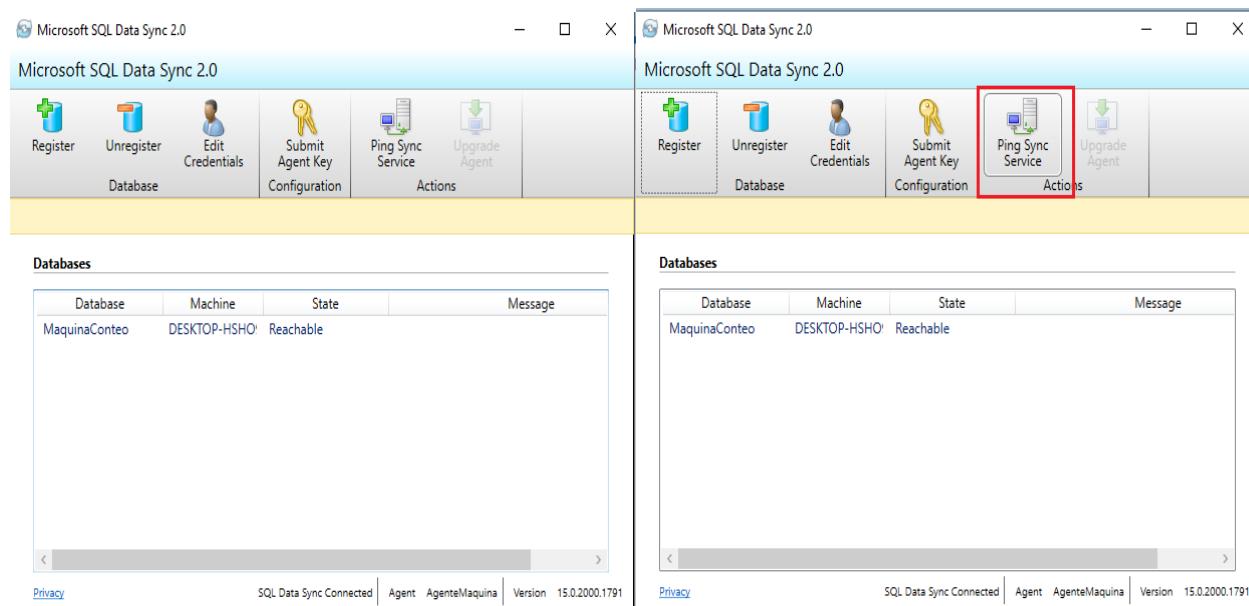


Figura 133. Se registro correctamente la base de datos OnPremise. [Autores]

La imagen es una captura de pantalla del portal de Azure. En la parte superior, se muestra la ruta 'Inicio > sqlazuredb (server... /sqlazuredb)'. El panel central muestra la configuración de sincronización:

- Grupo de sincronización:** Muestra un grupo llamado 'Azure_syncGroup' que aún no está listo.
- Agente de sincronización:** Muestra un agente llamado 'AgenteMaquina' que está conectado.

A la izquierda, hay un menú lateral con las siguientes opciones:

- Administración de datos
- Réplicas
- Sincronizar con otras bases de datos (selecctionada)
- Integraciones
- Stream Analytics (versión preliminar)
- Agregar Azure Search
- Seguridad
- Auditoría
- Libro de contabilidad
- Clasificación y detección de datos
- Enmascaramiento dinámico de datos
- Security Center
- Cifrado de datos transparente

Figura 134. Verificar estado del agente en Azure. [Autores]

Después de configurar el agente se debe seleccionar el registro creado en la Figura 133 dando click en “Seleccionar la base de datos”

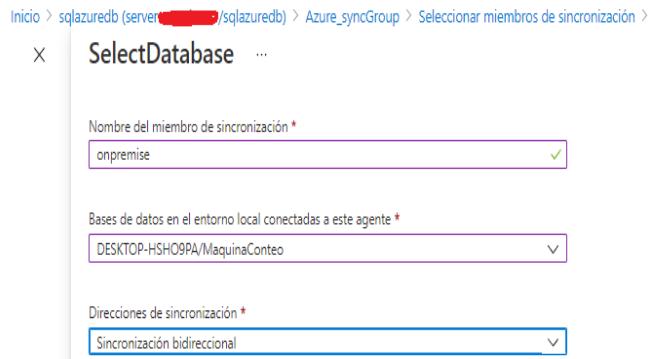
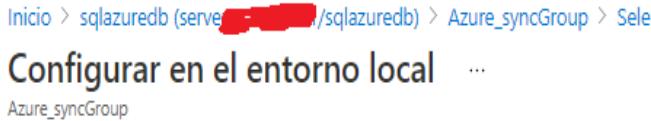


Figura 135. Seleccionar Base de datos On-premise. [Autores]
Si todo va bien, todo aparecerá como en la Figura 136.

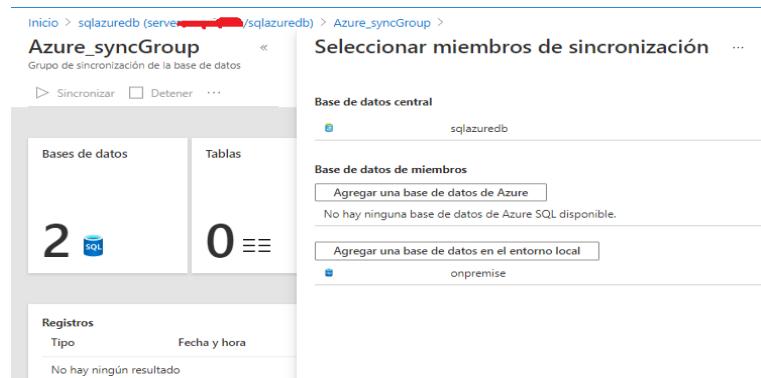


Figura 136. Comprobar miembros de sincronización (On-premise y Base de datos Azure).
[Autores]

De click en el apartado tablas, para así asignar la tabla que queremos seleccionar de nuestra base de datos On-Premise llamada (MaquinaConteo), tenga en cuenta que yo tengo una base de datos en SSMS pero puedo tener múltiples tablas por eso es necesario escoger cual de todas sincronizare... Data Sync permite conectar múltiples tablas, bastantes columnas; vea las limitaciones en la página oficial de Microsoft Azure.

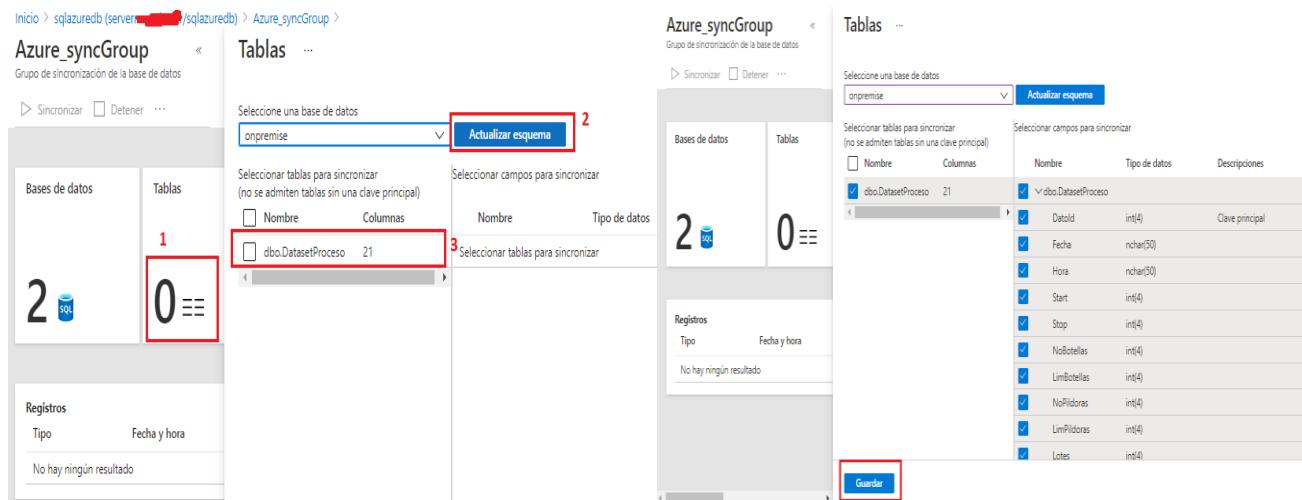


Figura 137. Seleccionar Base de datos On-premise. [Autores]

Ya para culminar el proceso presione el botón sincronizar y luego actualice los registros, nótese que hay errores en la conexión con la base de datos de azure (sqlazuredb) pero la conexión es exitosa entre el agente y la base de datos On-Premise, ver Figura 138. En nuestro caso nos dirigimos a la Figura 134 y se observa que sqlazuredb no esta listo por lo cual se vuelve a configurar, se revisa de nuevo el Agente y se vuelve a oprimir el Botón sincronizar (recuerde que Azure sincroniza los datos automáticamente cada 5 minutos o depende su tiempo, sin necesidad de presionar el botón) y luego actualiza los registros...observe la Figura 139 donde las conexiones son exitosas.

Tipo	Fecha y hora	Base de datos de mi...	Detalles
Error	21/09/21 15:51:41	sqlazuredb/serverma...	Database provisioning failed with the exception "Sql...
Error	21/09/21 15:46:35	sqlazuredb/serverma...	Database provisioning failed with the exception "Sql...
Success	21/09/21 15:46:31	MaquinaConteo/DES...	Database provisioned successfully in 1.54 seconds.

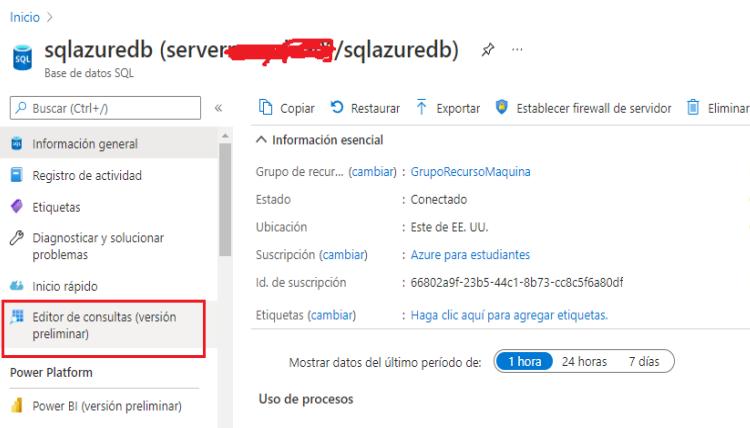
Figura 138. Sincronizar Grupo. [Autores]

Tipo	Fecha y hora	Base de datos de mi...	Detalles
Success	21/09/21 16:39:44	sqlazuredb/serverma...	Database provisioned successfully in 0.77 seconds.
Success	21/09/21 16:39:44	MaquinaConteo/DES...	Database provisioned successfully in 0.23 seconds.
Error	21/09/21 16:36:28	sqlazuredb/serverma...	Deprovisioning failed with the exception "Invalid job ...
Success	21/09/21 16:34:40	Deleted	Sync completed successfully in 6.14 seconds.
Success	21/09/21 16:34:31	sqlazuredb/serverma...	Database provisioned successfully in 3.86 seconds.
Error	21/09/21 16:23:02	sqlazuredb/serverma...	Database provisioning failed with the exception "Sql...

Figura 139. Sincronización exitosa. [Autores]

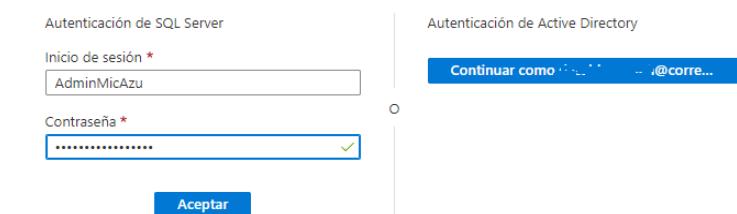
6. Probar envío de datos desde base On-Premise a Azure

Luego de estos mensajes exitosos, comprobaremos que los datos se sincronizan:



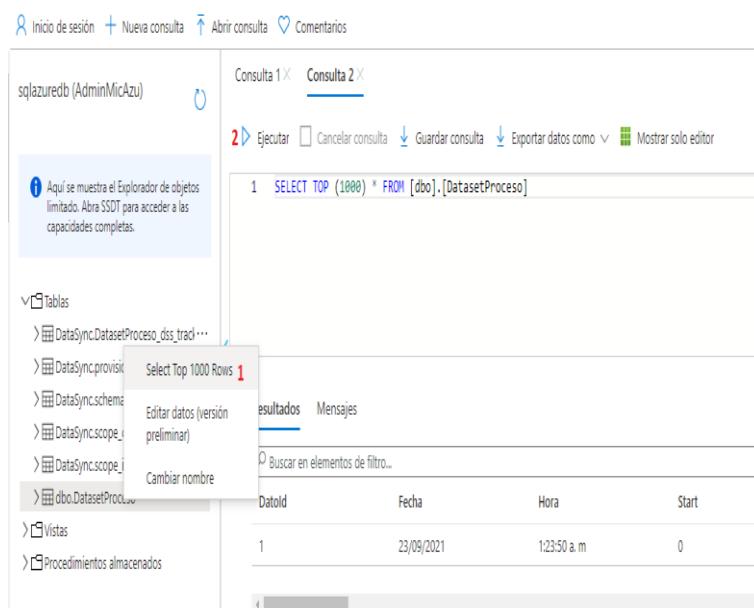
The screenshot shows the Azure portal interface for a SQL database named 'sqlazuredb'. The left sidebar has a red box around the 'Editor de consultas (versión preliminar)' link under the 'Power Platform' section. The main pane displays basic information about the database, including its connection status, location, and subscription details.

Figura 140. Editor de consultas en Azure. [Autores]



The login screen for the Azure SQL Database query editor. It offers two authentication methods: 'Autenticación de SQL Server' (using 'Inicio de sesión *' and 'Contraseña *') and 'Autenticación de Active Directory' (using 'Continuar como'). A blue 'Aceptar' button is at the bottom.

Figura 141. Ingrese con sus credenciales. [Autores]



The screenshot shows the Azure SQL Database query editor interface. The left sidebar lists database objects like tables and stored procedures. The main area contains a query window with the following content:

```
1 SELECT TOP (1000) * FROM [dbo].[DatasetProceso]
```

The results pane shows a table with three columns: 'Datold', 'Fecha', and 'Hora'. There is one row of data: 1, 23/09/2021, and 12:35 a.m.

Figura 142. Haga una consulta básica para ver los registros. [Autores]

7. Posibles Fallos

- Verifique la conexión online del Data Sync Agent (Figura 134)
- Suficiente almacenamiento para instalar programas
- Revise si el servicio de sincronización de datos esta pausado o eliminado, si esta eliminado cree un nuevo agente
- Si no conecta, trate de cambiar la clave del agente (Figura 131)
- Si no funciona lo anterior debe limpiar el Data Sync.

NOTA: Si ya no usara la sincronización, elimine el agente en Azure y en el programa Data Sync Agent para evitar consumos innecesarios.

8. Referencia

<https://www.youtube.com/watch?v=tVCOhm3aXVQ>

	<p>UNIDAD TEMÁTICA: Prácticas de laboratorio para la máquina de conteo</p> <p>ACTIVIDAD: Identificación y diseño del controlador discreto en Matlab</p> 	
Código: PC006 <input checked="" type="checkbox"/> ONLINE <input type="checkbox"/> OFFLINE		Duración: ---

PC006. Identificación y diseño del controlador discreto en Matlab

Objetivo de la práctica:

Usando el software MATLAB se desea encontrar el modelo matemático de la planta en lazo abierto mediante la identificación de sistemas, aplicando conceptos básicos y fundamentales en la ingeniería de control. El desarrollo de un control PID en un motor DC es esencial en el aprendizaje y su correcto diseño ayudara a afianzar las competencias del estudiante de ingeniería

Material Necesario y requisitos para el desarrollo:

- Codesys V3.5 SP16 o superior
- MATLAB R2019a

Esquema Grafico de la Actividad:



Figura 143. Esquema básico de la actividad a realizar. [Autores]

Requisitos previos:

- Reconocimiento Maquina de Conteo y SoftPLC

Resumen:

Para la regulación y la optimización del sistema se implementó un control PID para manejar el comportamiento del Motor DC de la banda transportadora, el objetivo de aplicar este controlador es poder regular la velocidad a la cual se desea realizar la producción, con base en esto, a la máquina se le designaron dos tipos de regulaciones una por margen de demanda y otro por distancia.

En el primer modo la regulación es por demanda y se tiene en cuenta los niveles de producción que determine el usuario dentro del MES, es decir que dependiendo del número de lotes y la cantidad de botellas que deban dispensar y salir de la producción, el sistema se ajustara de forma automática, entre más demanda se estipule para el proceso más velocidad tendrá la banda. Al no tener un modelo matemático de la planta el cual permita conocer las características dinámicas de la misma, se realizó el diseño de un controlador en un sistema de caja negra, los pasos a seguir se observan en la Figura 144.

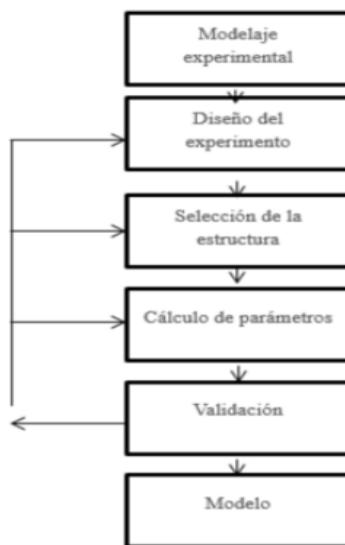


Figura 144. Metodología a seguir.

En el bloque de **Diseño del experimento** se usa una señal paso, generada en Codesys a partir de un pseudocódigo en estructurado, los datos almacenarán en un csv (archivo separado por comas) por medio de la librería CSVUtility del software mencionado anteriormente.

En el caso de **Selección de estructura** se deben ingresar al toolbox de identificación en MATLAB los datos obtenidos en el bloque anterior. En esta herramienta se generarán las distintas aproximaciones a arquitecturas en tiempo continuo(primer orden o segundo orden), también se puede crear modelos discretos lineales como el OE (Output error), ARX (Auto-regresivo con variable exógena, entre otros).

Para el **Cálculo de los parámetros** se generan los resultados de los parámetros por medio de Matlab ya que el software usar una aproximación por medio de mínimos cuadrados y regresión lineal. En el caso de un sistema de primer orden se obtendrá el K, TPI, y el porcentaje de ajuste.

Se **Valida** la respuesta del modelo con el toolbox de identificación en Matlab, para esto se debe graficar la curva de reacción y observaremos visualmente las señales del experimento y la señal del sistema que diseñemos, la estructura a usar es la que mejor aproximación tenga.

Por último, luego de los pasos anteriores se obtiene el **Modelo** que representara el sistema de velocidad de la máquina por medio de una función de transferencia.

Desarrollo de la Actividad:

1. Creación señales de identificación y almacenamiento de Datos

El experimento se realizó en Codesys las señales de identificación como la señal paso, señal paso modificada, la señal paso cuadrada, y la señal pseudoaleatoria, donde es posible modificar la composición de la señal, estimular el sistema con la señal respectiva y almacenar la información que se obtiene del sistema al reaccionar a la señal implementada. A continuación, se observa y la interface desarrollada para la manipulación de las señales y la programación para generar los estímulos respectivos del sistema:

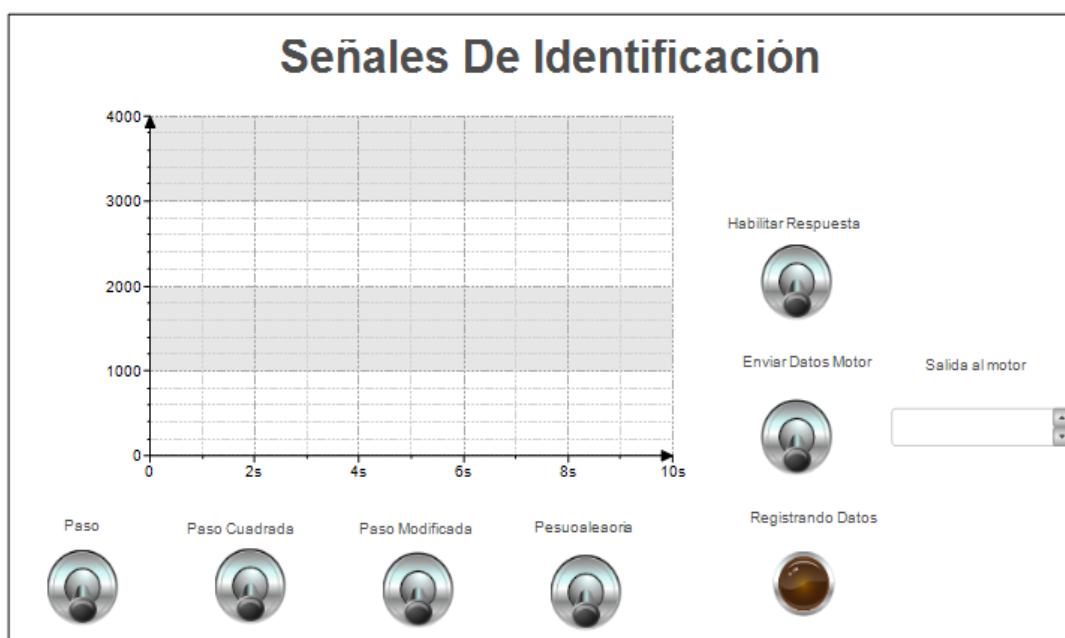


Figura 145. Interfaz Señales de identificación en Codesys.[Autores]

En el programa Codesys se estructuro una interface de usuario, ver Figura 145 en la cual se aplican las señales de identificaron de forma más sencilla al sistema, esta HMI se compone de botones para seleccionar el tipo de señal, un habilitador para generar la respuesta en el sistema con la señal elegida, un habilitador para almacenar los datos obtenidos en el experimento, hay un indicador para comprobar que los datos se están guardando correctamente y por ultimo hay una caja de selección donde se puede modificar la amplitud a la cual se va a estimular el sistema (punto de operación), cabe aclarar que para este experimento se están trabajo de 0 a 4000 bits ya que se están empleado las salidas análogas del softPLC.

1.1. Señal paso

Esta señal se compone de un estado que varía en un tiempo determinado y cambia de una amplitud cero a una amplitud máxima, manteniendo dicho valor durante el tiempo que dure energizado ver Figura 148 .El código para la construcción de la señal se realizó en lenguaje estructurado y se compone de dos condiciones: La primera se utiliza para accionar la señal que a su vez activa un temporizador que determina el tiempo de espera para que la señal cambie de estado aumentando la amplitud y produciendo así un pasó.

```
1  IF GVL.B_Paso AND Salir = FALSE THEN
2      Type_Signal:=1;
3      GVL.Guardar:=TRUE;
4      //Forma de la señal
5      IF Finalizado = TRUE THEN
6          Espera:=0;
7          GVL.Signal_P:=1;
8
9      ELSE
10         Espera:=1;
11     END_IF
12     //Número de datos para muestrear
13     IF GVL.Salida_CSV THEN
14         Numero_Datos_Aux:=Numero_Datos_Aux+1;
15     ELSE
16         Numero_Datos_Aux:=Numero_Datos_Aux;
17     END_IF
18     IF Numero_Datos_Aux = 1000 THEN
19         Salir:=TRUE;
20         GVL.B_Paso:=FALSE;
21     END_IF
22 END_IF
```

Figura 146. Código en Lenguaje estructurado señal paso. [Autores]

La segunda parte se emplea para realizar el proceso de adquisición de datos y guardar la información de la respuesta del sistema como un archivo CSV (este proceso se describirá con más detalle en el apartado de adquisición de datos). En la siguiente imagen se observa la secuencia en ladder que se encarga de ajustar los tiempos que dura el cambio de estado de la señal y en la Figura 146 el código realizado para la señal paso en estructurado



Figura 147. Código en Ladder para señal paso. [Autores]

Finalmente, en la Figura 148 se puede observar la respuesta de la señal paso en la interface de usuario, con sus respectivos habilitadores y su amplitud determinada.

Señales De Identificación

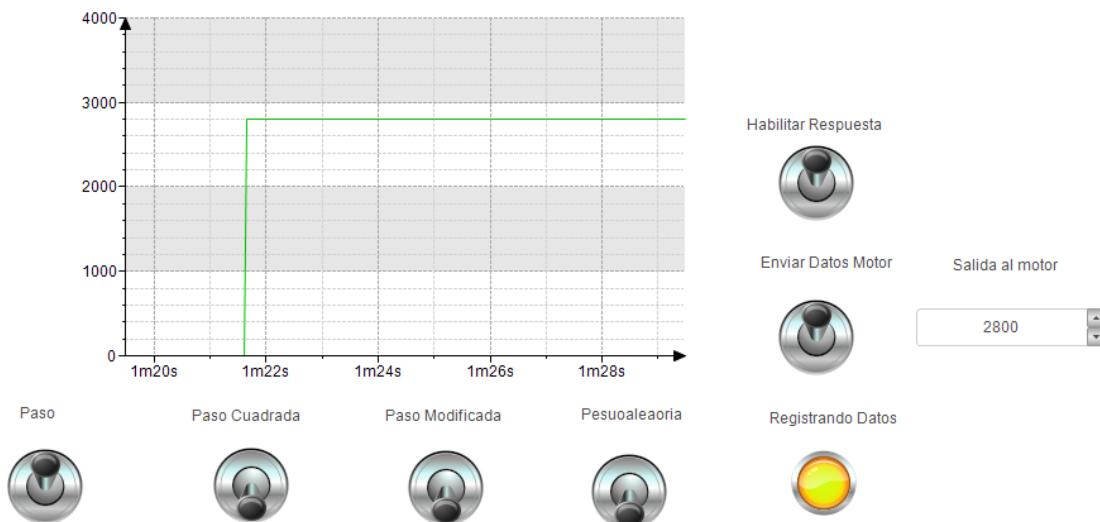


Figura 148. Visualización señal paso en Codesys. [Autores]

1.2. Señal paso cuadrada

Esta señal es una combinación de una señal paso y una señal cuadrada, esta realiza un cambio luego de un tiempo determinado hasta un valor máximo, cuando llega a este punto comienza con un proceso de oscilación entre un valor mínimo y un valor máximo, esta señal se comporta de forma simétrica respecto a su eje y posee una frecuencia de oscilación constante ver la Figura 152. El código de esta señal se realizó en lenguaje estructurado y posee las mismas condiciones que la señal paso explicada anteriormente, una para la creación de la señal y otro para el almacenamiento de información. El código combina el tiempo de espera para una señal paso común y luego responde al activación y desactivación de un generador que se encarga de producir la señal cuadrada luego de un tiempo determinando, este proceso se puede ver en la siguiente figura.

```

24 IF GVL.B_Paso_Cuadrada AND Salir = FALSE THEN
25     Type_Signal:=2;
26     GVL.Guardar:=TRUE;
27     //Forma de la señal
28     IF Finalizado = TRUE THEN
29         Espera:=0;
30         GVL.Signal_PC:=1;
31         IF Finalizado_2 THEN
32             Espera_2:=0;
33             GVL.Signal_PC:=1.2;
34             Pulsos_Cuadrado:=TRUE;
35             IF Tiempo_Cuadrada = TRUE THEN
36                 GVL.Signal_PC:=0.8;
37             ELSIF Tiempo_Cuadrada = FALSE THEN
38                 GVL.Signal_PC:=1.2;
39             END_IF
40         ELSE
41             Espera_2:=1;
42         END_IF
43     ELSE
44         Espera:=1;
45     END_IF
46
47 //Número de datos para muestrear
48 IF GVL.Salida_CSV THEN
49     Numero_Datos_Aux:=Numero_Datos_Aux+1;
50 ELSE
51     Numero_Datos_Aux:=Numero_Datos_Aux;
52 END_IF
53 IF Numero_Datos_Aux = 1000 THEN
54     Salir:=TRUE;
55     GVL.B_Paso_Cuadrada:=FALSE;
56 END_IF
57 END_IF

```

Figura 149. Código en Lenguaje estructurado señal paso cuadrada. [Autores]

Como se observa en la Figura 151 la aplicación del temporizador se utiliza para llevar la señal hasta un punto específico después de un tiempo determinando y luego se emplea un generador de pulsos que se activa y desactiva en un tiempo escogido.



Figura 150. Código en Ladder para señal paso cuadrada. [Autores]



Figura 151. Temporizador en Ladder. [Autores]

En la siguiente imagen se puede observar la respuesta que proporciona la señal paso cuadrada al ser seleccionada en el HMI.

Señales De Identificación

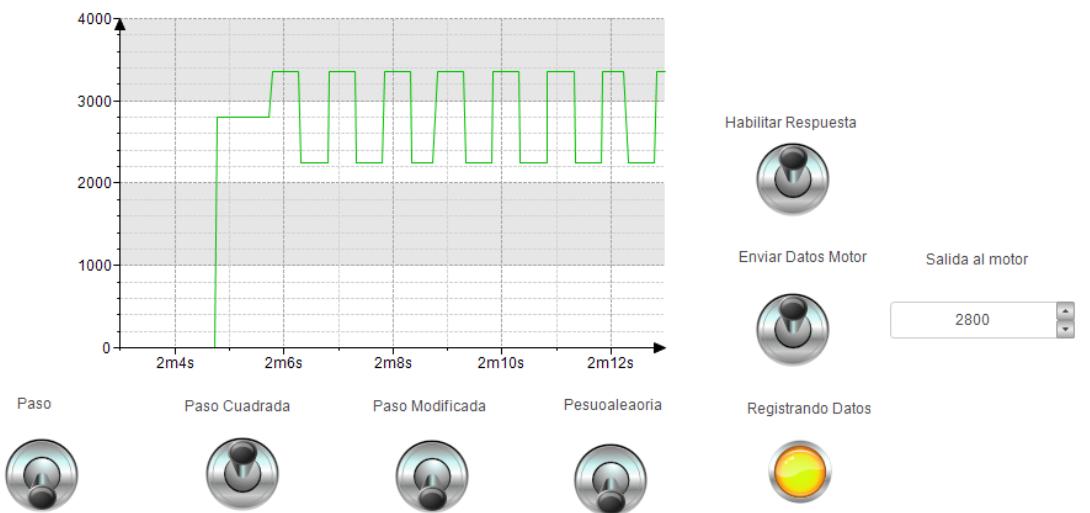


Figura 152. Visualización señal paso cuadrada en Codesys. [Autores]

1.3. Señal paso modificada

Esta señal es la misma señal paso convencional, pero tiene una pequeña perturbación luego de un tiempo determinando, esto quiere decir que la señal se acciona luego de un tiempo hasta un valor específico, espera otro tiempo y vuelve a realizar un cambio hasta otro valor. La programación de esta señal contiene las mismas dos secciones que se han descrito en las señales anteriores, condición para programación y para adquisición de datos. En la siguiente imagen se observa que el código es similar al de la señal paso pero se agrega otro variable de espera.

```

59      IF GVL.B_Paso_Modificada AND Salir = FALSE THEN
60          Type_Signal:=3;
61          GVL.Guardar:=TRUE;
62          IF Finalizado = TRUE THEN
63              Espera:=0;
64              GVL.Signal_CM:=1;
65              IF Finalizado_2 THEN
66                  Espera_2:=0;
67                  GVL.Signal_CM:=1.2;
68              ELSE
69                  Espera_2:=1;
70              END_IF
71
72          ELSE
73              Espera:=1;
74          END_IF
75          //Número de datos para muestrear
76          IF GVL.Salida_CSV THEN
77              Numero_Datos_Aux:=Numero_Datos_Aux+1;
78          ELSE
79              Numero_Datos_Aux:=Numero_Datos_Aux;
80          END_IF
81          IF Numero_Datos_Aux = 1000 THEN
82              Salir:=TRUE;
83              GVL.B_Paso_Modificada:=FALSE;
84          END_IF
85      END_IF

```

Figura 153. Código en Lenguaje estructurado señal paso modificada. [Autores]
En el siguiente código en ladder se observa de forma más clara los dos temporizadores que se emplean para la construcción de esta señal



Figura 154. Código en Ladder para señal paso modificada. [Autores]
Y por último vemos la representación de la señal paso modificada en la interface de usuario

Señales De Identificación

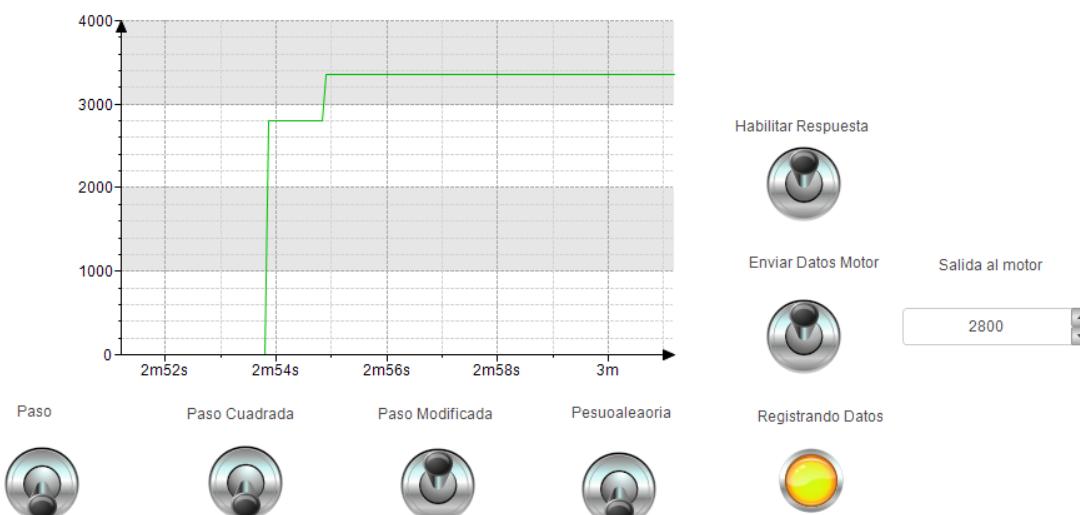


Figura 155. Visualización señal paso modificada en Codesys. [Autores]
1.4. Señal pseudoaleatoria

La última señal es un poco más compleja que las anteriores, pero su principio es el mismo, para esta señal tanto la amplitud como la frecuencia varían dentro de un rango máximo y mínimo determinado ver Figura 158. El código realizado tiene la estructura de las señales anteriores y combina los recursos ya mencionados en su programación para la construcción de la señal, este código emplea un acumulador que modifica el periodo de la señal, se trabajan tiempos de espera que van modificándose y se producen pulsos de forma aleatoria, tal como se muestra en la siguiente imagen

```

87 IF GVL.B_Pseudealeatoria AND Salir = FALSE THEN
88   Type_Signal:=4;
89   GVL.Guardar:=TRUE;
90   Periodo_PA:=REAL_TO_TIME(Periodo_Aleatorio+Periodo_Aleatorio_Entero);
91   IF Finalizado = TRUE THEN
92     Espera:=0;
93     GVL.Signal_PA:=1;
94     IF Finalizado_2 THEN
95       Espera_2:=0;
96       Pulso_Aleatorios:=TRUE;
97       GVL.Signal_PA:=1+Salida_Random;
98     ELSE
99       Espera_2:=1;
100    END_IF
101
102  ELSE
103    Espera:=1;
104  END_IF
105 //Número de datos para muestrear
106 IF GVL.Salida_CSV THEN
107   Numero_Datos_Aux:=Numero_Datos_Aux+1;
108 ELSE
109   Numero_Datos_Aux:=Numero_Datos_Aux;
110 END_IF
111 IF Numero_Datos_Aux = 1000 THEN
112   Salir:=TRUE;
113   GVL.B_Pseudealeatoria:=FALSE;
114 END_IF
115 END_IF

```

Figura 156. Código en Lenguaje estructurado señal pseudoaleatoria. [Autores]

Los tiempos de esta señal van cambiando en todo momento por esta razón se trabajó en conjunto con un generador de numero aleatorios, este me permite almacenar un número diferente en cada momento para que se cambie el periodo de la señal, la amplitud y los tiempos de espera, de esta forma se obtienen un señal que no es fija y que cada vez que obtiene un valor de estabilización este se actualiza. En la siguiente imagen se observan los módulos que se emplearon en ladder para determinar los tiempos del proceso descrito.

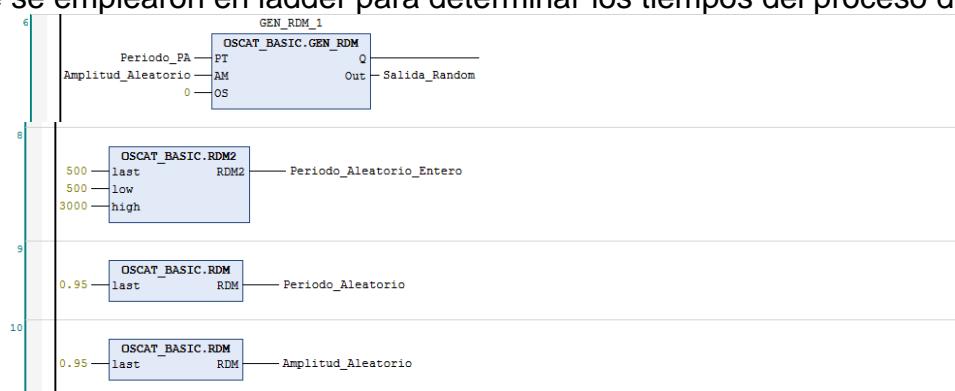


Figura 157. Código en Ladder para señal pseudoaleatoria. [Autores]

En la siguiente imagen se observa la representación de la señal pseudoaleatoria en el HMI, además en los anexos puede encontrar el código de las señales en estructurado.

Señales De Identificación

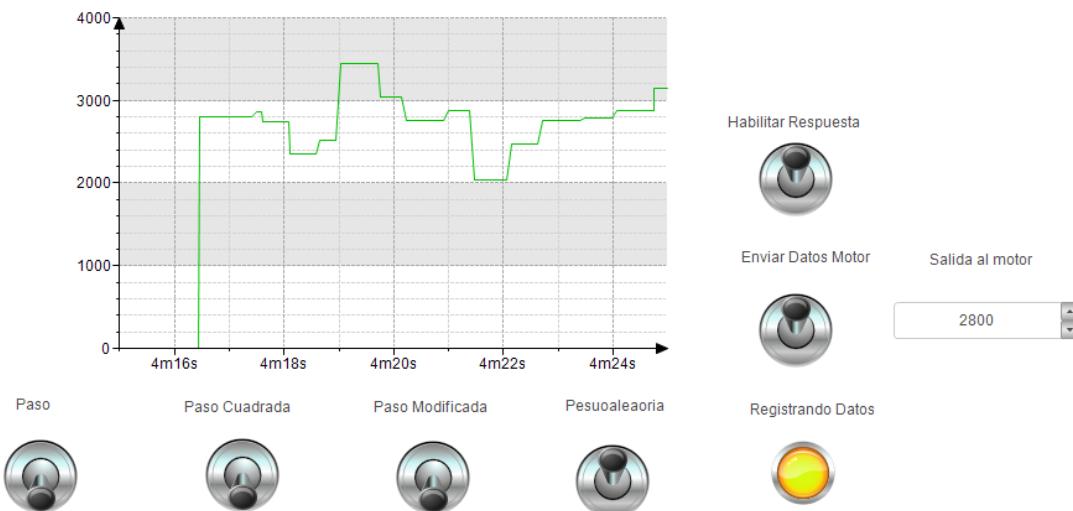


Figura 158. Visualización señal pseudoaleatoria en Codesys. [Autores]

1.5. Adquisición de datos

Para observar con detenimiento las señales de identificación y la respuesta que tiene el sistema al ser estimulado con dichas señales es necesario almacenar la lectura de los valores que se presentan en la salida y en la entrada del sistema, es decir la salida es la señal que se genera para estimular el sistema y la entrada es el valor que se obtiene al realizar un proceso de realimentación donde se genera una representación de la respuesta con la que se estimula el sistema. Como el proceso de adquisición de datos se realiza en el mismo software de Codesys, se debe recopilar la información con la que se produce dicha señal, como el sistema que se trabaja es un softPLC solo se debe leer la salida analógica que estimula el sistema y la entrada analógica que representa la respuesta del sistema. Teniendo esto en cuenta se empleó la librería CSVutility de codesys, la cual permite generar archivos csv y almacenarlos en una dirección específica.

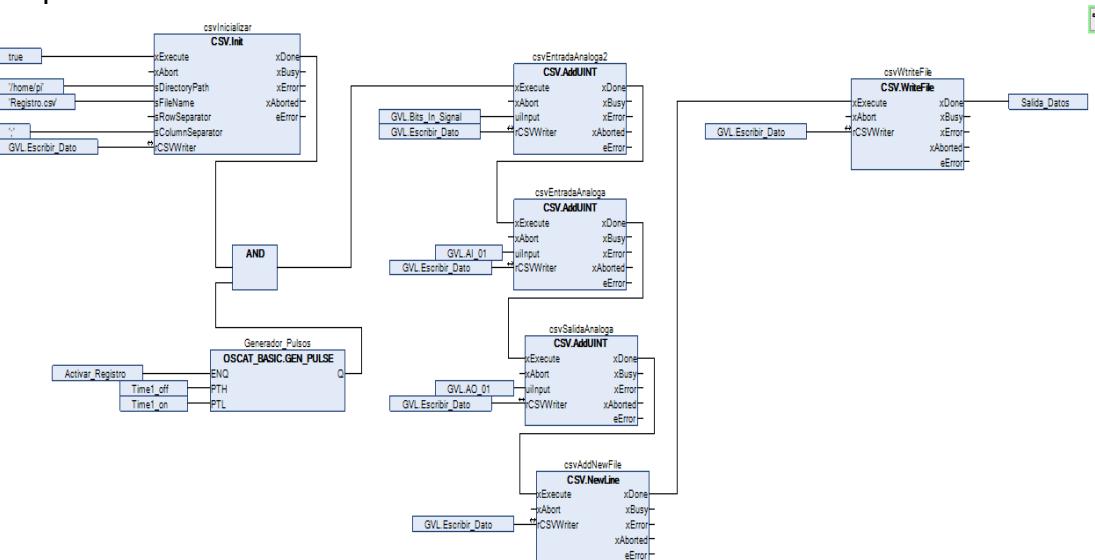


Figura 159. Bloques CFC para almacenamiento datos formato csv. [Autores]

En la anterior imagen se observa el código en CFC donde se emplea un generar de pulsos para almacenar datos cada cierto tiempo, el resto de bloques corresponde a los datos que se guardan. Un aspecto importante es que, el tiempo con el cual se van a guardar los datos, corresponde al tiempo de muestreo de la señal, por lo tanto, es un dato necesario para el proceso de identificación y este se debe indicar en el programa al inicializar las variables Time1_off y Time1_on Figura 160.

```

1  FUNCTION_BLOCK Datos_CSV
2  VAR_INPUT
3      Activar_Registro:BOOL;
4  END_VAR
5  VAR_OUTPUT
6      Salida_Datos:BOOL;
7  END_VAR
8  VAR
9      csvInicializar: CSV.Init;
10     csvEntradaAnaloga: CSV.AddUINT;
11     csvSalidaAnaloga: CSV.AddUINT;
12     csvAddNewFile: CSV.NewLine;
13     csvWriteFile: CSV.WriteLine;
14     csvWriteFile: CSV.WriteLine;
15     Generador_Pulsos: OSCAT_BASIC.GEN_PULSE;
16     //Tiempo de muestreo
17     Timel_off: TIME:=T#0.01S;
18     Timel_on: TIME:=T#0.01S;
19     csvEntradaAnaloga2: CSV.AddUINT;
20 END_VAR

```

Figura 160. Definir los tiempos para recolectar datos. [Autores]

Cabe aclarar que como el softPLC trabaja sobre un entorno embebido la información va quedar almacenada en el sistema operativo de la Raspberry Pi tal como se muestra en la siguiente figura.

La información que se almaceno hace referencia a un motor DC por lo cual los datos que se tomaron están en bits, pero se realiza una serie de conversiones para poder trabajar con diferentes unidades de medida, en la siguiente tabla se puede observar el modo con el que se organizó la información.

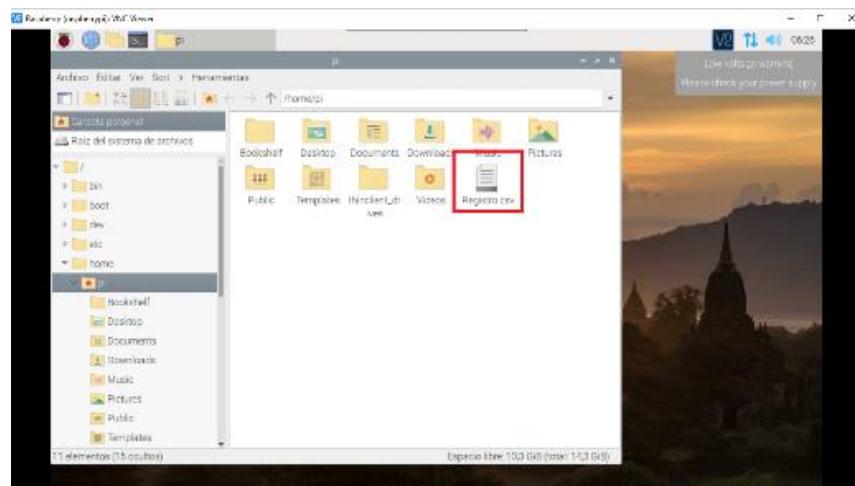


Figura 161. Ruta del csv en la Raspberry. [Autores]

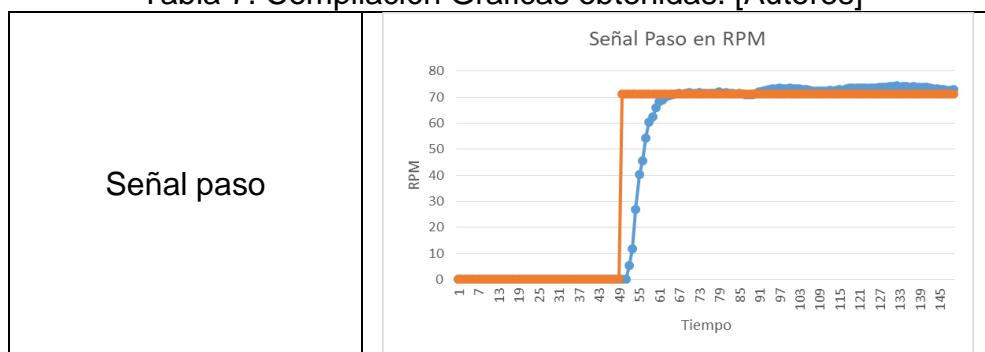
Tabla 6. Datos tomados para una de las señales. [Autores]

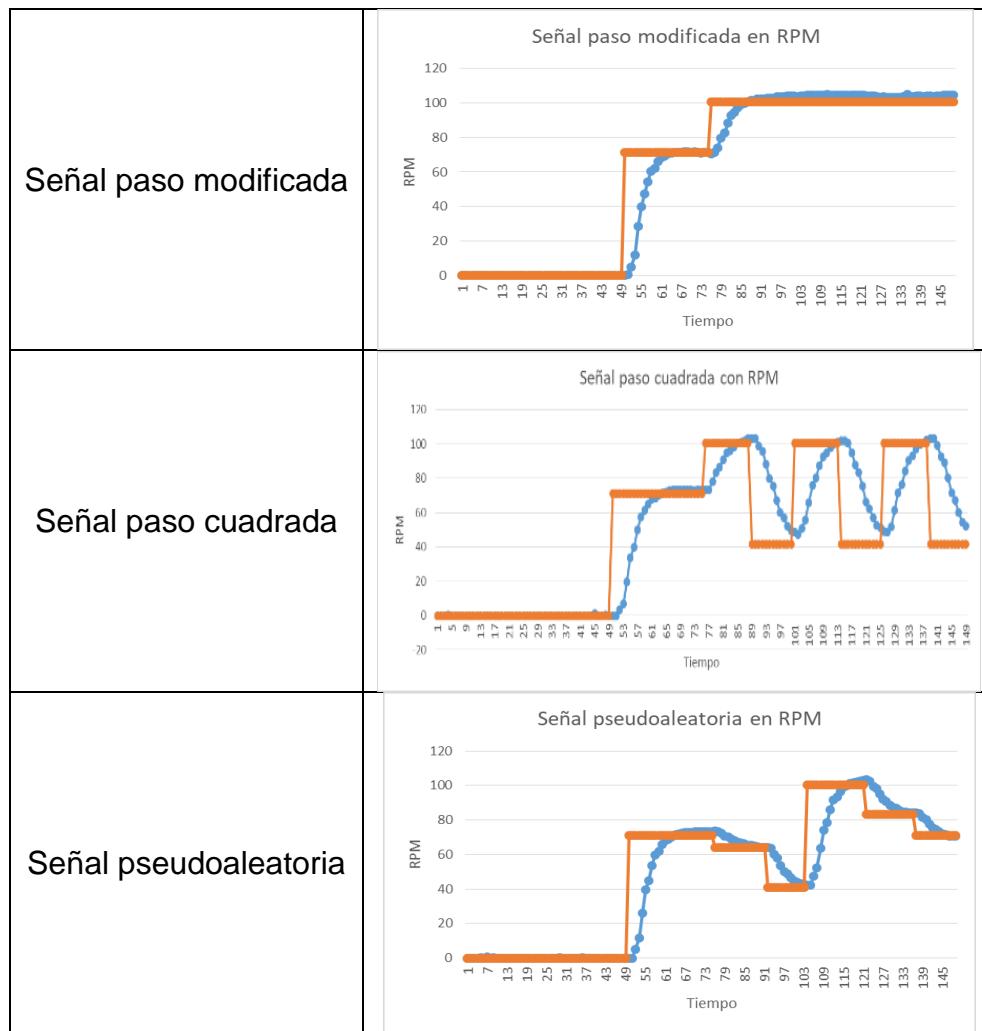
y(b) offset	y(b)	u(b)	y(v) offset	y(v)	u(v)	RPM	SP
1546	0	0	3,05131579	0	0	-0,045	0
1546	2	0	3,05131579	0,00394737	0	-0,045	0

1567	27	0	3,09276316	0,05328947		0	0,984	0
1546	0	0	3,05131579	0		0	-0,045	0
1546	0	0	3,05131579	0		0	-0,045	0
1552	7	0	3,06315789	0,01381579		0	0,249	0
1546	0	0	3,05131579	0		0	-0,045	0
1546	0	3000	3,05131579	0	5,92105263	-0,045	71,201	
1546	0	3000	3,05131579	0	5,92105263	-0,045	71,201	
1610	82	3000	3,17763158	0,16184211	5,92105263	3,091	71,201	
1682	331	3000	3,31973684	0,65328947	5,92105263	6,619	71,201	
1951	520	3000	3,85065789	1,02631579	5,92105263	19,8	71,201	
2233	883	3000	4,40723684	1,74276316	5,92105263	33,618	71,201	
2363	1194	3000	4,66381579	2,35657895	5,92105263	39,988	71,201	
2566	1311	3000	5,06447368	2,5875	5,92105263	49,935	71,201	
2720	1509	3000	5,36842105	2,97828947	5,92105263	57,481	71,201	
2803	1654	3000	5,53223684	3,26447368	5,92105263	61,548	71,201	
2870	1701	3000	5,66447368	3,35723684	5,92105263	64,831	71,201	
2933	1782	3000	5,78881579	3,51710526	5,92105263	67,918	71,201	
2947	1826	3000	5,81644737	3,60394737	5,92105263	68,604	71,201	
2984	1848	3000	5,88947368	3,64736842	5,92105263	70,417	71,201	
3006	1877	3000	5,93289474	3,70460526	5,92105263	71,495	71,201	
3016	1898	3000	5,95263158	3,74605263	5,92105263	71,985	71,201	
3029	1906	3000	5,97828947	3,76184211	5,92105263	72,622	71,201	
3040	1920	3000		6	5,92105263	73,161	71,201	
3040	1919	3000		6	5,92105263	73,161	71,201	
3042	1923	3000	6,00394737	3,79539474	5,92105263	73,259	71,201	

De acuerdo a la tabla anterior los tópicos amarillos corresponden a los datos en bits, los azules a valores de voltaje y los verdes a las RPM, este proceso se replicó para cada una de las señales de identificación y con base en esto se logró graficar las respuestas del sistema de acuerdo a los distintos estímulos. En la tabla siguiente se puede ver el setpoint (línea naranja) que hace alusión a la señal de identificación y la respuesta del sistema ante el estímulo (línea azul), todo en unidades de RPM, para cada experimento se tomaron 150 datos y el tiempo de muestreo es cada 10 ms.

Tabla 7. Compilación Graficas obtenidas. [Autores]





Otra forma de almacenar datos sin usar CSVUtility es por medio de la creación de una traza en Codesys la cual permite graficar más de una variable y permite exportar los datos con un par de clicks. La importancia de un archivo csv radica en que es muy usado para datasets y almacenar datos en el mundo del análisis de datos. Es importante irse acostumbrando a los formatos usados hoy en día.

2. Diseño de la estructura Continua

Luego de capturar los datos en un csv, se utiliza el IDENT de Matlab, en la siguiente figura se observa el comando para ejecutar el toolbox de Ident.



```

Command Window
New to MATLAB? See resources for Getting Started.

>> ident
Warning: The "ident" command is obsolete and may be removed in a future release of MATLAB. Use the "systemIdentification"
command instead.
> In ctrlMugUtils.warning (line 25)
    In ident (line 44)

f> >> |

```

Figura 162. Command Window de MATLAB. [Autores]

Como se observa en la Figura 163 el sistema de identificación de MATLAB está compuesto básicamente por dos secciones, en una se importan los datos y en la otra sección se observarán los modelos creados.

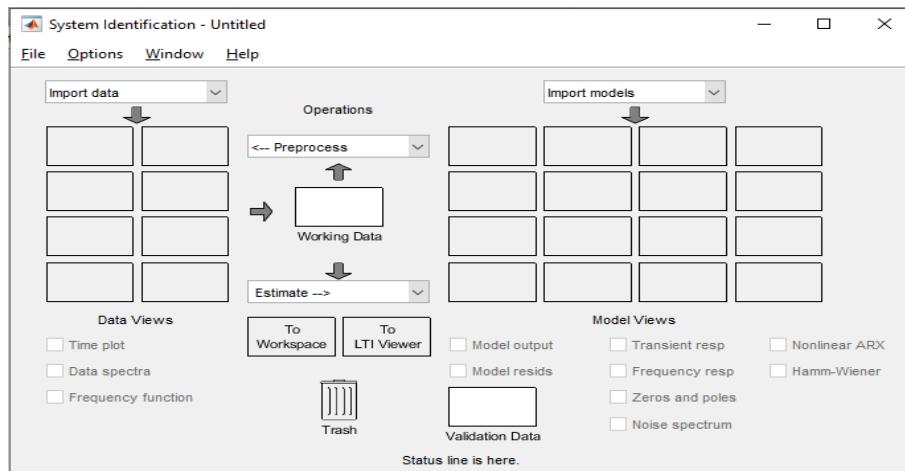


Figura 163. System Identification MATLAB.[Autores]

En primer lugar, ya sea en caso discreto o continuo, se deben exportar los datos tomados en el csv y almacenarlos en variables.

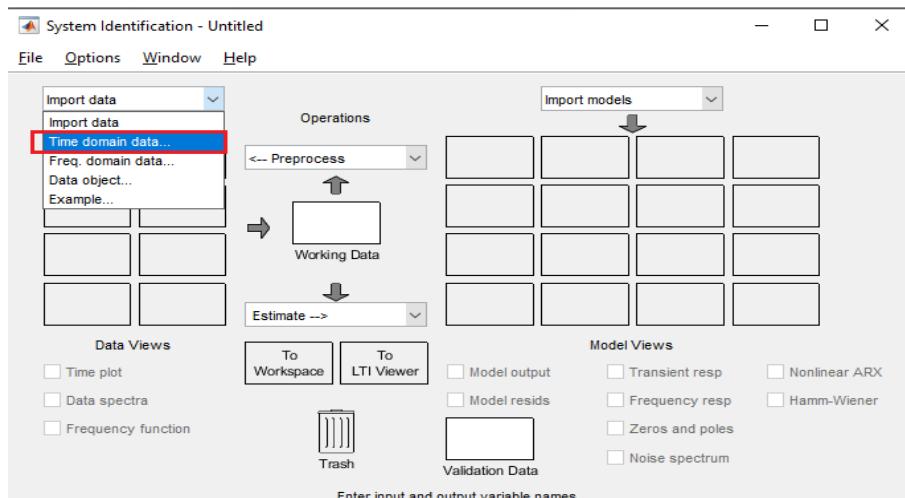


Figura 164. Datos en el dominio del tiempo en IDENT. [Autores]

Por tal motivo antes de seleccionar la opción de la Figura 164, proceda a cargar los datos en MATLAB, como se observa en la figura siguiente

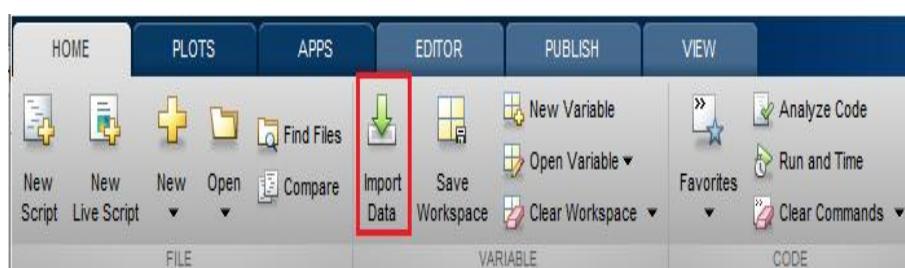


Figura 165. Importar Datos en Matlab. [Autores]

Para el diseño del controlador se usará en primer lugar la señal paso y los datos obtenidos en dicho experimento, como se puede observar en la siguiente figura hay un tratamiento previo

como por ejemplo filtrar el rango de muestras que deseamos, configurar los atributos a exportar como columnas de vectores de tipo double, por ultimo selección **Import Selection**.

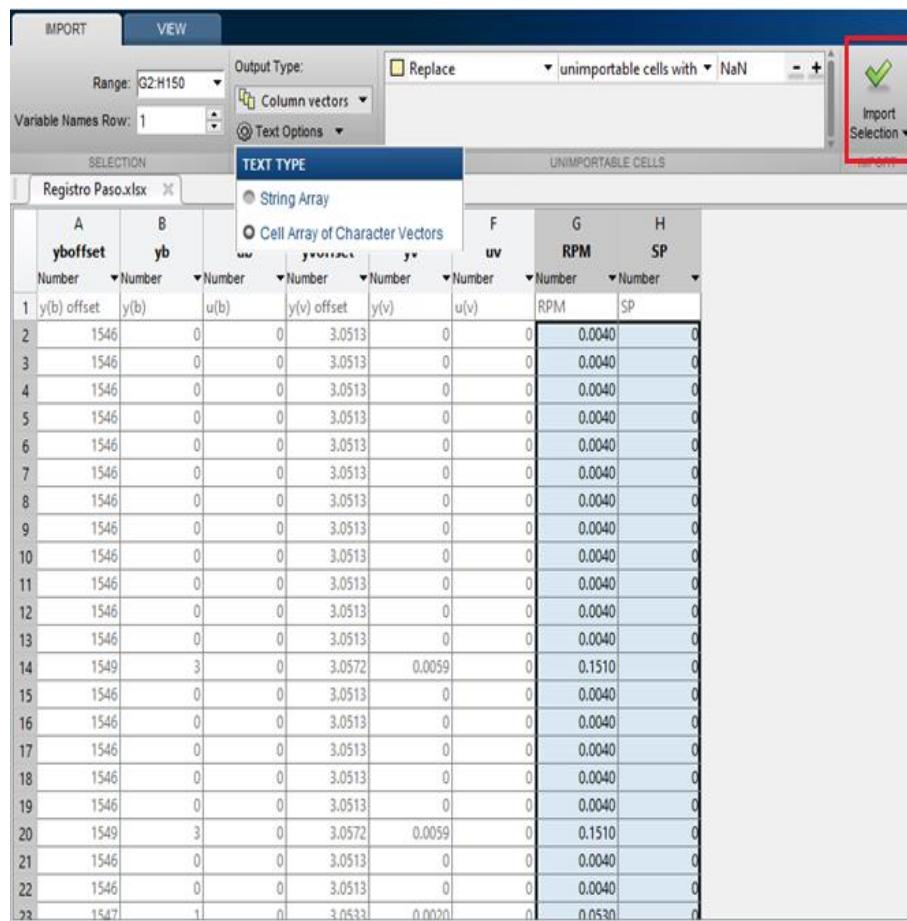


Figura 166. Filtrar Datos a usar. [Autores]

En la Figura 167 se observan los atributos exportados en el Workspace de MATLAB.

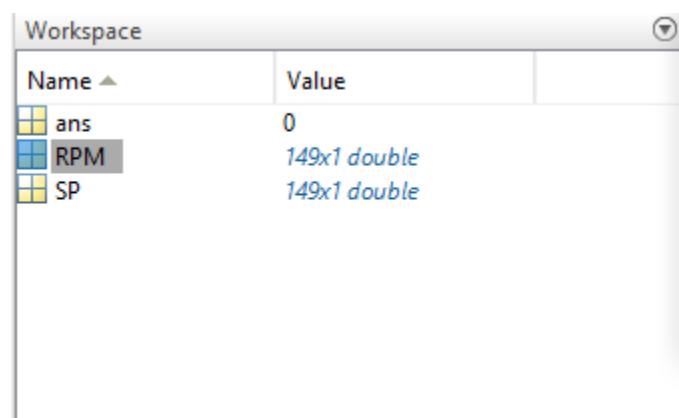


Figura 167. Atributos exportados al Workspace de MATLAB. [Autores]

Primero se configuran los parametros como los valores de entrada y salida del experimento. Observe el tiempo de muestro de 10ms (los datos se tomaron con esa frecuencia de muestreo), por ultimo oprima el boton **Import**.

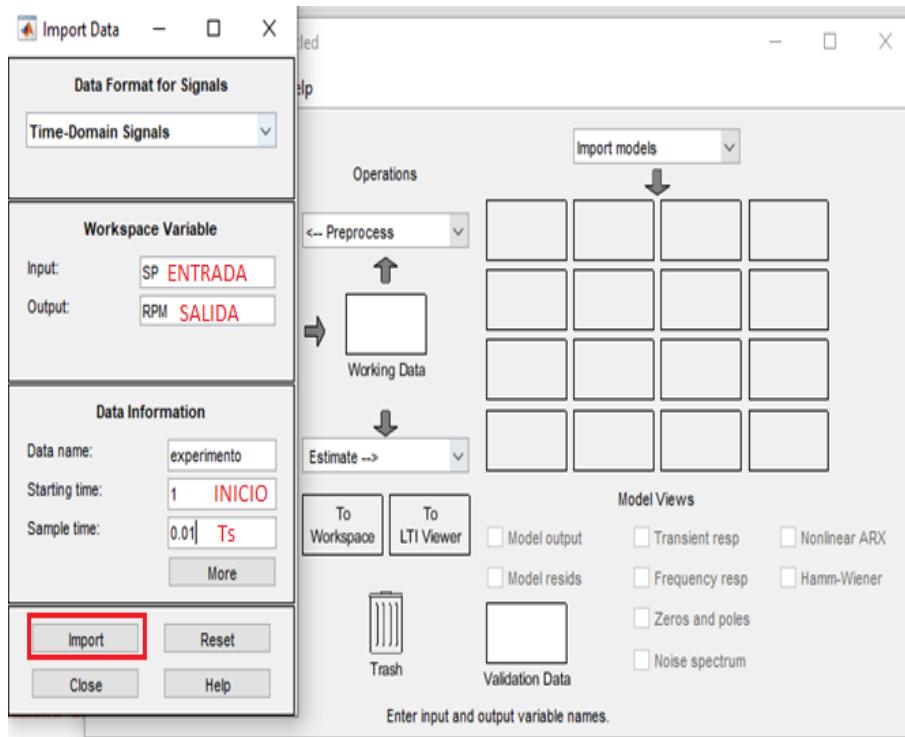


Figura 168. Configurar la señal en IDENT. [Autores]

Luego de importar los datos, seleccione **Time Plot** para observar la grafica que se genera automáticamente

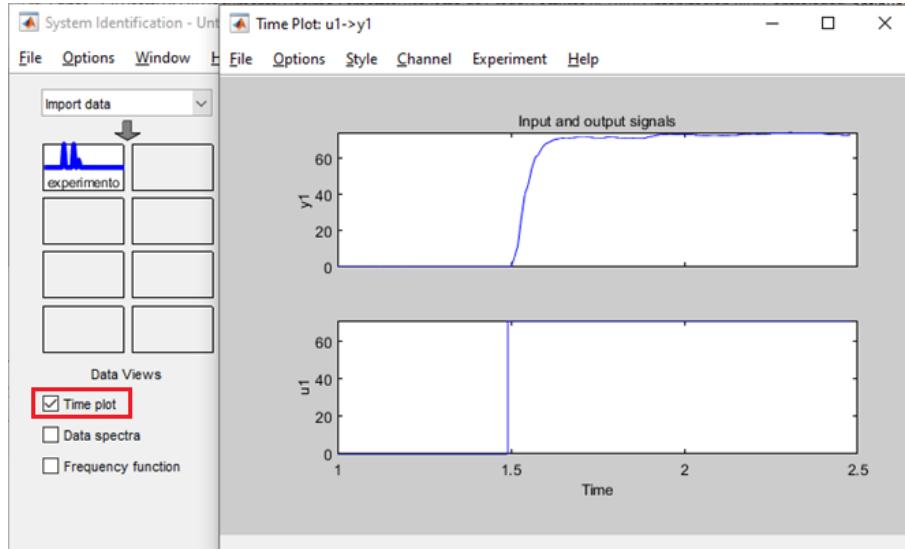


Figura 169. Observar la salida y entrada. [Autores]

Fue importante modificar el grafico para poder ver con más detalle el experimento ya que debe tener en cuenta que entre mejor estén los datos tomados, el controlador deberá ser mejor. Se observan que los datos tomados oscilan bastante y no se han podido mantener en el valor nominal de 71 rpm. Otra cosa que ha de poder observarse es el retraso que tiene el experimento, ya que la señal de entrada (u1) se inyecta antes del segundo 1.5 pero observe que la salida responde en el segundo 1.5 . Tenga en cuenta este **Delay**.

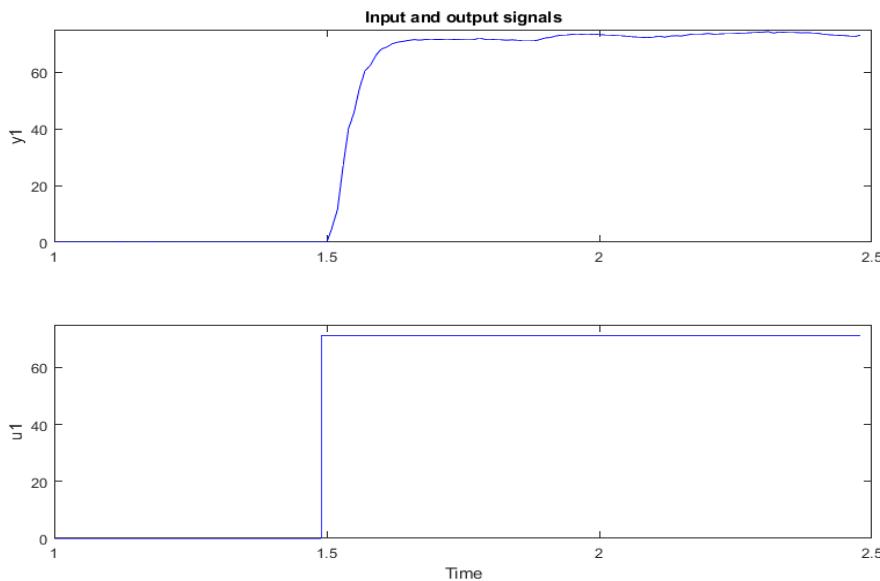


Figura 170. Visualización entrada/salida. [Autores]

Se debe realizar un filtrado previo al análisis tanto en tiempo continuo como discreto; seleccione en operaciones de Ident, la opción de seleccionar rangos para definir en que intervalo o muestras desea analizar el experimento.

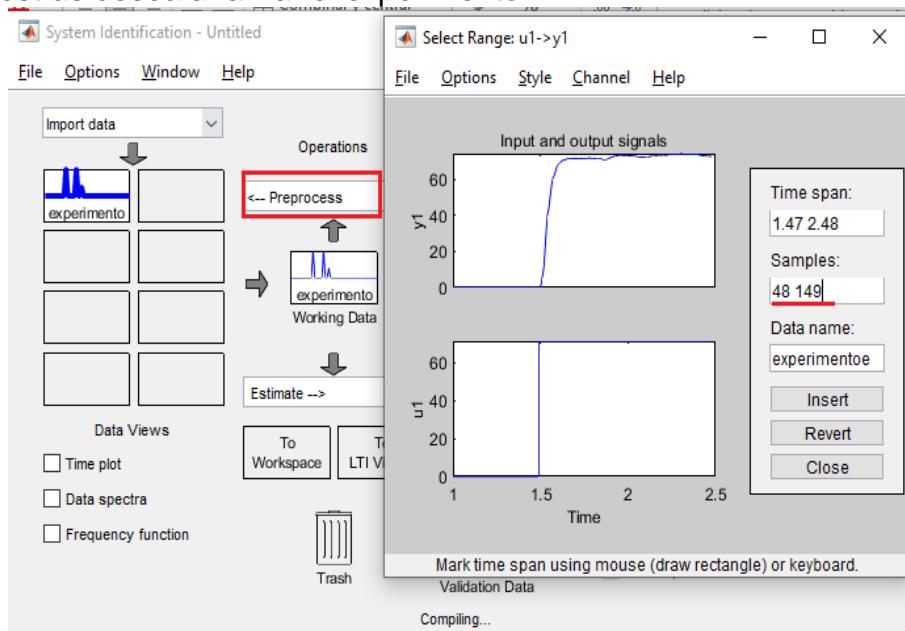


Figura 171. Seleccionar Rangos de trabajo. [Autores]

Al graficar la señal **experimentoe**, se observa con mejor detalle el delay de la planta. En la siguiente figura también podemos ver que IDENT grafica la señal que este seleccionada , en nuestro caso **experimentoe**.

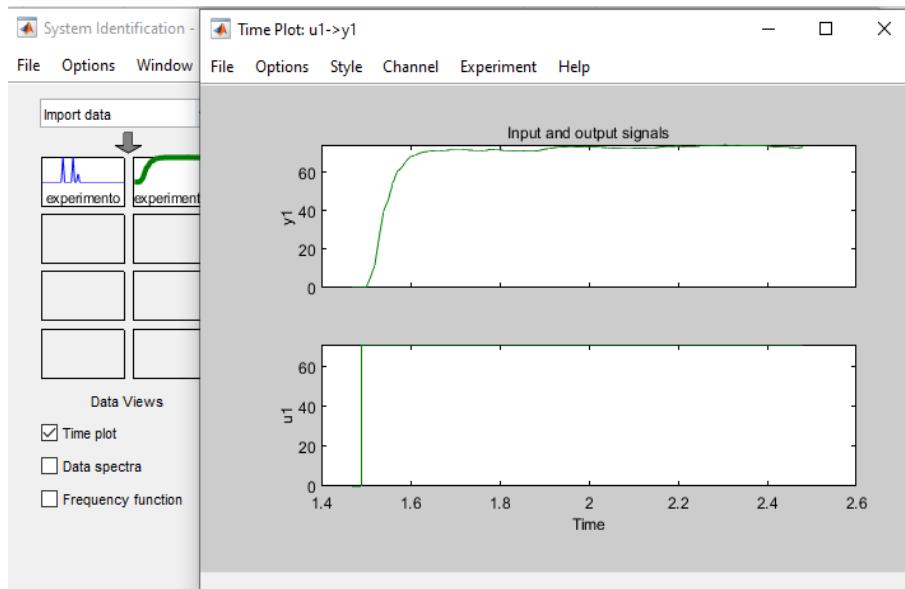


Figura 172. Resultado señal filtrada. [Autores]

Hasta este momento ya se hizo el prefiltrado necesario para crear la estructura en tiempo continuo o discreto. Observe que hay 8 espacios para las señales del experimento, en **Working Data** se arrastra la señal que se trabajara, en **Estimate** se escoge que se desea hacer con la señal y en la generación del modelo se tienen 16 espacios para crear diferentes modelos.

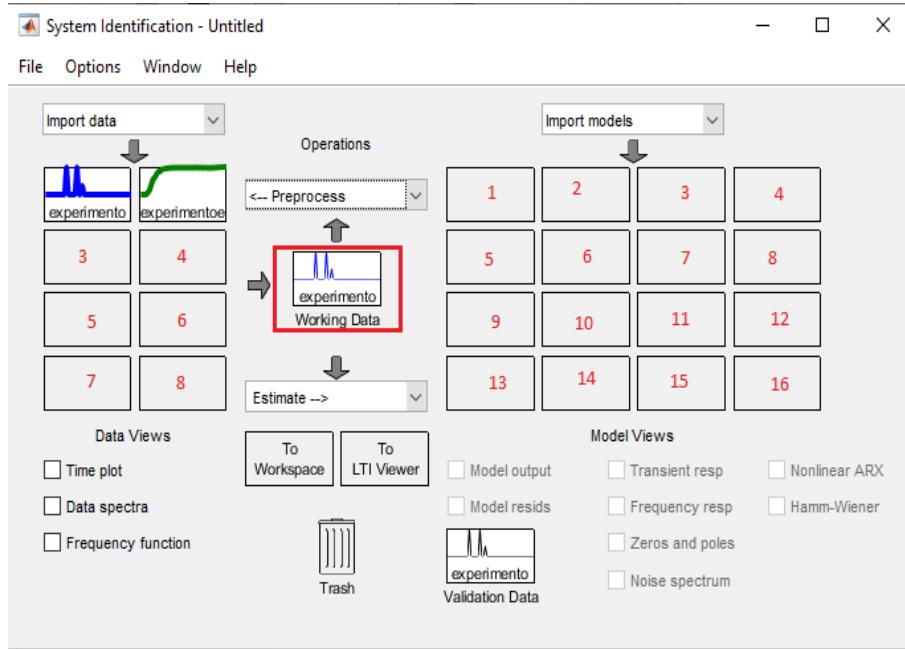


Figura 173. Interfaz IDENT. [Autores]

En la imagen anterior se observa que en el **Working Data** se tiene la señal sin selección de rango, por ende debe arrastrar esta señal ahí y a la caja **Validation Data**.

Ahora se procederá a crear la estructura en continuo, seleccionando la opción “Process Models”

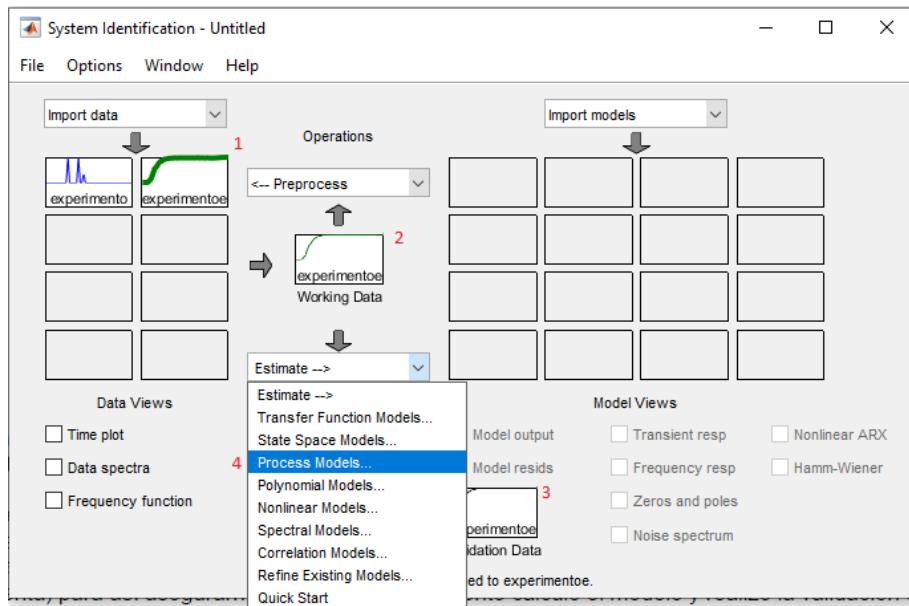


Figura 174. Trabajar con la señal deseada. [Autores]

2.1. Cálculo de parámetros

En el caso de la estructura continua, vamos a calcular los parámetros de la función del sistema de la planta. Tenga en cuenta que, si diseña la estructura continua, indirectamente esta calculando el modelo de la planta, esta es diferente al diseño del controlador, si no es clara esta nota, observe la Figura 203 en la cual se puede apreciar el modelo de la planta y el modelo del controlador en lazo cerrado y que obedece la estructura básica de un controlador en lazo cerrado, ver Figura 192.

En la ventana del Process Models seleccione la cantidad de polos, un retraso y al finalizar oprima **Estimate**. Recuerde el polinomio del denominador debe ser mayor o igual que el del numerador, es decir el orden de los polos en la función de transferencia debe ser mayor o igual que el orden de los ceros.

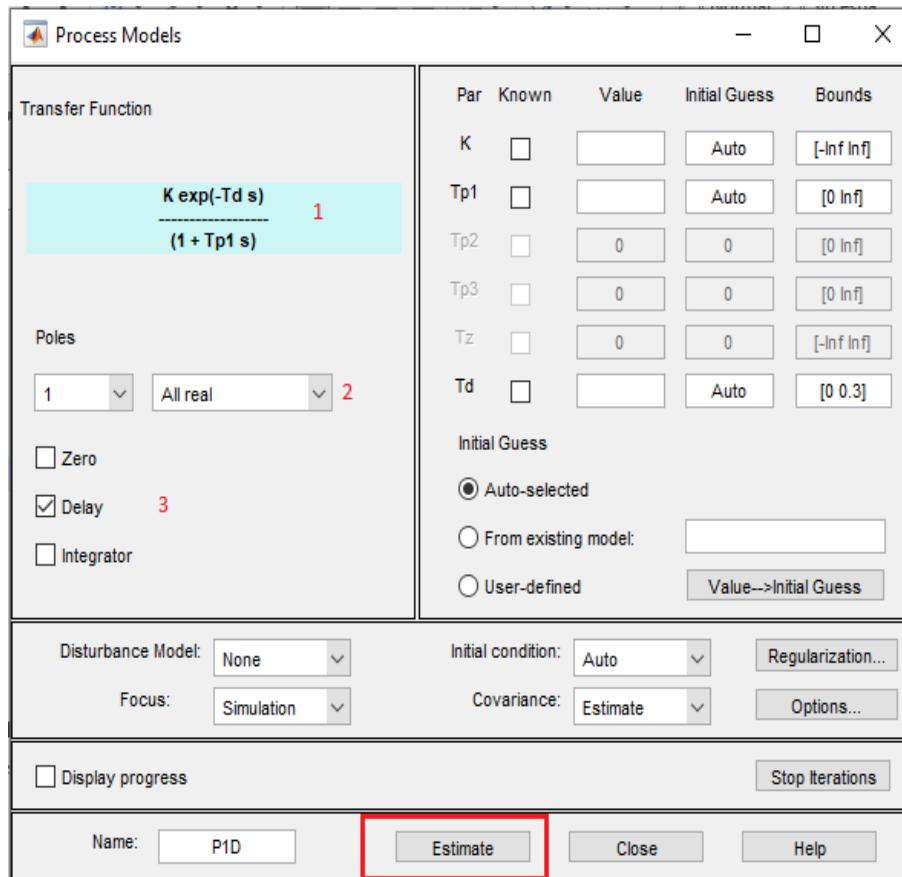


Figura 175. Process Models para calcular parámetros. [Autores]

Al estimar la función de transferencia continua, se observa que en la salida del toolbox se creo y podemos ver sus parámetros al oprimir doble click.

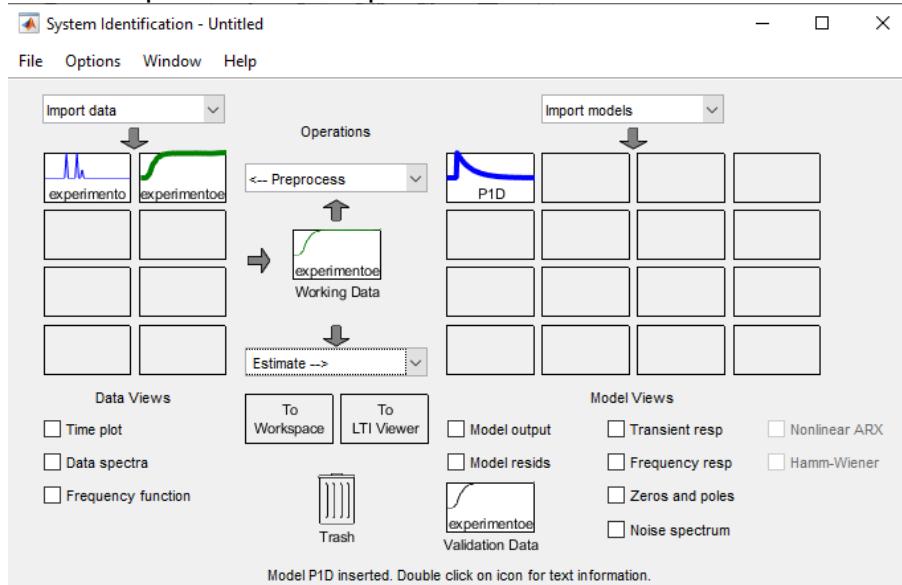


Figura 176. Obtención del modelo continuo

En el caso del modelo (P1D) se observa una función de transferencia con un polo, un retraso y estos valores se observan como K_p , T_p y T_d :

Kp = 1.0217
Tp1 = 0.033467
Td = 0.02437

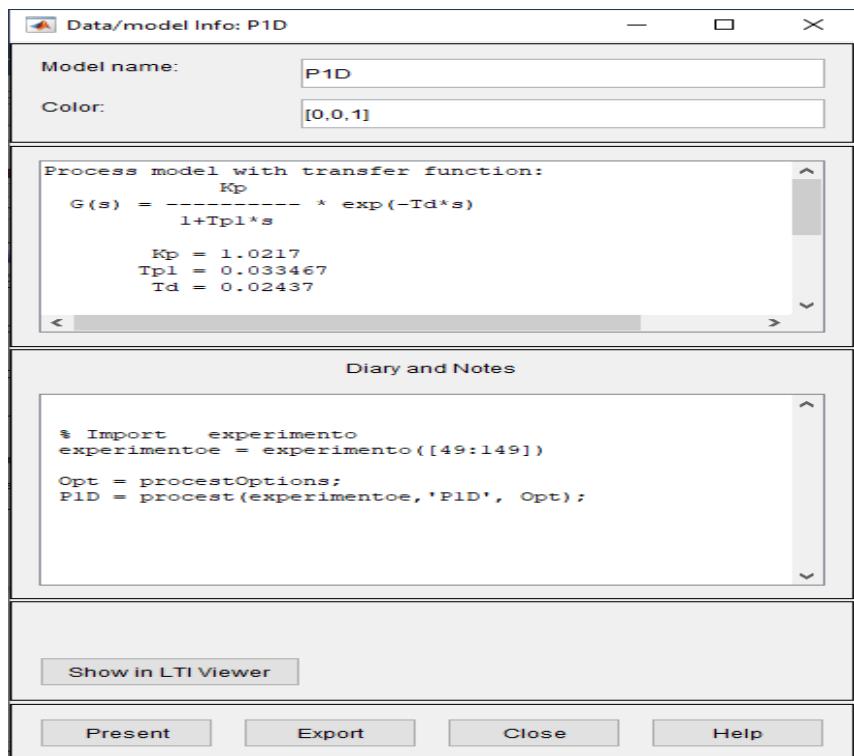


Figura 177. Visualización de los parámetros en el modelo. [Autores]
MATLAB usa mínimos cuadrados y regresión lineal para el cálculo de estos valores.

2.2. Validación experimental

En la Figura 176 seleccione en la salida la casilla **Model Output** de esta forma puede visualizar y validar el modelo creado de mejor manera, como se observa a continuación se ve el % de correlación del modelo, entre más cercano a 100 mejor será el resultado.

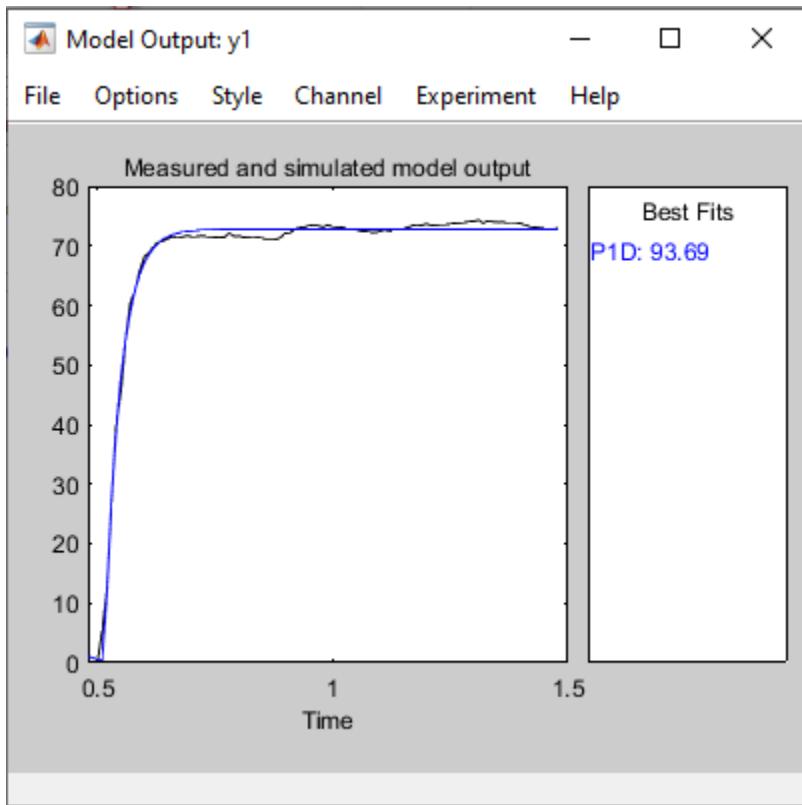


Figura 178. Validación del modelo en continuo. [Autores]

Con este proceso de la sección 2, se culmina el diseño de la estructura continua. Ahora se desarrollará el paso a paso para estructuras discretas.

3. Diseño de la estructura Discreta

En la sección anterior se seleccionó la opción **Process Models** para tiempo continuo, en este caso vamos a seleccionar **Transfer Function Models**. Luego se despliega la ventana de la Figura 180 donde se procede a configurar la estructura dicreta.

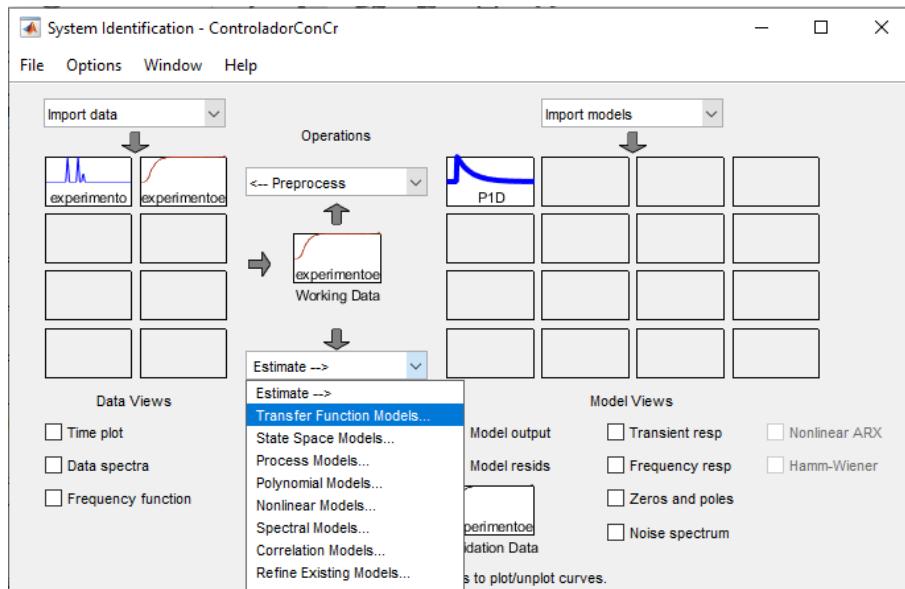


Figura 179. Diseñar Estructura discreta. [Autores]

Para este caso se ha seleccionado 1 polo, 1 cero, el controlador a de ser en tiempo discreto y hemos inyectado un Delay de 3 ya que nuestra señal paso posee un pequeño retardo con respecto a la señal de entrada.

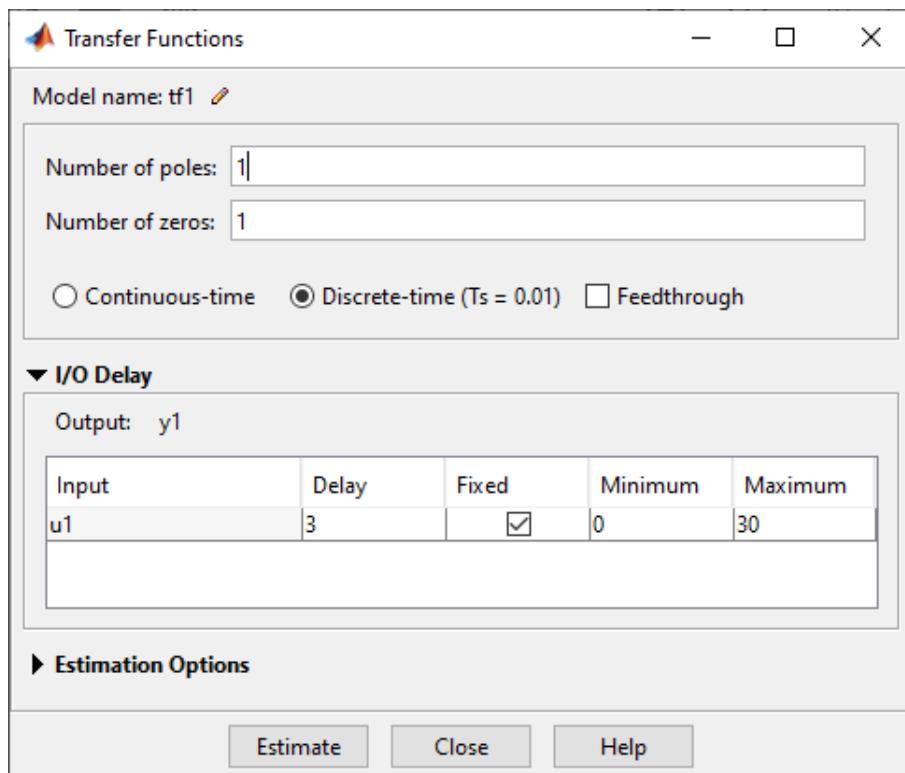


Figura 180. Configurar los parámetros. [Autores]

Al estimar este modelo se observa en la siguiente figura un breve resumen de todo el modelo creado en IDENT.

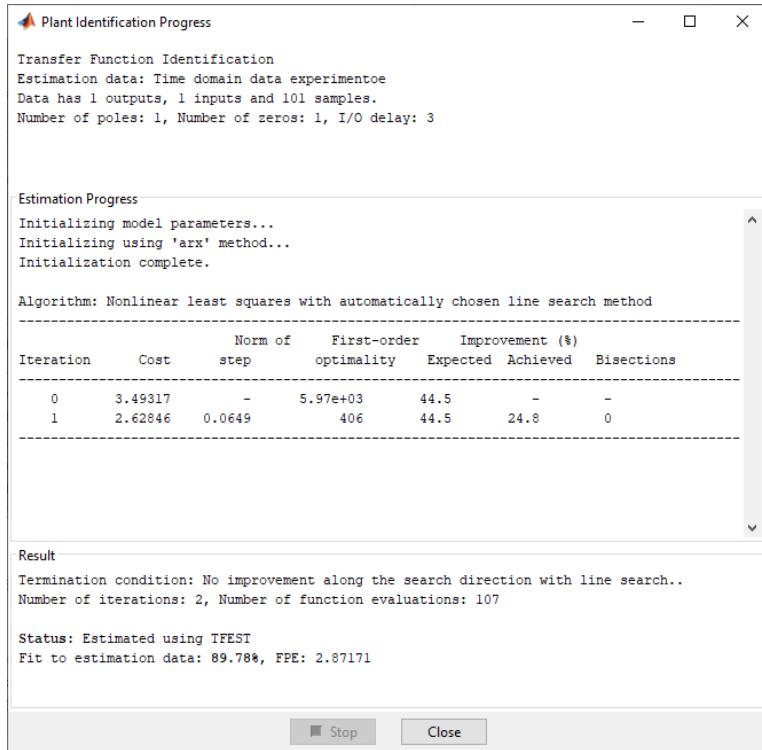


Figura 181. Resumen de la estructura creada. [Autores]

Por ultimo observamos una comparación entre el diseño continuo (señal azul) y discreto (señal verde) con respecto a la señal del experimento (señal negra). Ademas de observar el procentaje de correlacion en ambos modelos creados, Figura 183.

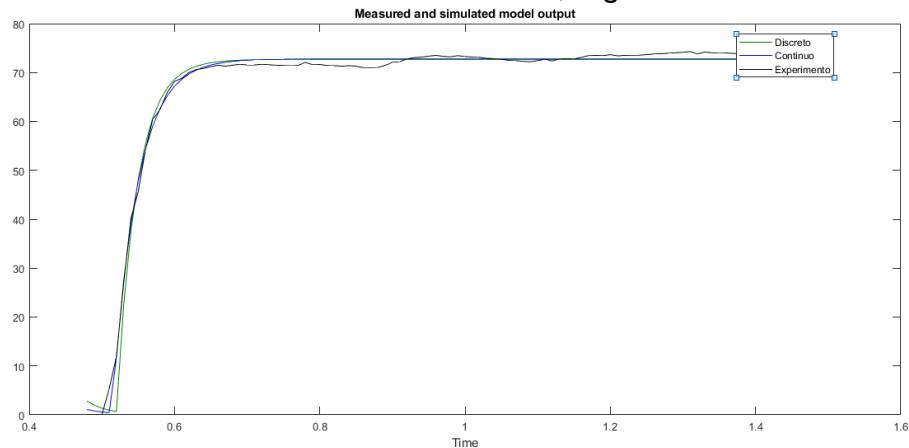


Figura 182. Visualización de las dos estructuras creadas. [Autores]

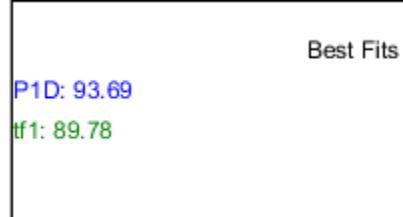


Figura 183. Porcentaje de estimación en modelos. [Autores]

4. Selección del mejor modelo (¿Continuo o Discreto?)

Ya hemos diseñado un modelo continuo al igual que uno discreto, pero ¿cuál ha de ser el más apropiado para llevarlo a SISOTool y así desarrollar el controlador?

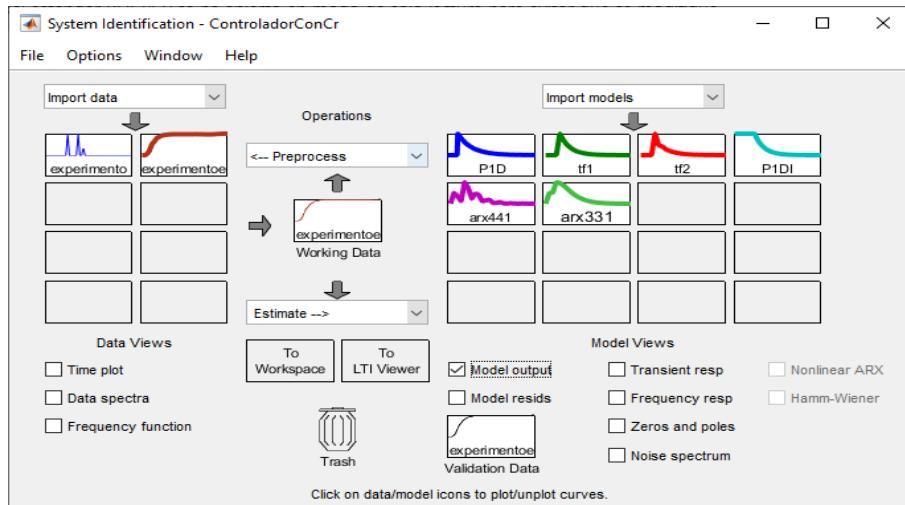


Figura 184. Creación de distintos modelos. [Autores]

Observe en la figura anterior la creación de distintos modelos, se diseñaron dos modelos continuos, dos discretos y dos arx. Luego al graficar la respuesta de cada modelo, se observa

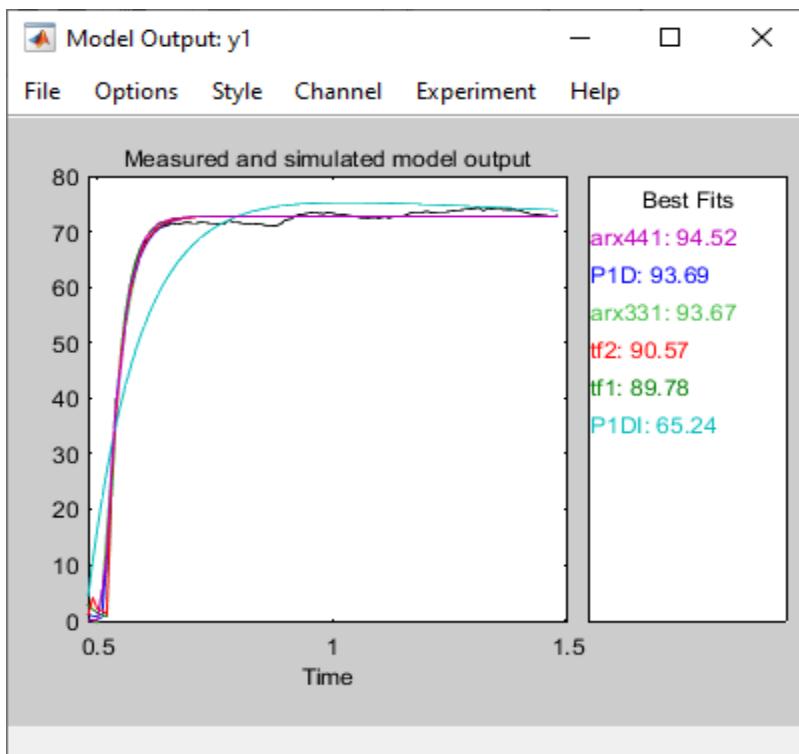


Figura 185. Distintos porcentajes de estimación. [Autores]

El modelo arx441 el cual puede verse en detalle sus características en la Figura 186 tiene un mejor porcentaje de correlación, siempre tenga en cuenta que el mejor modelo sera el que mejor porcentaje de correlación tenga (ideal 100%) y que el orden de la función de transferencia sea el menor (orden 1 sera mejor que un orden 2,3,4) teniendo esta

consideración se observa que sin duda el mejor modelo es **P1D** ya que tiene un porcentaje de 93.69 % y es de primer orden.

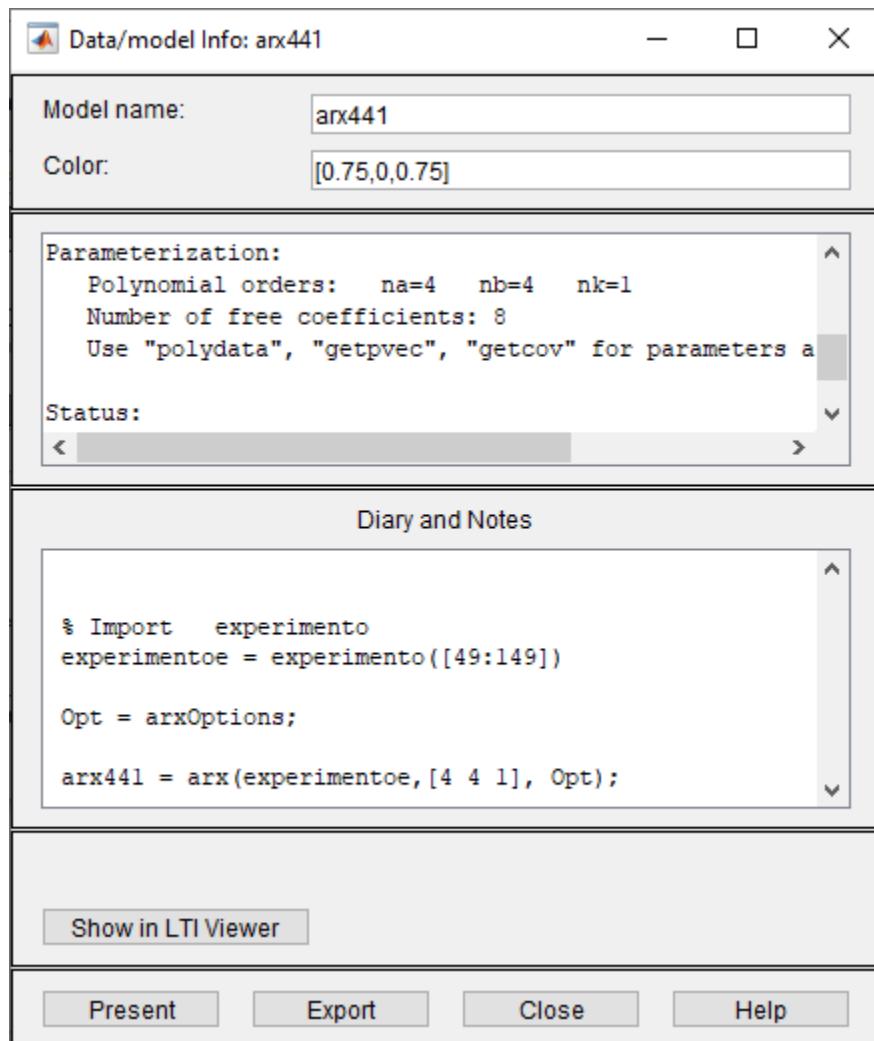


Figura 186. Orden del Modelo ARX creado. [Autores]

Por otro lado, el Modelo ARX es de 4 orden y su porcentaje es de 94.52% de correlación, eso es tan solo de 0.83% con respecto al modelo continuo **P1D** aunque su orden es muchísimo más grande por lo cual la mejor opción es **P1D**.

Aquí se termina el proceso para encontrar el modelo matemático el cual muestra el comportamiento físico de la planta y ahora se procede a diseñar el controlador tomando de base el modelo continuo que se generó.

5. Diseño del controlador

5.1. SISOTOOL

Antes de llevar el modelo a SISOTOOL para diseñar el controlador por Root-Locus, debemos hacer unos pasos previos. En la Figura 187 se exportara el modelo continuo diseñado (P1D) al workspace de MATLAB.

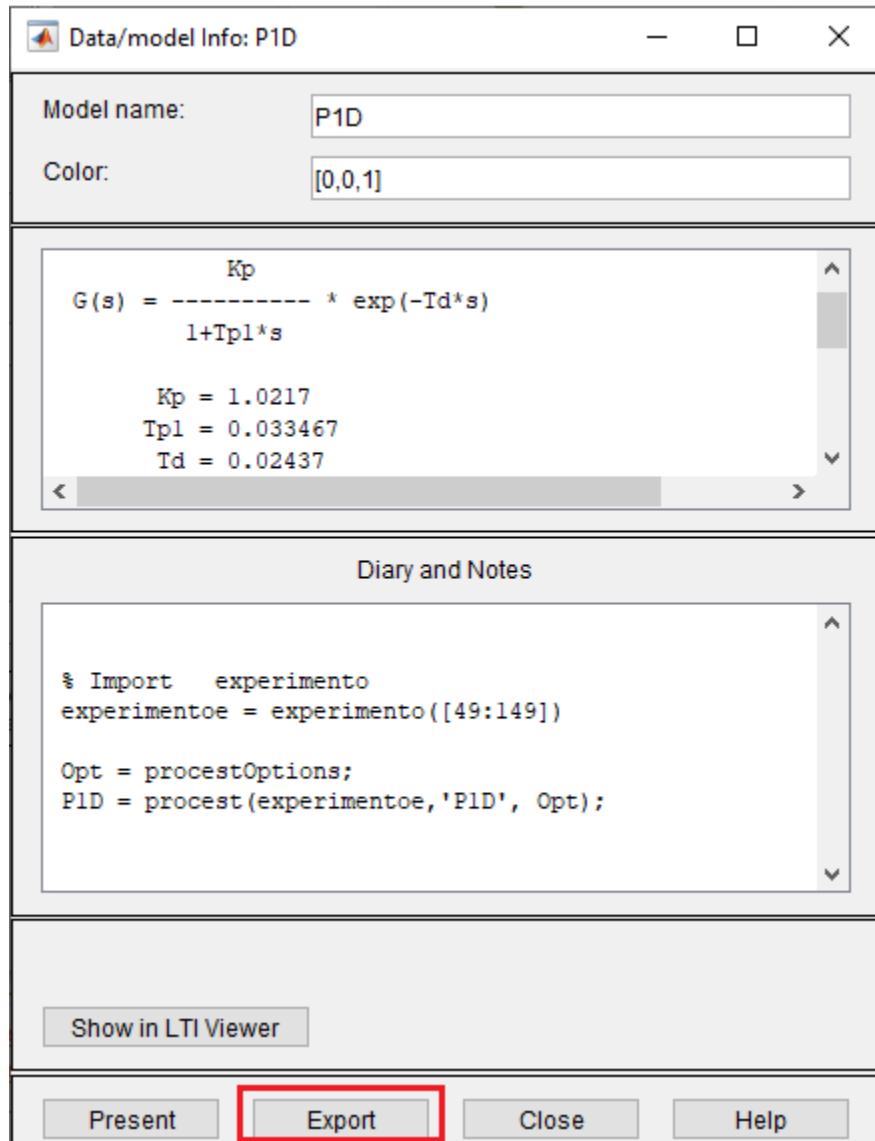


Figura 187. Exportar el Modelo creado en tiempo continuo. [Autores]

En el Command Window podemos observar el modelo continuo al llamarlo en esta ventana

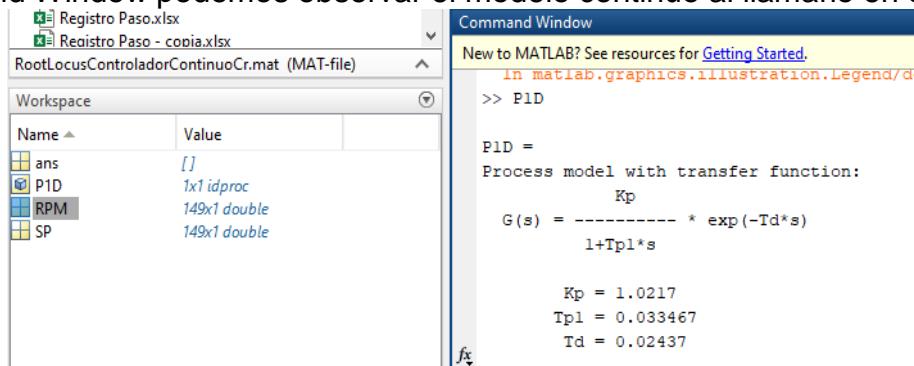


Figura 188. Visualizar el modelo exportado en Command Window. [Autores]

Al exportar el modelo continuo debemos saber que:

1. Al exportar el modelo a sisotool no se puede directamente, ya que esta herramienta no identifica los retrasos de las funciones de transferencia continuo (ver linea 6)
2. Luego debemos discretizar el modelo continuo con delay por medio del retenedor de orden zero (ZOH), observe la linea 8 de la siguiente figura, el "0.01" es el tiempo de muestreo.

```

Editor - C:\Users\USER\Documents\Decimo Semestre\Proyecto de Grado\Lecturas PG\PID\Sintonizacion PID\
ControladorContinuoCr.m + 

1 %<-- LINEAS DE CÓDIGO PREVIAS A SISOTOOL(Controlador continuo)
2 %Tomar el controlador continuo y discretizarlo por medio
3 %del retenedor de orden cero (ZOH) ade más debemos colocar
4 %el retraso que tiene la señal continua ya que sisotool
5 %no lo identifica
6 Gs=tf(1.0217 ,[0.033467 1], 'inputdelay', 0.02437 );
7 
8 Gs_D=c2d(Gs,0.01, 'zoh');
9

```

Figura 189. Líneas previas a SISOTool. [Autores]

Por ultimo en el Command Window de MATLAB, ejecute el comando SISOTOOL(Gs_D) para abrir la herramienta y cargar de una vez el modelo discretizado de la planta.

Current Folder

- Name
- slprj
- ~\$Registro Paso - copia.xlsx
- Comprobar Controlador.slx
- RootLocusControladorContinuoCr.mat (MAT-file)

Workspace

Name	Value
ans	[]
Gs	1x1 tf
Gs_D	1x1 tf
PID	1x1 idproc
RPM	149x1 double
SP	149x1 double

Editor - C:\Users\USER\Documents\Decimo Semestre\Proyecto de Grado\Lecturas PG\PID\Sintonizacion PID\ControladorContinuoCr.m +

```

%<-- LINEAS DE CÓDIGO PREVIAS A SISOTOOL(CONTROLADOR CONTINUO)
2 %Tomar el controlador continuo y discretizarlo por medio
3 %del retenedor de orden cero (ZOH) ade más debemos colocar
4 %el retraso que tiene la señal continua ya que sisotool
5 %no lo identifica
6 Gs=tf(1.0217 ,[0.033467 1], 'inputdelay', 0.02437 );
7 
8 Gs_D=c2d(Gs,0.01, 'zoh');
9

```

Command Window

New to MATLAB? See resources for Getting Started.

```

>> Gs

Gs =

```

$$\frac{1.022}{\exp(-0.0244s) * \frac{0.03347}{s+1}}$$

Continuous-time transfer function.

```

>> Gs_D=c2d(Gs,0.01, 'zoh');
>> sisotool(Gs_D)
fz >> |

```

Figura 190. Discretizar el modelo continuo antes de SISOTool. [Autores]

5.1.1. Diseño por el lugar geométrico de las raíces (Root locus)

En la figura siguiente observe la interfaz principal de SISOTool, en 1 se observan los valores de nuestros componentes de la estructura, en 2 observamos la función de la planta (recuerde que es el mismo modelo continuo de la sección 2 pero aplicamos ZOH), en 3 se muestra la

función de transferencia del controlador, se encuentra en 1 porque aún no lo hemos diseñado y por ultimo en 4 se aprecia Root Locus y la respuesta del controlador actual.

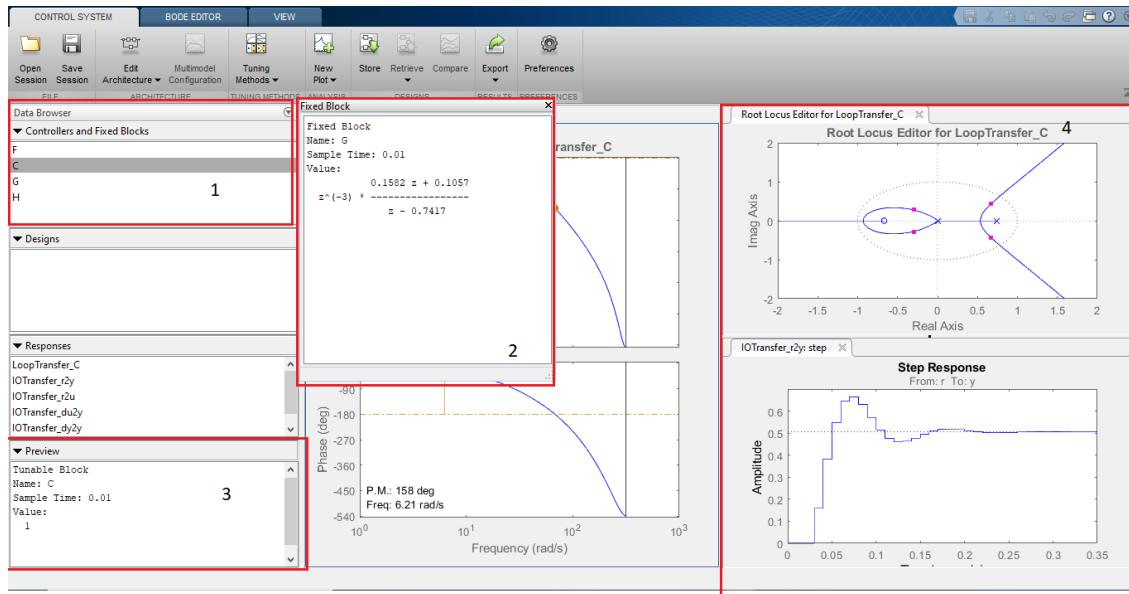


Figura 191. Toolbox de SISOTOOL. [Autores]

Se selecciona la estructura convencional del controlador en lazo cerrado, donde G es planta y C controlador, para esto diríjase a la opción **Edit Architecture**, ver Figura 191.

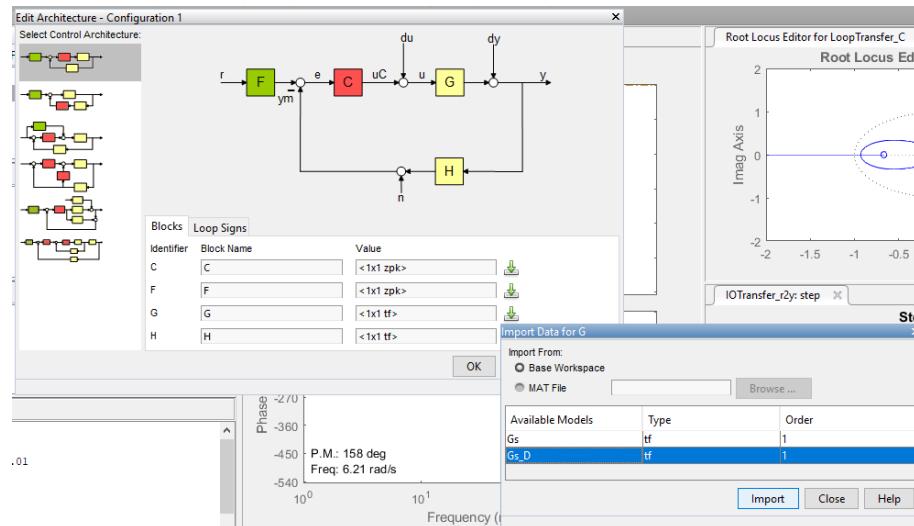


Figura 192. Seleccionar estructura. [Autores]

En la figura anterior se observa como cargar en G la función de la planta (haga esto si no aparece la función de transferencia de su planta (G)). Al diseñar un controlador se deben tener en cuenta los parámetros de diseño, los cuales uno los define.

Sugerimos que usted haga una investigación de estos parámetros de diseño para un controlador en un motor DC:

Máximo sobre impulso: 0.5% (Muy pequeño, contemplado la probabilidad de un sobre impulso)

Error de estado estacionario: se desea no error de estado estacionario

Tiempo de estabilización: 0.24 ms (menor al tiempo de estabilización en lazo abierto)

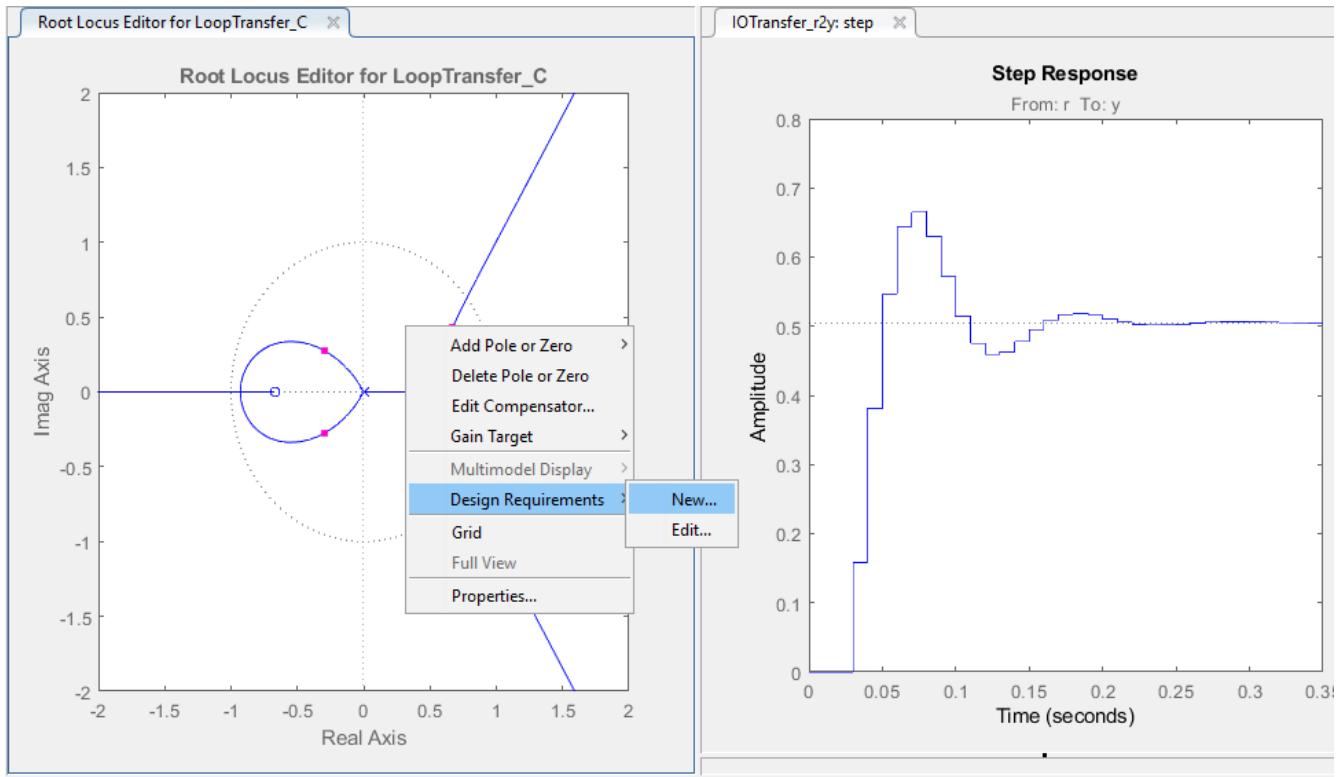


Figura 193. Crear requerimientos para el diseño del controlador. [Autores]

Configure los requisitos del controlador en el grafico Root Locus. Observe que cada vez más la zona apropiada de trabajo va limitandose según los requisitos.

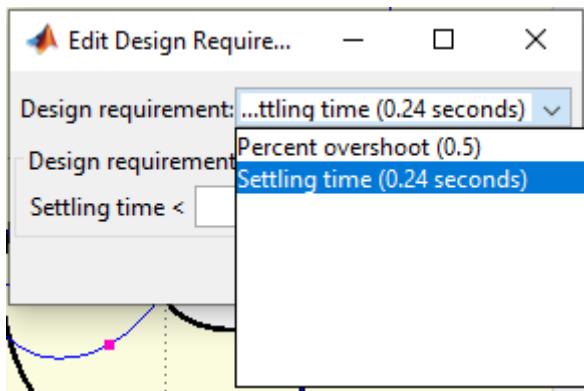


Figura 194. Requerimientos del diseñador. [Autores]

Para terminar la configuración basica y que así sea más facil diseñar el controlador, es necesario configurar la grafica de salida, observe las características mencionadas en los diseños y como en la grafica de la salida podemos verlas con mejor detalle, ver Figura 196.

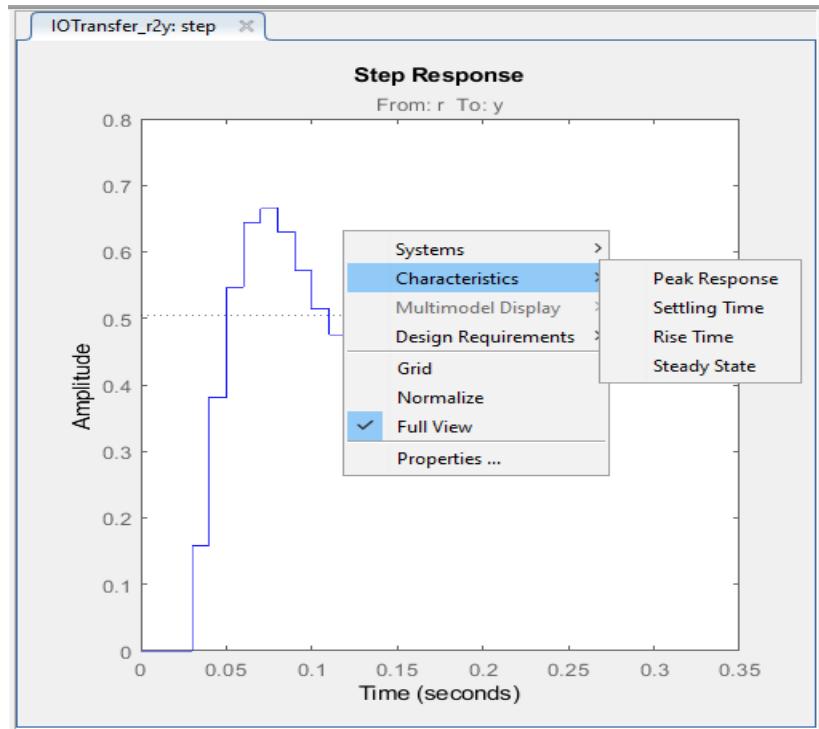


Figura 195. Configurar la visualización de la señal. [Autores]

Hasta aquí, debes tener la misma configuración del sisotool como se muestra a continuacion:

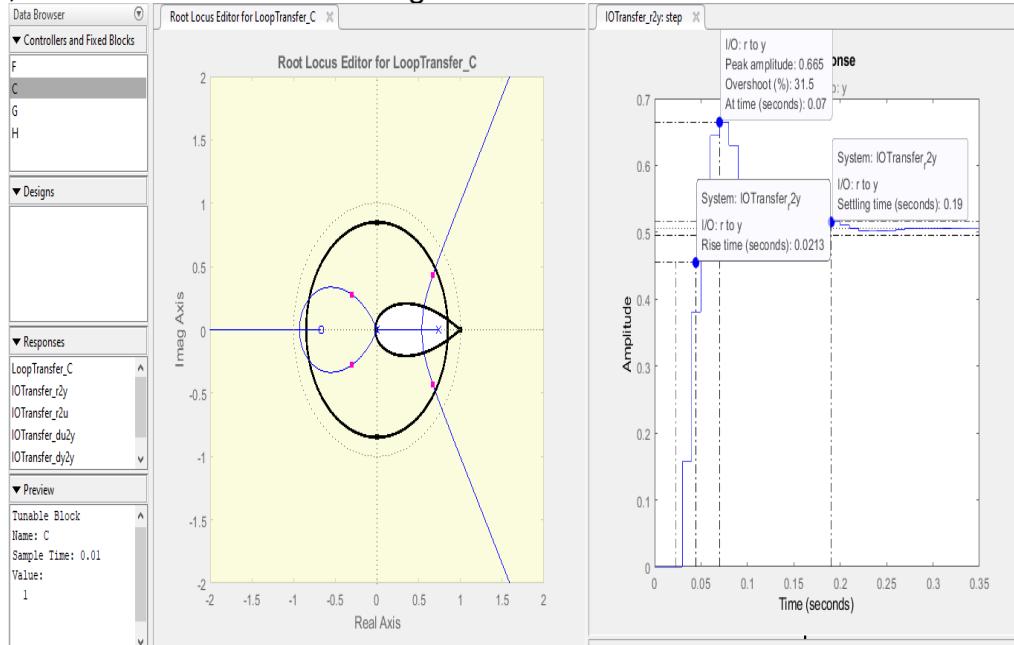


Figura 196. Espacio de trabajo para Root-Locus. [Autores]

Como se ve en el paso 1, seleccione el controlador y editelo. Antes de empezar a agregar polos o ceros observe que el Controlador tiene un valor de 1. Al agregar algo, este valor cambiara, observe la Figura 198.

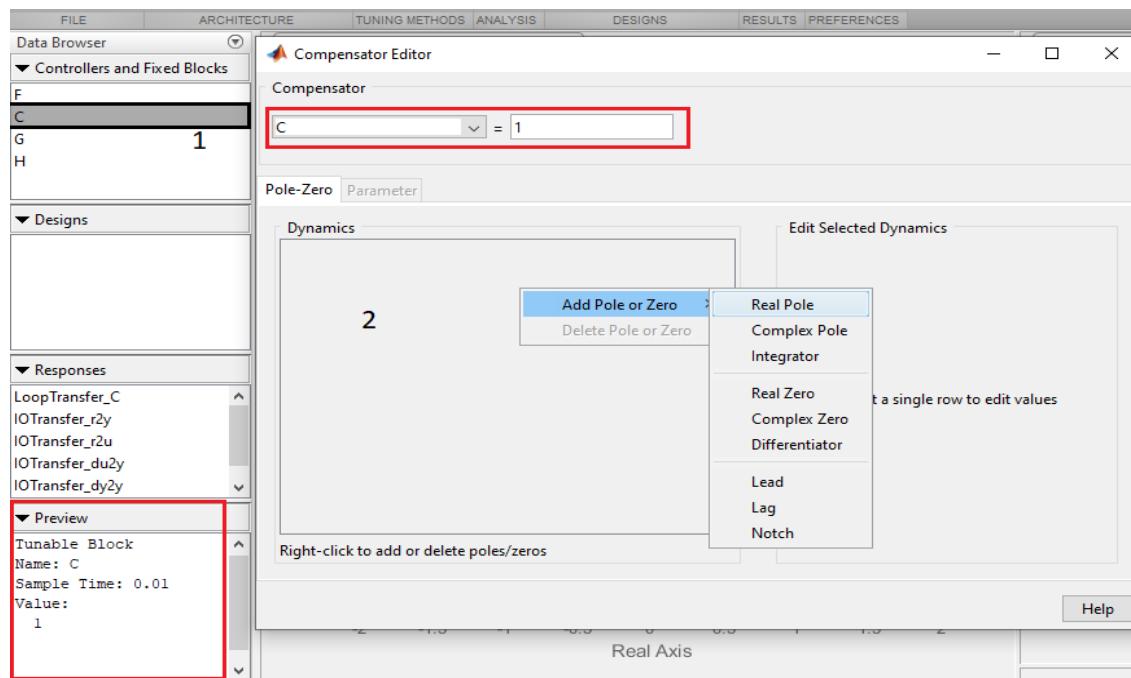


Figura 197. Diseño del controlador. [Autores]

Tenga en cuenta la amplitud de la respuesta (observe como se encuentra en 0.5), esto quiere decir que atenúa la respuesta del controlador por tal razón déjelo en 1. Al mismo tiempo debe estar observando los 3 parámetros de diseño los cuales son rise time, overshoot, steady state. Cada vez que usted cambie los valores del controlador, la respuesta será distinta, recomendamos no usar polos/ceros imaginarios seria dimensionar aún más el controlador discreto de un motor DC.

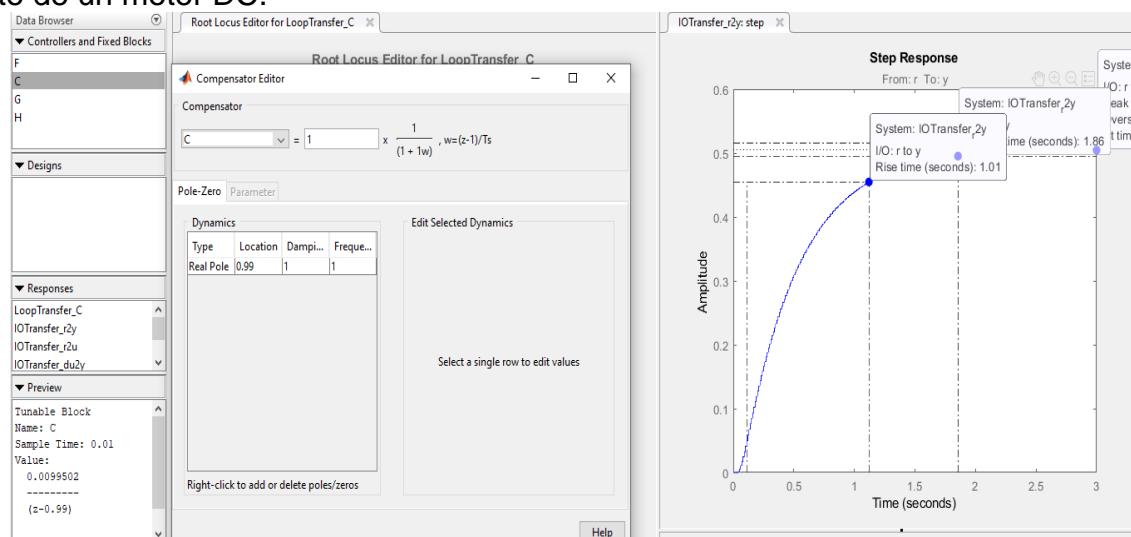


Figura 198. Diseño del controlador agregar polo real. [Autores]

La función de transferencia que los autores han logrado obtener se observa en la siguiente figura, si desea puede copiar dichos valores.

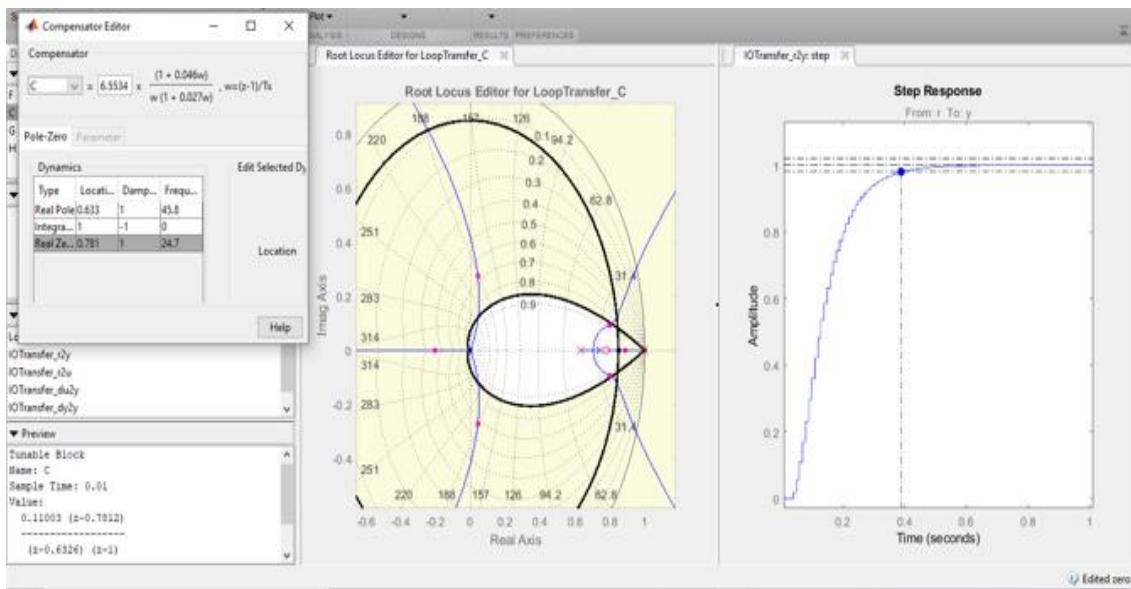


Figura 199. Configuración final del controlador diseñado. [Autores]

Observe como en la Figura 200 se aprecia con mejor detalle la respuesta y el grafico de diseño, los parametro de tiempo de asentamiento y maximo sobre impulso se cumplen. Se observa que el sistema tiene un pequeño impulso.

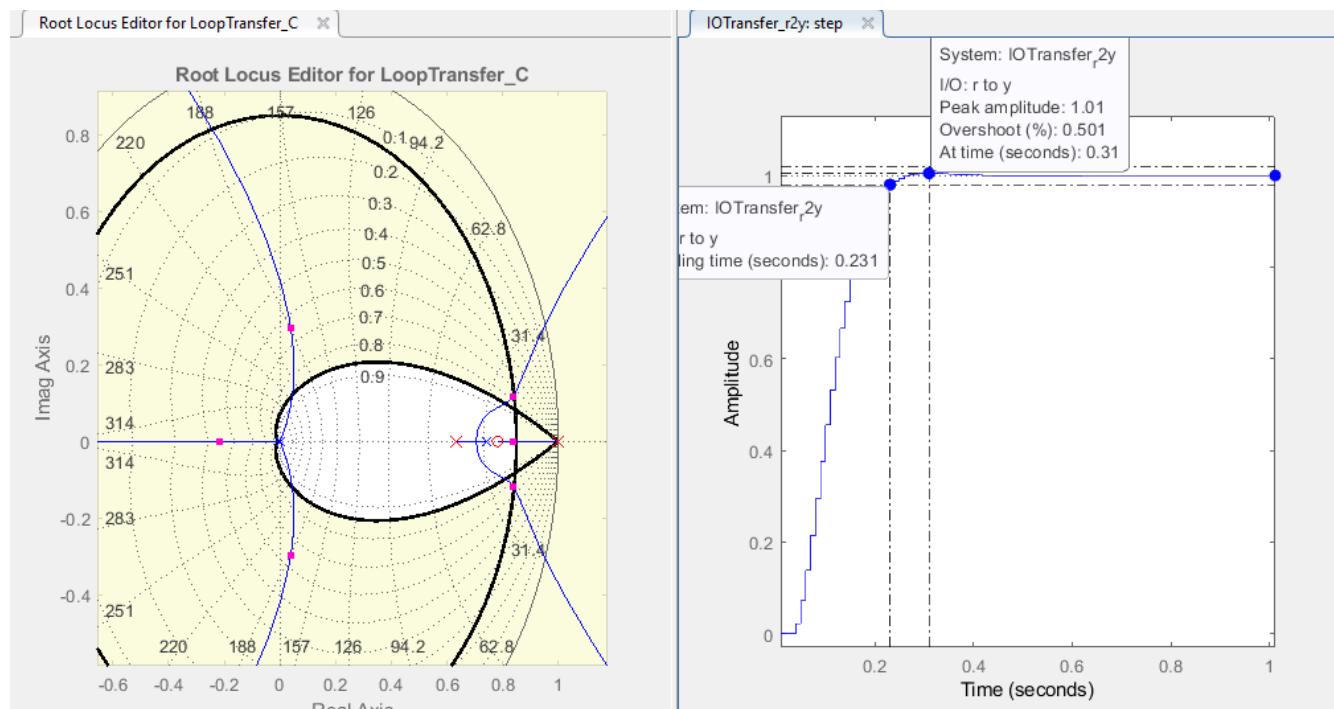


Figura 200.Root Locus controlador. [Autores]

Por último, puede exportar el diseño del controlador a la consola de Matlab, ya que en la sección 6 validaremos el controlador en Simulink. La función de transferencia del controlador discreto se observa en la parte inferior izquierda de la imagen anterior.

5.1.2. Elaboración pseudocódigo en Lenguaje estructurado

Antes de realizar el pseudocódigo es necesario despejar la función de transferencia del controlador discreto diseñado para así pasarlo a memorias e implementar el código en Codesys. El controlador PID está definido por la función de transferencia de la ecuación (1).

$$C(z) = \frac{0.11003(z - 0.7812)}{(z - 0.6326)(z - 1)} \quad (1)$$

Represente en primer lugar la función de transferencia como una ecuación salida sobre entrada

$$C(z) = \frac{Y(z)}{E(z)} = \frac{0.11003(z - 0.7812)}{(z - 0.6326)(z - 1)} \quad (2)$$

Aplique ley distributiva en el numerador y denominador como en las ecuaciones 3-4.

$$\frac{Y(z)}{E(z)} = \frac{0.11003z - 0.08595}{(z - 0.6326)(z - 1)} \quad (3)$$

$$\frac{Y(z)}{E(z)} = \frac{0.11003z - 0.08595}{z^2 - 1.6326z + 0.6326} \quad (4)$$

El polinomio anterior tiene potencias positivas, por tal razón debe escribirse en potencias negativas para así realizar la transformada Z inversa.

$$\frac{Y(z)}{E(z)} = \frac{0.11003z - 0.08595}{z^2 - 1.6326z + 0.6326} * \frac{z^{-2}}{z^{-2}} \quad (5)$$

En la ecuación 6 se observa el resultado de la fracción al ser multiplicado por potencias negativas de z

$$\frac{Y(z)}{E(z)} = \frac{0.11003z^{-1} - 0.08595z^{-2}}{1 - 1.6326z^{-1} + 0.6326z^{-2}} \quad (6)$$

Ahora se debe despejar Y(z), como se observa en las siguientes ecuaciones.

$$Y(z)[1 - 1.6326z^{-1} + 0.6326z^{-2}] = E(z)[0.11003z^{-1} - 0.08595z^{-2}] \quad (7)$$

$$Y(z) - 1.6326z^{-1}Y(z) + 0.6326z^{-2}Y(z) = 0.11003z^{-1}E(z) - 0.08595z^{-2}E(z) \quad (8)$$

Al despejar la salida se obtiene la ecuación (9)

$$Y(z) = 1.6326z^{-1}Y(z) - 0.6326z^{-2}Y(z) + 0.11003z^{-1}E(z) - 0.08595z^{-2}E(z) \quad (9)$$

Ya teniendo dicha ecuación, se procede a aplicar la transformada Z inversa como se observa en la ecuación (10)

$$Z^{-1}\{Y(z) = 1.6326z^{-1}Y(z) - 0.6326z^{-2}Y(z) + 0.11003z^{-1}E(z) - 0.08595z^{-2}E(z)\} \quad (10)$$

$$Y[n] = 1.6326 * Y[n - 1] - 0.6326 * Y[n - 2] + 0.11003 * E[n - 1] - 0.08595 * E[n - 2] \quad (11)$$

La ecuación en diferencias presentada en la ecuación (12) debe pasarse a memorias para así elaborar el pseudocódigo.

$$Y[n] = 0.11003 * E[n - 1] - 0.08595 * E[n - 2] - 0.6326 * Y[n - 2] + 1.6326 * Y[n - 1] \quad (12)$$

En la siguiente tabla se presenta la creación de las variables para elaborar el código en Codesys.

Tabla 8. Asignación de memorias. [Autores]

Variable	Memoria
$Y[n]$	M1
$Y[n - 1]$	M2
$Y[n - 2]$	M3
$E[n - 1]$	M4
$E[n - 2]$	M5

Reemplazando las memorias en la ecuación (12) se obtendrá, la ecuación (13)

$$M1 = 0.11003 * M4 - 0.08595 * M5 - 0.6326 * M3 + 1.6326 * M2 \quad (13)$$

Para acabar con la ecuación en diferencias, se definen las variables que tendrán el valor de los coeficientes que acompañan las memorias.

$$M1 = Q0 * M4 - Q1 * M5 - Q2 * M3 + Q3 * M2$$

En Codesys se realizó el siguiente código:

```

A Acciones_Secundarias.Controlador_active X Datos_CSV Output In
1 //Controlador obtenido con la señal paso
2 M6:=0.0;//E[n]
3 Q0:=0.1103;
4 Q1:=0.08595;
5 Q2:=0.6326 ;
6 Q3 := 1.6326;
7 TS(IN:=Inicio_Ts , PT:=T#0.01S);
8 IF TS.Q THEN
9     GVL.Error:= GVL.Set_point_RPM-GVL.RPM;
10    M6:=GVL.Error;
11    //Y[n]:=0.1103E[n-1] - 0.08595E[n-2] - 0.6326Y[n-2] + 1.6326Y[n-1]
12    M1:= Q0*M4 - Q1*M5 - Q2*M3+ Q3*M2;
13    //Memorias para los desplazamientos
14    M5:=M6;
15    M4:=M5;
16    M3:=M2;
17    M2:=M1;
18    GVL.Out_RPM:=(M1);
19    //Limites de funcionamiento
20    IF GVL.Out_RPM <= 0 THEN
21        GVL.Out_RPM:=0;
22    END_IF
23    IF GVL.Out_RPM >= 116 THEN
24        GVL.Out_RPM:=116;
25    END_IF
26    //IF GVL.Out_RPM > 0 AND M1 < 116 THEN
27        GVL.Out_RPM:=M1;
28    //END_IF
29    Inicio_Ts:=FALSE;
30 END_IF
31

```

Figura 201. Pseudocódigo parte 1. [Autores]

En la figura anterior se observa el pseudo código, en términos generales se debe crear las memorias, las memorias deben ser inicializadas en cero, asigne las memorias actuales a una memoria anterior como se observa en la línea 17 ya que a la memoria retrasada una posición se le asigna su estado anterior. Observe que en la ecuación (42) no disponemos de un valor E[n], solo tenemos dos atrasos de la entrada (error, recuerde el esquema de un sistema básico en lazo cerrado con controlador), pero debemos definir ese error en M6. Otra cosa a tener en cuenta es que debemos crear un timer que actualizara las memorias y salida (GVL.Out_RPM) cada 10 ms, ya que este es nuestro tiempo de muestreo y tiempo del controlador diseñado.

```

31
32      IF  M1 >= 2000 THEN
33          M1:=0.0; //y[n]
34          M2:=0.0; //Y[n-1]
35          M3:=0.0; //Y[n-2]
36          M4:=0.0; //E[n-1]
37          M5:= 0.0;//E[n-2]
38          M6:=0.0;//E[n]
39          Q0:=0.1103;
40          Q1:=0.08166 ;
41          Q2:=0.6326 ;
42          Q3 := 1.6326;
43      END_IF
44      IF GVL.M_banda = FALSE THEN
45          M1:=0.0; //y[n]
46          M2:=0.0; //Y[n-1]
47          M3:=0.0; //Y[n-2]
48          M4:=0.0; //E[n-1]
49          M5:= 0.0;//E[n-2]
50          M6:=0.0;//E[n]
51          Q0:=0.1103;
52          Q1:=0.08166 ;
53          Q2:=0.6326 ;
54          Q3 := 1.6326;
55      END_IF
56

```

Figura 202. Pseudocódigo parte 2. [Autores]

Por último, en la Figura 202 se limita la salida M1 para que no siga acumulándose este valor y provoque un fallo en la tarjeta y la ultima estructura condicional es obligatoria porque cada vez que se apague la banda transportadora y volvamos a ejecutar el controlador PID las memorias deben estar en cero. En los anexos encontrara el código implementado en Codesys.

6. Validación del controlador (Simulink)

Una forma de validar el controlador discreto diseñado por Root-Locus es por medio de Simulink, para esto se deben crear unos cuantos scopes, generadores de señales paso, paso cuadrada y pseudoaleatoria para así observar la respuesta del controlador

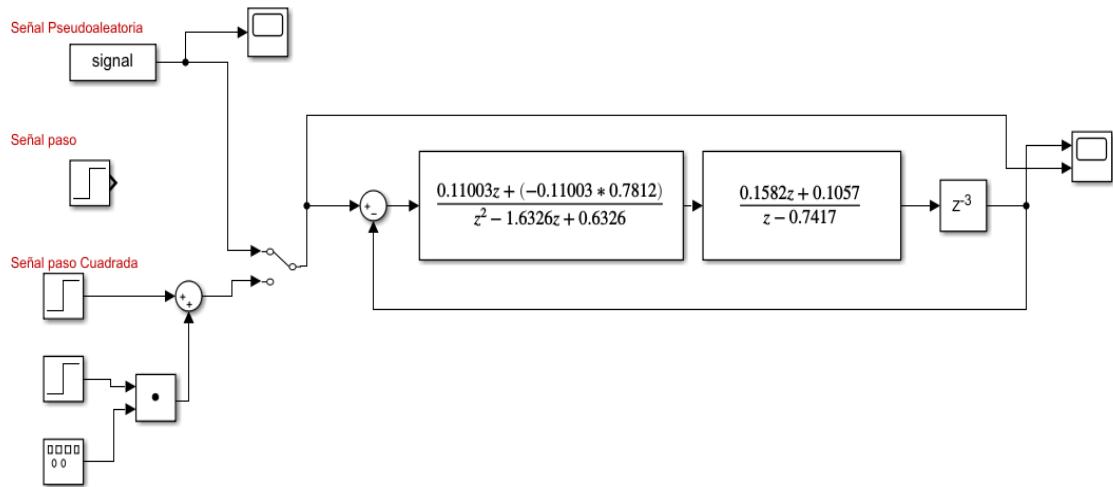


Figura 203. Validación del controlador creado en Simulink. [Autores]

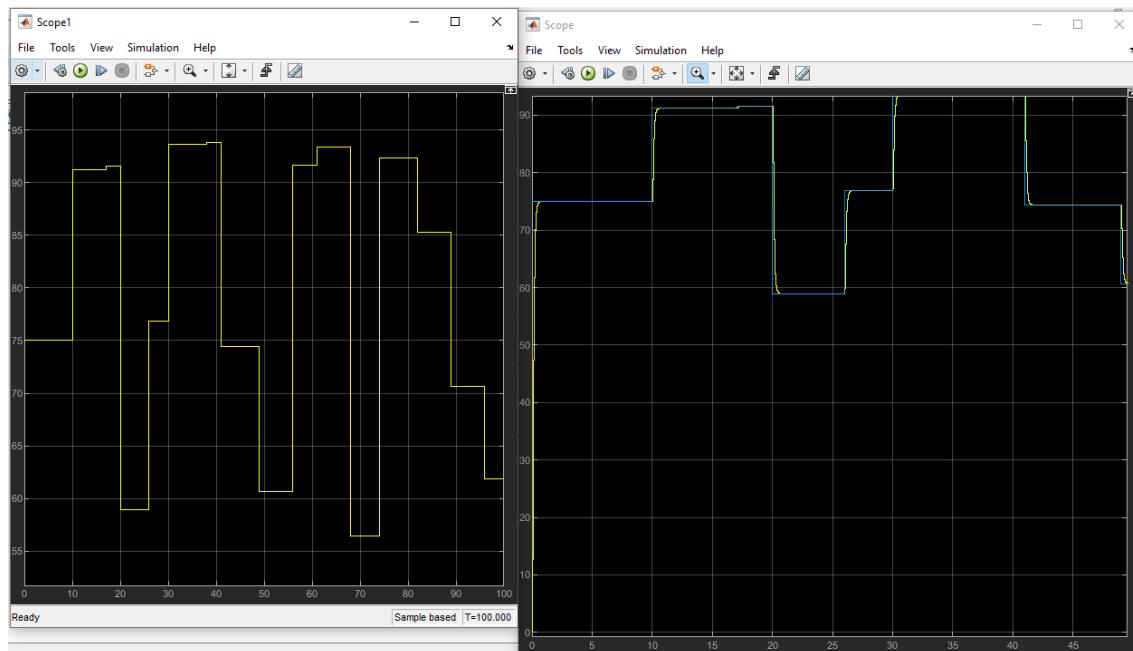


Figura 204. Visualización de la respuesta del controlador

Esta práctica se realizó con una señal paso, el sistema debe estimularse con señales más complejas, ya que al configurar el sistema con la señal paso pude ir aumentando la dificultad y con cada señal se obtiene un parámetro para que el controlador mejore y sea mucho más exacto. Con la señal paso se pudo calcular lo siguiente:

- Constante del tiempo del sistema
- Ganancia del sistema(K, por medio de IDENT)
- Selección del periodo de muestreo.

	<p>UNIDAD TEMÁtica: Prácticas de laboratorio para la máquina de conteo</p> <p>ACTIVIDAD: Comparación entre controlador PID y Bloque PID en Codesys</p>	
Código: PC005	<input checked="" type="checkbox"/> ONLINE <input type="checkbox"/> OFFLINE	Duración: ---

PC007. Comparación entre controlador PID y Bloque PID en Codesys

Objetivo de la práctica:

A partir del controlador discreto PID diseñado en la práctica **PC006. Identificación y diseño del controlador discreto en Matlab** se busca que el estudiante pueda comparar intuitivamente los resultados de dos controladores diseñados en la maquina contadora de capsulas. Esto fortalecerá el razonamiento de los estudiantes a la hora de escoger que tipo de controlador se adapta mejor a un proceso industrial.

Material Necesario y requisitos para el desarrollo:

-Codesys V3.5 SP16 o superior

Esquema Grafico de la Actividad:

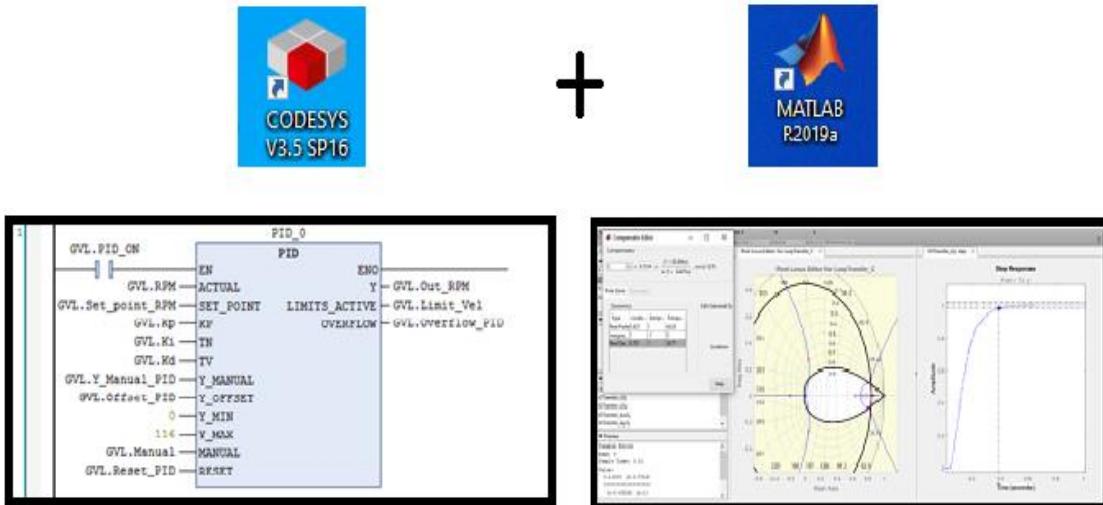


Figura 205. Esquema básico de la actividad a realizar. [Autores]

Requisitos previos:

- **PC006. Identificación y diseño del controlador discreto en Matlab**

Resumen:

La máquina posee dos tipos de regulaciones una por margen de demanda y otra por distancia, en la siguiente tabla se observa su descripción.

Tabla 9. Tipos de regulación para aplicar el PID. [Autores]

Regulación por margen de demanda	Regulación por distancia
En la regulación por margen de demanda se tiene en cuenta los niveles de producción que determine el usuario dentro del MES, es decir que dependiendo del número de lotes y la cantidad de botellas que deban dispensar y salir de la producción, el sistema se ajustara de forma automática, entre más demanda se estipule para el proceso más velocidad tendrá la banda.	El modo de regulación por distancia tiene en cuenta el sensor inteligente y en lugar de basarse en la producción se fija es en la distancia de las botellas dispensadas, entre más lejos se encuentre la botella de la etapa de centrifugado más velocidad será suministra en la banda y a medida que van llegando al punto de llenado este disminuye la velocidad.
Nota: La idea de proceder con dos tipos de regulaciones es que, al momento de utilizar la máquina, ya sea el estudiante o el operario, este pueda analizar y comprobar la aplicación de un PID en un sistema.	

A partir de esta información se busca que el estudiante cree dos tipos de controladores, uno usando el bloque PID de Codesys el cual se sintoniza de forma experimental/intuitiva y el otro PID será un controlador discreto. Se programarán ambos modos de regulación de producción en la máquina, como se ve en Tabla 9 y se observaran gráficamente que respuesta es mejor.

Desarrollo de la Actividad:

1. Creación del bloque PID en Codesys
 - 1.1. Bloque en Codesys y creaciones variables

Para este tipo de implementación del PID es necesario aplicar un bloque de función que representa el comportamiento de este controlador (ver Figura 206). Dicho bloque se encarga de eliminar el error de la entrada y el valor de referencia al cual queremos ajustar la respuesta del controlador, para esto el bloque PID maneja internamente un algoritmo basado en correcciones proporcionales, integrales y derivativas.¹ En configuración se declararon diferentes variables donde se indican los parámetros de funcionamiento, entre ellas se encuentran las constantes del controlador, la entrada y salida del sistema que en este caso son las RPM del motor de la banda, los límites de funcionamiento, entre otros parámetros.

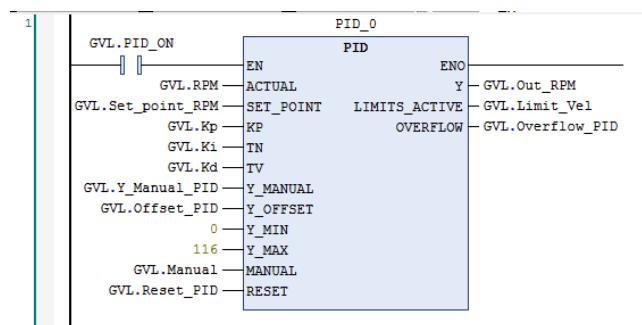
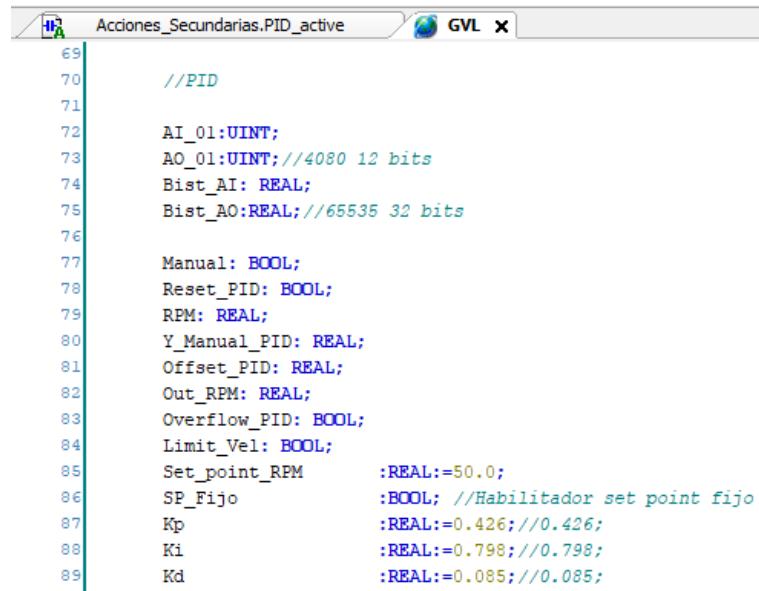


Figura 206. Bloque de función del PID en Codesys. [Autores]

¹ Para más información del bloque PID de codesys ver la página de ayudas: <https://help.codesys.com/>

El manejo de la aplicación del PID en la velocidad del motor de la banda transportadora se emplea un sistema en lazo cerrado, donde se envía una salida de voltaje regulable por medio de un DAC del softPLC y se toma de referencia la velocidad del motor por medio un encoder que con la ayuda del circuito de acondicionamiento se envía a un ADC del softPLC para tener el valor que permita realimentar el sistema. A continuación, se observan las variables creadas en Codesys para el desarrollo del bloque PID. Donde la variable **AI_01** es la entrada(0-65535), **AO_01** la salida(0-4080) y los **Bits_AI** es la entrada en bits y **Bits_AO** es la salida de bits. La entrada máxima llega a 50000 bits por lo cual se pierden 15mil bits perdiéndose 2 voltios.



```

69
70      //PID
71
72      AI_01:UINT;
73      AO_01:UINT;//4080 12 bits
74      Bits_AI: REAL;
75      Bits_AO:REAL;//65535 32 bits
76
77      Manual: BOOL;
78      Reset_PID: BOOL;
79      RPM: REAL;
80      Y_Manual_PID: REAL;
81      Offset_PID: REAL;
82      Out_RPM: REAL;
83      Overflow_PID: BOOL;
84      Limit_Vel: BOOL;
85      Set_point_RPM      :REAL:=50.0;
86      SP_Fijo           :BOOL; //Habilitador set point fijo
87      Kp                :REAL:=0.426;//0.426;
88      Ki                :REAL:=0.798;//0.798;
89      Kd                :REAL:=0.085;//0.085;

```

Figura 207. Variables globales para el Bloque PID. [Autores]

1.2. Rutina de conversiones

Esta rutina es necesaria ya que el SoftPLC por defecto solo puede leer de 0 a 7 voltios, es decir que no trabaja con todo el rango de bits (no tiene toda su área de trabajo para poder leer una señal análoga), por lo cual es necesario que la señal entrante al SoftPLC se maneje en el mismo rango de bits que la salida, para esto se realiza una linealización encargada de generar la equivalencia entre ambos valores

```

A Acciones_Secundarias.Conversiones_active X A Acciones_Secundarias.PID_active GVL
1 //GVL.Bist_AI := (0.0458*GVL.AI_01) + 1348.2;
2 //GVL.RPM:=((GVL.Bist_AI*115)/3656);//Entrada PID
3 IF GVL.PID_ON = TRUE AND GVL.C_PID_ON = FALSE THEN
4   GVL.LED_C_PID_ON:=FALSE;
5   GVL.LED_PID_ON:=TRUE;
6   GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
7   GVL.RPM:=((GVL.Bist_AI*0.049)-75.75); //Entrada ADC
8   GVL.AO_01:=REAL_TO_UINT(GVL.Out_RPM*3800)/115;//Salida PID
9
10 ELSIF GVL.C_PID_ON = TRUE AND GVL.PID_ON = FALSE THEN
11   GVL.LED_C_PID_ON:=TRUE;
12   GVL.LED_PID_ON:=FALSE;
13   GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
14   GVL.RPM:=((GVL.Bist_AI*0.049)-75.75); //Entrada ADC
15   GVL.AO_01:=REAL_TO_UINT(GVL.Out_RPM*3800)/115;//Salida PID
16   Inicio_TS:=TRUE;
17
18 ELSE
19   GVL.LED_C_PID_ON:=FALSE;
20   GVL.LED_PID_ON:=FALSE;
21   IF GVL.Out_Singal_M = TRUE THEN
22     // GVL.AO_01:=0;
23     GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
24     GVL.Bits_In_Signal:=REAL_TO_UINT(GVL.Bist_AI);
25   END_IF
26
27   GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
28   GVL.RPM:=((GVL.Bist_AI*0.049)-75.75); //Entrada ADC
29
30 END_IF
31
32 IF GVL.AO_01 >=4000 THEN
33   GVL.AO_01 :=4000;
34 END_IF
35

```

Figura 208. Código Conversiones PID. [Autores]

Al observar las ecuaciones de linealización, se aprecia que cuando se tenga el valor máximo de bits a la entrada se tendrá el mismo valor de bits a la salida que corresponde a 4080. Las variables **GVL.PID_ON** y **GVL.C_PID_ON** hacen mención a los habilitadores que se usaran para activar cada PID (el bloque o el controlador discreto). En los anexos se encuentra el código de conversiones en estructurado.

1.3. Sintonización PID (Ejemplo)

Para encontrar las variables del modelo matemático que aplica el control PID se realizaron varias pruebas de forma empírica, es decir que el sistema se sometió a varios procesos de producción con los dos tipos de regulaciones definidas al comienzo de la sección y se comprobó cual es el ajuste más óptimo para que los parámetros del tiempo de estabilización fuera pequeño, el maximo sobre impulso tendiera a cero y el error de estado estacionario fuera bajo. En la búsqueda de estas constantes se tuvo en cuenta la variación que causa cada uno de los parametros en el sistema. El control proporcional aumenta la ganancia del sistema es decir aumenta las oscilaciones, reduce el maximo sobre impulso y el tiempo de estabilización. El control integral elimina el error de estado estacionario por lo que incrementa el tiempo de subida, reduce el maximo sobre impulso, disminuye el tiempo de estabilización y mejora el sobre amortiguamiento y por ultimo la acción derivativa aumenta las velocidades de reacción, es decir reduce el maximo sobre impulso, disminuye el tiempo de estabilización y aumenta el ancho de banda. De acuerdo a estos parametros se realizo la búsqueda de las constantes del modelo y a su vez se observava la reaccion del sistema con cada cambio. Un ejemplo de esta prueba se puede ver en la siguiente figura:

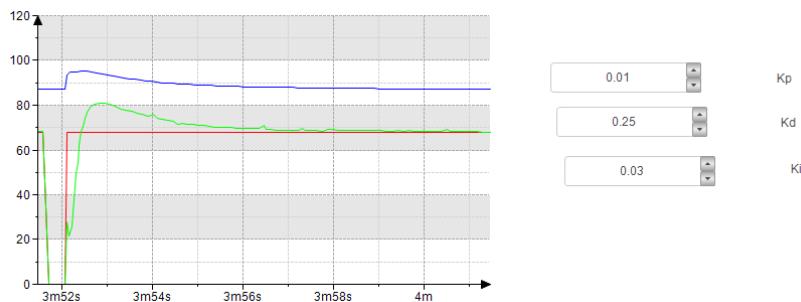


Figura 209. Ejemplo de sintonización para el controlador PID, verde: Entrada RPM, azul: Salida RPM y rojo: Valor de referencia. [Autores]

Las variables a las cuales el sistema tuvo mejor comportamiento son las que se ven en la Figura 210, ya que dichas constantes tanto para el control proporcional, integral y derivativo se adaptaban dentro de la respuesta necesaria para el proceso de producción en donde se necesita una regulación de velocidad precisa y rápida.

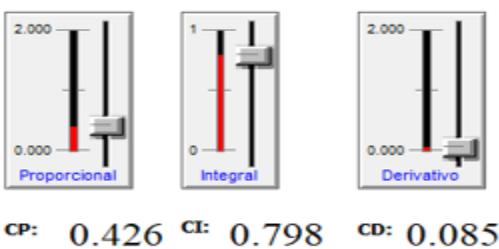


Figura 210. Constante proporcional, integral y derivativa definida para el controlador PID. [Autores]

2. Creación del controlador discreto PID en Codesys

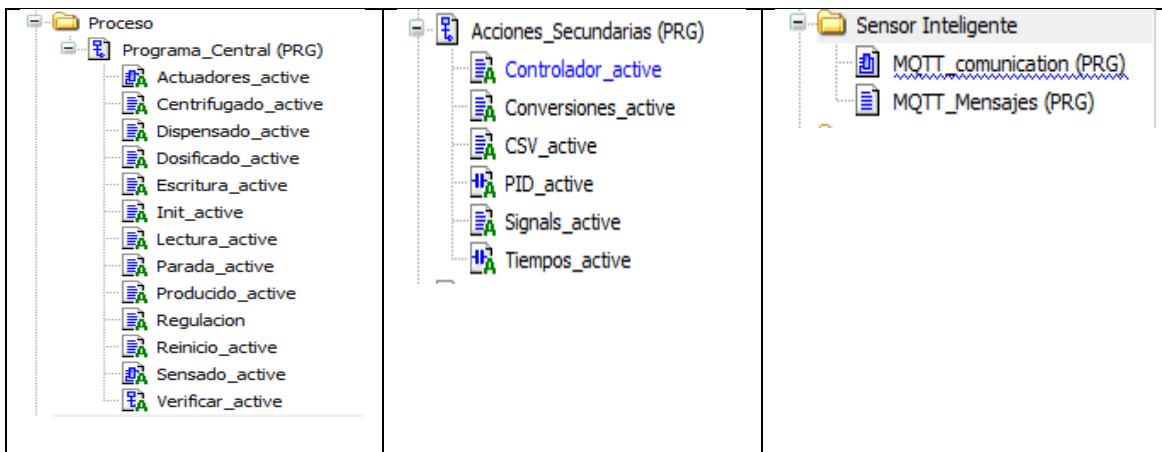
En la PC006, se desarrolló el paso a paso para realizar el controlador discreto en Matlab para luego desarrollar el código en Codesys. Solo para recordar, la ecuación de diferencias se observa en la ecuación **¡Error! No se encuentra el origen de la referencia.** (12).

3. Programación proceso en Codesys

Para la elaboración del proceso en Codesys, es necesario que el estudiante entienda la composición básica del proyecto. Observe la Tabla 10, en la cual se depositaron las estructuras creadas para el programa en Codesys, se aprecia una carpeta del **proceso** la cual tiene ciertas acciones asociadas a las etapas del proceso como el centrifugado, dispensado, dosificado y producido; pero también posee otras rutinas como la verificación de los sensores y actuadores. Tenemos otro programa llamado **Acciones Secundarias** donde se evalúan algunos procesos complementarios como la generación de señales de la PC006 las conversiones nombradas en la Figura 208, el bloque pid de la Figura 206, entre otros.

Tabla 10. Parte de la estructura del programa. [Autores]

--	--	--



El sensor inteligente que usa el ESP8266 posee solo un programa en CFC para la comunicación MQTT con la librería JanzTec, en la figura Figura 211 se aprecia el proceso general en GRAFCET el cual se encarga de realizar las distintas etapas definidas. Cabe destacar que en cada bloque del proceso, la rutina de **Regulación** (la que nos compete en esta práctica), se ejecuta en cada uno.

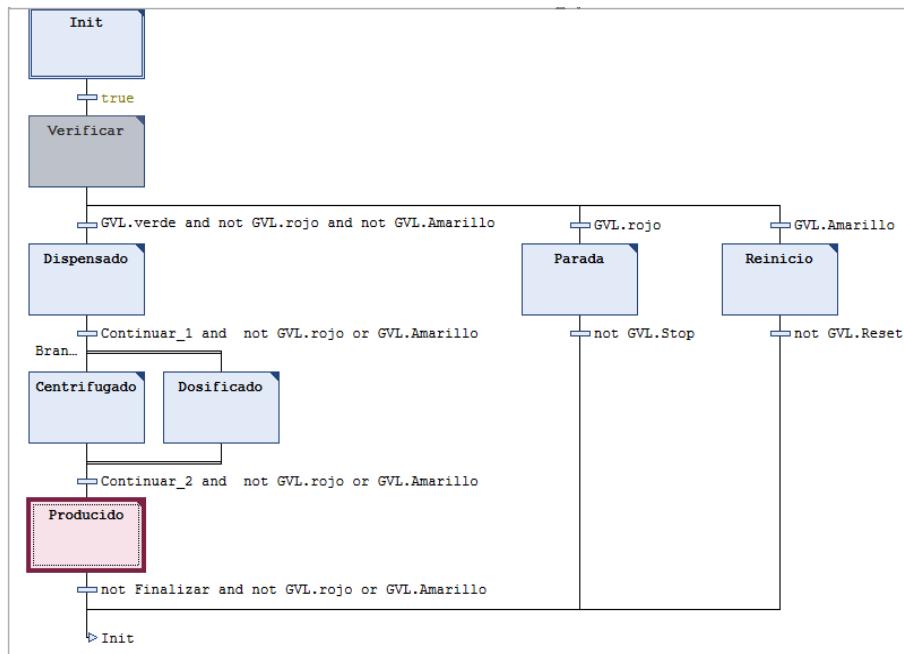


Figura 211. Grafket del proceso general. [Autores]

La rutina **Regulación** se observa en la figura a continuación y en los anexos se encuentra el código implementado en lenguaje estructurado. En la Figura 212 se observa la regulación por lotes y en la Figura 213 por distancia, esta última regulación necesitará el uso del sensor inteligente.

```

1 //Regulacion por numero de botellas utilizando el bloque PID
2 IF GVL.Mod_Vel_Dis = FALSE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON = FALSE THEN
3   IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE THEN
4     GVL.Set_point_RPM:=((GVL.Total_Botellas*3.75)+45);
5
6     GVL.PID_ON:=TRUE;
7   ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE THEN
8     GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
9     GVL.PID_ON:=TRUE;
10 ELSE
11   GVL.PID_ON:=FALSE;
12   GVL.AO_01:=0;
13 END_IF
14 END_IF
15
16 //Regulacion por numero de botellas utilizando el controlador PID
17 IF GVL.Mod_Vel_Dis = FALSE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON = TRUE THEN
18   GVL.PID_ON:=FALSE;
19   IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE THEN
20     GVL.Set_point_RPM:=((GVL.Total_Botellas*3.75)+45);
21   ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE THEN
22     GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
23   ELSE
24     GVL.AO_01:=0;
25   END_IF
26 END_IF
27

```

Figura 212. Código Regulación por lotes para PID. [Autores]

```

28 D_Aux:=REAL_TO_INT(GVL.Distancia);
29 //Regulacion por distancia con bloque PID
30 IF GVL.Mod_Vel_Dis = TRUE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON = FALSE THEN
31   IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE THEN
32     GVL.PID_ON:=TRUE;
33     CASE D_Aux OF
34       15..23:GVL.Set_point_RPM:=60;
35       24..33:GVL.Set_point_RPM:=68;
36       34..43:GVL.Set_point_RPM:=76;
37       44..53:GVL.Set_point_RPM:=84;
38       54..63:GVL.Set_point_RPM:=92;
39       64..73:GVL.Set_point_RPM:=100;
40       74..83:GVL.Set_point_RPM:=108;
41       84..100:GVL.Set_point_RPM:=115;
42   ELSE
43     GVL.Set_point_RPM:=0;
44     GVL.PID_ON:=FALSE;
45     GVL.AO_01:=0;
46   END_CASE;
47   ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE THEN
48     GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
49     GVL.PID_ON:=TRUE;
50   ELSE
51     GVL.PID_ON:=FALSE;
52     GVL.AO_01:=0;
53   END_IF;
54 END_IF;
55
56 //Regulacion por distancia con controlador PID
57 IF GVL.Mod_Vel_Dis = TRUE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON = TRUE THEN
58   GVL.PID_ON:=FALSE;
59   IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE THEN
60
61     CASE D_Aux OF
62       15..23:GVL.Set_point_RPM:=60;
63       24..33:GVL.Set_point_RPM:=65;
64       34..43:GVL.Set_point_RPM:=68;
65       44..53:GVL.Set_point_RPM:=74;
66       54..63:GVL.Set_point_RPM:=78;
67

```

Figura 213. Código Regulación por distancia para PID

Recuerde que el pseudocódigo del controlador discreto se realizó en la práctica requisito de esta y en la figura siguiente se observa el paso **Controlador** el cual tiene el pseudocódigo de la Figura 201 y para el caso del bloque PID, nosotros llamamos el bloque en el programa de **Acciones secundarias** también observe que el paso **PID** de la Figura 214 es el bloque de la Figura 206. Para la práctica puede omitir los Pasos que no necesite usar como csv.

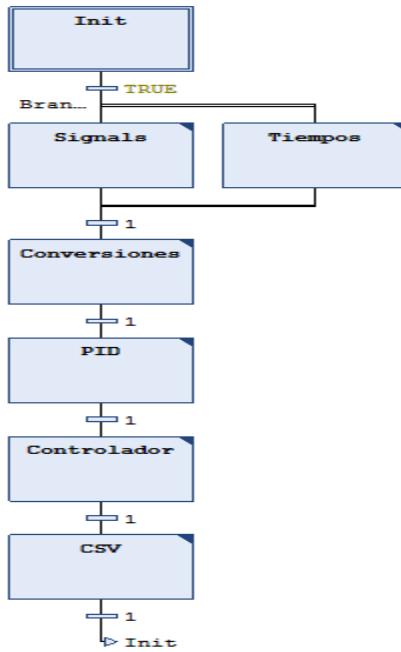


Figura 214. Grafcet acciones secundarias. [Autores]

Para realizar la regulación de distancia es necesario que cree un programa **MQTT_Communication** como en la Tabla 10 y configure la comunicación MQTT del sensor inteligente, diríjase al **repositorio** <https://github.com/cdpinzonm/Maquina-Conteo-capsulas> en el cual se encuentra el programa del sensor inteligente para que lo cargue en el módulo ESP8266 con un compilador de Arduino (IDE Arduino).

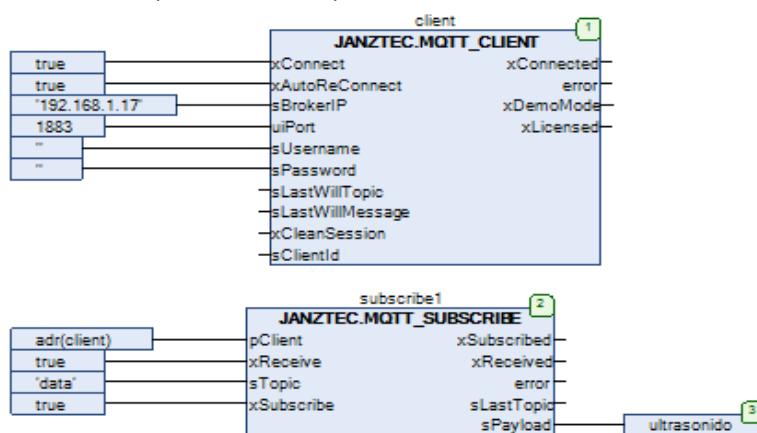


Figura 215. Sensor Inteligente comunicación MQTT. [Autores]

4. Comparación intuitiva entre ambos PIDs

En las prácticas anteriores ya se ha descrito como usar Intouch para crear interfaces, por tal razón pasamos a mostrar las HMI que nos permiten comparar visualmente las respuestas de los controladores. La interfaz de la configuración PID se observa a continuación:

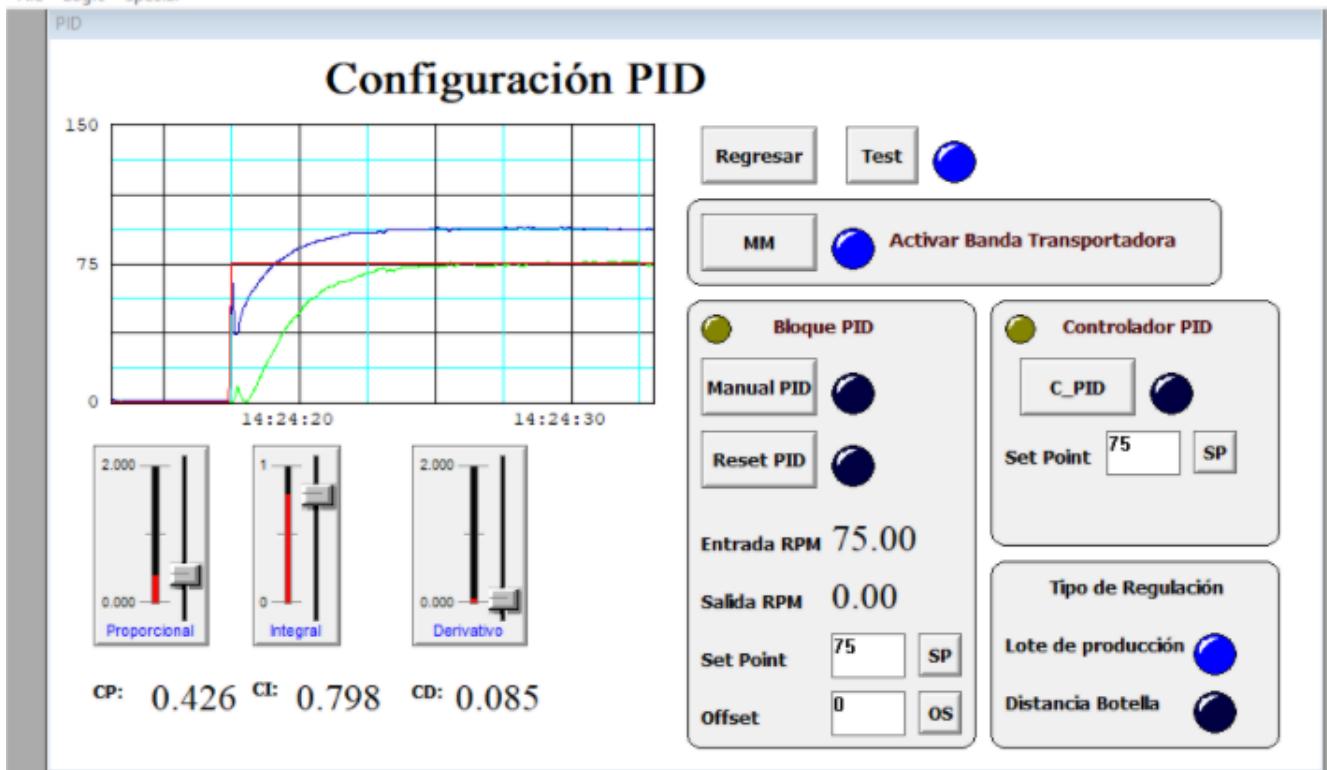


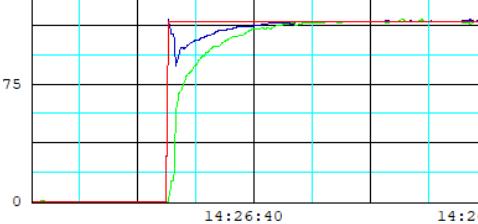
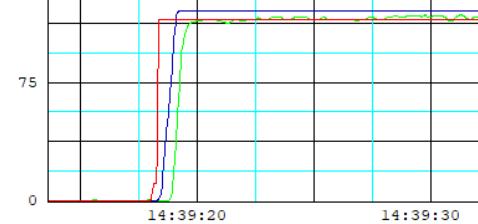
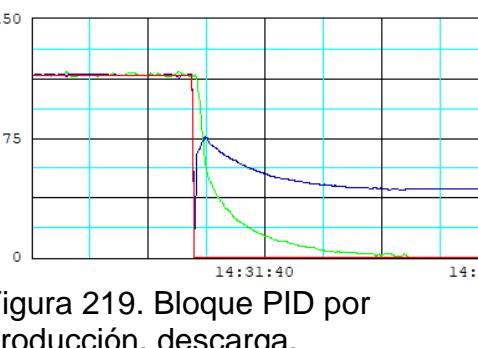
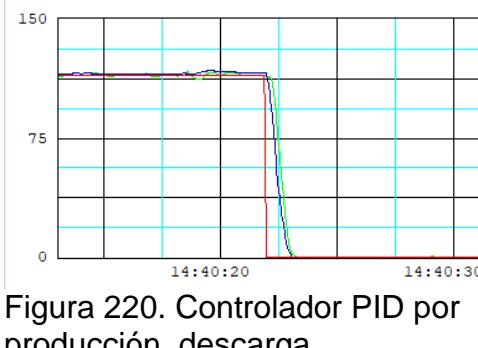
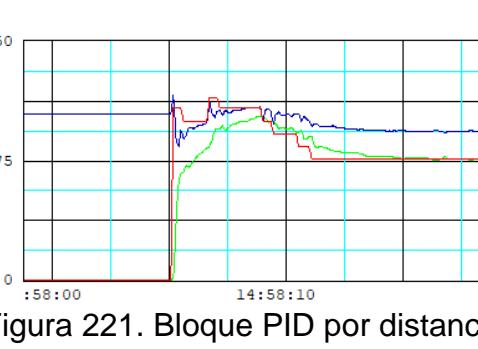
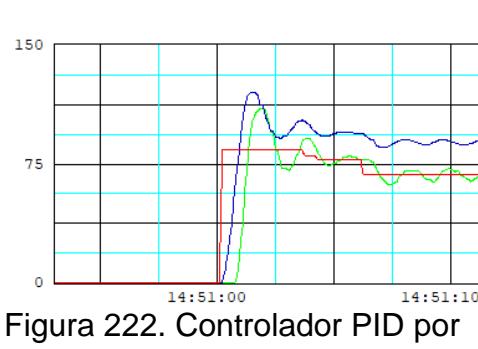
Figura 216. HMI Configuración PID en Intouch. [Autores]

En la Figura 216 Observe la creación de sliders de las constantes KP, KI, KD del bloque PID, se crearon varios botones para seleccionar el tipo de controlador, el tipo de regulación y unas cajas de texto para digitar el Setpoint de forma manual(una especie de testeo) ya que en la rutina **Regulación**, las revoluciones por minuto varias según las condiciones expresadas en la Figura 212 y Figura 213. Al observar la figura anterior se aprecia la referencia en rojo (setpoint), la señal en color verde es la entrada y la señal azul es la salida del controlador. El offset que se observa se debe a lo explicado en la sección 1.1 de esta práctica, ya que la salida será mucho mayor en resolución que la misma entrada por tal motivo se implementa una ecuación que relación los bits de entrada y salida, ver Figura 223

Se recopilaron las diferentes graficas obtenidas para cada PID y cada metodo de regulacion, para procede a compararlas intuitivamente, para ello responda las preguntas planteadas al final de esta sección.

Tabla 11. Comparación entre Bloque PID y Controlador PID

TIPO REGULACIÓN	TIPO DE PID	
	Bloque PID	Controlador PID

Por producción	Analisis de la Carga. SP 115 RPM	Analisis de la Carga. SP 115 RPM
		
	Figura 217. Bloque PID por producción, carga.	Figura 218. Controlador PID por producción, carga.
	Analisis de la Descarga. SP 0 RPM	Analisis de la Descarga. SP 0 RPM
Por distancia		
	Figura 219. Bloque PID por producción, descarga.	Figura 220. Controlador PID por producción, descarga.
Por distancia		
	Figura 221. Bloque PID por distancia.	Figura 222. Controlador PID por distancia.

The screenshot shows a ladder logic program with two main sections: GVL.PID_ON and GVL.PID.

GVL.PID_ON (Rows 1-16):

- IF GVL.PID_ON = TRUE AND GVL.C_PID_ON_FALSE = FALSE THEN
- GVL.LED_C_PID_ON=False;
- GVL.LED_PID_ON=True;
- GVL.Bist_AI_155E03 := (GVL.AI_01[0] + 1546.3);
- GVL.RPM_0.0187 := ((GVL.Bist_AI_155E03 * 0.049) - 75.75); //Entrada ADC
- GVL.AO_01[1393] := REAL_TO_UINT(GVL.Out_RPM / 115); //Salida PID

GVL.PID (Rows 17-31):

	PID			
EN	PID	ENO		
ACTUAL		Y	GVL.Out_RPM	41.3
SET_POINT		Y0	GVL.Limit_Vel	FALSE
KP		Y1	GVL.Overflow_FID	FALSE
TN		Y2		
KD		Y3		
TV		Y4		
Y_MANUAL		Y5		
Y_OFFSET		Y6		
Y_MIN		Y7		
Y_MAX		Y8		
MANUAL		Y9		
RESET		Y10		

Inputs (left side):

- GVL.PID_ON (coil)
- GVL.RPM (coil, value 0.0187)
- GVL.Set_point_RPM (coil, value 0)
- GVL.Kp (coil, value 0.426)
- GVL.Tn (coil, value 0.798)
- GVL.Kd (coil, value 0.058)
- GVL.Y_Manual_PID (coil, value 0)
- GVL.Offset_PID (coil, value 0)
- GVL.Manual (coil, value 116)
- GVL.Reset_PID (coil, value FALSE)

Outputs (right side):

- GVL.Out_RPM (coil, value 41.3)
- GVL.Limit_Vel (coil, value FALSE)
- GVL.Overflow_FID (coil, value FALSE)

Figura 223. Conversiones Entrada/Salida ADC. [Autores]

PREGUNTAS: COMPARACIÓN INTUITIVA DE LAS GRAFICAS EN LA Tabla 11.

- ¿Qué puede notar en la señal de salida (azul) en la Figura 217 y Figura 218?.

- ¿De cuánto es aproximadamente el tiempo de subida y de asentamiento en el modo de regulación por producción de lotes? Según lo realizado en las prácticas de PID ¿Por qué esos tiempos son mejores en el controlador discreto?

- Para el caso de la descarga en la regulación por producción de lotes, ¿un tiempo de reacción más rápido o más lento como se ve reflejado físicamente en la máquina?

- Si la banda transportadora va a una velocidad de 115 RPM y necesitamos frenarla sin que la inercia del motor afecte la posición de la botella en la banda (haga de cuenta que la botella no queda ubicada debajo de la boquilla donde caen las pastas a la botella), ¿qué tipo de controlador implementaría?

- En la Figura 221 y Figura 222 ¿Cómo se comportan los dos PID con respecto a las perturbaciones generadas constantemente por el sensor de distancia (ultrasonido)?

- En la regulación por distancia que prefiere: ¿Un controlador que responda más lento a los cambios o un controlador con mejor respuesta, pero con algunas oscilaciones como en el caso del controlador de la Figura 222?.

Anexos

1. Recopilaciones variables Intouch/Codesys[PC003]

SCADA/Intouch	Type	Codesys	HMI					
			Windows Script	Maquina HMI	Pane I_Control	Graficas	Conteo	PLD
Aux_Bot1	Memory Integer	----						
Aux_Distancia_Bot	Memory	----						

	Intege r							
Aux_Limit_Bo t	Memo ry Discre te	----				X		
Aviso_Botella s	Memo ry Mesag e	----	WS.Alarmas					
Aviso_Capsul as	Memo ry Mesag e	----	WS.Alarmas					
Avisos_Proce so	Memo ry Mesag e	----	Alarmas					
Bot_Text	Memo ry Mesag e	----	WS.Alarmas / WS.Condeo/ WS. PanelControl		X			
Botellas	I/O Intege r	Botellas	WS.Condeo/		X	X	X	
BotonSQL	Memo ry Discre te	----	Conexion					
Cap_Text	Memo ry Messa ge	----	WS.Alarmas / WS.Condeo/ WS.PanelCo ntrol		x			
Capsulas	I/O Intege r	Limit_Pill	WS.Condeo					
ErrorIns	Memo ry Mesag e	----	Conexion					
Estado	I/O Messa ge	Etapa		X			X	
Estado2	Memo ry	----	Conexion					

	Messa ge							
Inicio	I/O Discrete	Start		X	x			
Kd	I/O Real	Kd						X
Ki	I/O Real	Ki						X
Kp	I/O Real	Kp						X
Led_M1	I/O Discrete	M1_out			x			
Led_M2	I/O Discrete	M2_out			x			
Led_M3	I/O Discrete	M3_out		X	x			X
Led_M4	I/O Discrete	M4_out			X			
Led_M5	I/O Discrete	M5_out			X			
Led_Mod_Vel	I/O Discrete	Mod_Vel_Dis			X			
Led_RC	I/O Discrete	Led_Pill_End						
Led_Ultrasonido	I/O Discrete	Led_SU			X			
Limit_B_Max	Memo ry Discrete	----						
Limit_B_Min	Memo ry Discrete	----						
Limit_Bot	I/O Integre	Limit_Bottle	WS.Conleo/ WS. PanelControl				X	

Limit_Caps	I/O Integer	Limit_Pill	WS.Conleo/ WS. PanelControl				X	
Lotes	I/O Integer	Lotes		X	X		X	
M_banda	I/O Discrete	M_Banda			X			X
M_SP	Memory Discrete	-----						
Max_Pill_Con t	Memory Integer	----	/WS.Grafica s			X		
Min_Pill_Cont	Memory Integer	----	/WS.Grafica s			X		
Mod_Vel_Dis	I/O Discrete	Mod_Vel_Dis			X			
Motor_Actuad or	I/O Discrete	M_actuador			X			
Motor_Boquilla	I/O Discrete	M_boquilla			X			
Motor_Centrif uga	I/O Discrete	M_centrifuga			X			
Motor_Servo	I/O Discrete	M_Servo			X			
NB	Memory Discrete	-----	WS. PanelControl		X			
NC	Memory Discrete	-----	WS.Conleo// WS. PanelControl		X			
Num_Botellas	I/O Integer	Num_Bottle				X	X	

Num_Pastas	I/O Integer	Num_Pill		X		X	X	
Offset_Text	Memory Message							X
Out_RPM	I/O Real	Out_RPM						X
Reset	I/O Discrete	Reset		X	X			
Reset_Botellas	I/O Discrete	Reset_Cont_Bottle						
Reset_Capsulas	I/O Discrete	Reset_Cont_Pill						
Reset_PID	I/O Discrete	Reset_PID						X
ResultCode	Memory Integer	----	Conexion					
Salida_Amarilla	I/O Discrete	Amarillo		X	X			
Salida_Rojo	I/O Discrete	Rojo		X	X			
Salida_Verde	I/O Discrete	Verde		X	X			
SB1	I/O Discrete	Led_SB			X			
SC1	I/O Discrete	Led_SC			X			
SD1	I/O Discrete	Led_SD			X			
Set_Point	I/O Real	Set_Point_RPM						X
Slider_Servo	I/O Integer	Servo_M	WS.Conleo/ WS. PanelControl		X			

Slider_SP_fijo	I/O Integer	M_Banda_M	WS.Condeo/ WS. PanelControl		X			
SM3_Aux	Memory Integer	----	WS.Condeo/ WS. PanelControl		X			
SP_Text	Memory Message	-----						X
SS_Aux	Memory Real	-----	WS.Condeo/ /WS. PanelControl		X			
Stop	I/O Discrete	Stop		X	X			
Test_Maquina	I/O Discrete	Mod_Manual			X			
Total_Bot	I/O Integer	Total_Botellas		X			X	
Ultrasonido	I/O Real	Distancia			X			
Velocidad	I/O Integer	RPM				X		X
Velocidad_Out	I/O Real	Out_RPM						X
Y_Manual_PID	I/O Discrete	Y_Manual_PID						X

2. Código creación Tabla por SQL[PC004]

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DatasetProceso](
    [Datold] [int] IDENTITY(1,1) NOT NULL,
    [Fecha] [nchar](50) NULL,
    [Hora] [nchar](50) NULL,
    [Start] [int] NULL,
    [Stop] [int] NULL,
    [NoBotellas] [int] NULL,
    [LimBotellas] [int] NULL,
    [NoPildoras] [int] NULL,
    [LimPildoras] [int] NULL,
    [Lotes] [int] NULL,
    [Etapa] [nchar](50) NULL,
    [SensorDispensador] [int] NULL,
    [SensorBanda] [int] NULL,
    [SensorCentrifuga] [int] NULL,
    [SensorUltrasonido] [int] NULL,
    [MotorActuador] [int] NULL,
    [MotorBanda] [int] NULL,
    [MotorCentrifuga] [int] NULL,
    [MotorBoquilla] [int] NULL,
    [ServoM_Porcen] [real] NULL,
    [RPMBanda] [int] NULL,
CONSTRAINT [PK_DatasetProceso] PRIMARY KEY CLUSTERED
(
    [Datold] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

3. Código Data Change Script en Intouch[PC004]

```
{Código enviar SQL on premise}
IF BotonSQL ==1 THEN
    ResultCode = SQLConnect( ConnectionId,"DSN=SQLMaquina; User      ID
=<suUsuario>;Password=<su contraseña>; ");
    IF ResultCode == 0 THEN
        Estado2 = "Tu Conexion SQL esta trabajando bien";
    ELSE
        Estado2 = "Conexion SQL mala, verSMC log.";
    ENDIF;
    RC = SQLInsert( ConnectionId, "DatasetProceso", "BindSqlMaquina" );
    ErrorIns= SQLErrorMsg(RC);
    ResultCode = SQLDisconnect(ConnectionId);
ENDIF;
```

4. Código señales de identificación en Codesys[PC006]

```
IF GVL.B_Paso AND Salir = FALSE THEN
    Type_Signal:=1;
    GVL.Guardar:=TRUE;
    //Forma de la señal
    IF Finalizado = TRUE THEN
        Espera:=0;
        GVL.Signal_P:=1;

    ELSE
        Espera:=1;
    END_IF
    //Número de datos para muestrear
    IF GVL.Salida_CSV THEN
        Numero_Datos_Aux:=Numero_Datos_Aux+1;
    ELSE
        Numero_Datos_Aux:=Numero_Datos_Aux;
    END_IF
    IF Numero_Datos_Aux = 1000 THEN
        Salir:=TRUE;
        GVL.B_Paso:=FALSE;
    END_IF
END_IF

IF GVL.B_Paso_Cuadrada AND Salir = FALSE THEN
    Type_Signal:=2;
    GVL.Guardar:=TRUE;
    //Forma de la señal
    IF Finalizado = TRUE THEN
        Espera:=0;
        GVL.Signal_PC:=1;
        IF Finalizado_2 THEN
            Espera_2:=0;
            GVL.Signal_PC:=1.2;
            Pulsos_Cuadrado:=TRUE;
            IF Tiempo_Cuadrada = TRUE THEN
                GVL.Signal_PC:=0.8;
            ELSIF Tiempo_Cuadrada = FALSE THEN
```

```

        GVL.Signal_PC:=1.2;
    END_IF
ELSE
    Espera_2:=1;
END_IF

ELSE
    Espera:=1;;
END_IF
//Número de datos para muestrear
IF GVL.Salida_CSV THEN
    Numero_Datos_Aux:=Numero_Datos_Aux+1;
ELSE
    Numero_Datos_Aux:=Numero_Datos_Aux;
END_IF
IF Numero_Datos_Aux = 1000 THEN
    Salir:=TRUE;
    GVL.B_Paso_Cuadrada:=FALSE;
END_IF
END_IF

IF GVL.B_Paso_Modificada AND Salir = FALSE THEN
    Type_Signal:=3;
    GVL.Guardar:=TRUE;
    IF Finalizado = TRUE THEN
        Espera:=0;
        GVL.Signal_CM:=1;
        IF Finalizado_2 THEN
            Espera_2:=0;
            GVL.Signal_CM:=1.2;
        ELSE
            Espera_2:=1;
        END_IF
    ELSE
        Espera:=1;
    END_IF
    //Número de datos para muestrear
    IF GVL.Salida_CSV THEN
        Numero_Datos_Aux:=Numero_Datos_Aux+1;
    ELSE
        Numero_Datos_Aux:=Numero_Datos_Aux;
    END_IF
    IF Numero_Datos_Aux = 1000 THEN
        Salir:=TRUE;
        GVL.B_Paso_Modificada:=FALSE;
    END_IF
END_IF

```

```

IF GVL.B_Pseudoaleatoria AND Salir = FALSE THEN
    Type_Signal:=4;
    GVL.Guardar:=TRUE;
    Periodo_PA:=REAL_TO_TIME(Periodo_Aleatorio+Periodo_Aleatorio_Entero);
    IF Finalizado = TRUE THEN
        Espera:=0;
        GVL.Signal_PA:=1;
        IF Finalizado_2 THEN
            Espera_2:=0;
            Pulsos_Aleatorioss:=TRUE;
            GVL.Signal_PA:=1+Salida_Random;
        ELSE
            Espera_2:=1;
        END_IF

    ELSE
        Espera:=1;
    END_IF
//Número de datos para muestrear
IF GVL.Salida_CSV THEN
    Numero_Datos_Aux:=Numero_Datos_Aux+1;
ELSE
    Numero_Datos_Aux:=Numero_Datos_Aux;
END_IF
IF Numero_Datos_Aux = 1000 THEN
    Salir:=TRUE;
    GVL.B_Pseudoaleatoria:=FALSE;
END_IF
END_IF

IF GVL.B_Paso = FALSE AND GVL.B_Paso_Cuadrada = FALSE AND
GVL.B_Paso_Modificada = FALSE AND GVL.B_Pseudoaleatoria = FALSE THEN
    GVL.Guardar:=FALSE;
    Pulsos_Cuadrado:=FALSE;
    Pulsos_Aleatorioss:=FALSE;
    Numero_Datos_Aux:=0;
    Salir:=FALSE;
    Espera:=FALSE;
    Espera_2:=FALSE;
    Tiempo_Cuadrada:=FALSE;
    Tiempo_Aleatorio:=FALSE;
    Finalizado:=FALSE;
    Finalizado_2:=FALSE;
    GVL.Signal_P:=0;
    GVL.Signal_PC:=0;
    GVL.Signal_CM:=0;
    GVL.Signal_PA:=0;

```

```

Type_Signal:=0;
Reset:=TRUE;
ELSE
    Reset:=FALSE;
END_IF

IF GVL.Out_Singal_M AND GVL.Verde = FALSE THEN
    CASE Type_Signal OF
        1: GVL.AO_01:=REAL_TO_UINT(GVL.Signal_P*GVL.Valor_Out_M);
        GVL.Set_Point_Signal:=REAL_TO_UINT(GVL.Valor_Out_M*GVL.Signal_P);

        2: GVL.AO_01:=REAL_TO_UINT(GVL.Signal_PC*GVL.Valor_Out_M);
        GVL.Set_Point_Signal:=REAL_TO_UINT(GVL.Valor_Out_M*GVL.Signal_P);

        3: GVL.AO_01:=REAL_TO_UINT(GVL.Signal_CM*GVL.Valor_Out_M);
        GVL.Set_Point_Signal:=REAL_TO_UINT(GVL.Valor_Out_M*GVL.Signal_P);

        4: GVL.AO_01:=REAL_TO_UINT(GVL.Signal_PA*GVL.Valor_Out_M);
        GVL.Set_Point_Signal:=REAL_TO_UINT(GVL.Valor_Out_M*GVL.Signal_P);
    ELSE
        GVL.AO_01:=0;
        GVL.Set_Point_Signal:=0;
    END_CASE
END_IF

```

5. Pseudocodigo Controlador PID[PC006]

```

//Controlador obtenido con la señal paso
M6:=0.0;//E[n]
Q0:=0.1103;
Q1:=0.08595;
Q2:=0.6326 ;

```

```

Q3 := 1.6326;
TS(IN:=Inicio_Ts , PT:=T#0.01S);
IF TS.Q THEN
    GVL.Error:= GVL.Set_point_RPM-GVL.RPM;
    M6:=GVL.Error;
    //Y[n]:=0.1103E[n-1] - 0.08595E[n-2] - 0.6326Y[n-2] + 1.6326Y[n-1]
    M1:= Q0*M4 - Q1*M5 - Q2*M3+ Q3*M2;
    //Memorias para los desplazamientos
    M5:=M6;
    M4:=M5;
    M3:=M2;
    M2:=M1;
    GVL.Out_RPM:= (M1);
    //Limites de funcionamiento
    IF GVL.Out_RPM <= 0 THEN
        GVL.Out_RPM:=0;
    END_IF
    IF GVL.Out_RPM >= 116 THEN
        GVL.Out_RPM:=116;
    END_IF
    //IF GVL.Out_RPM > 0 AND M1 < 116 THEN
    //    GVL.Out_RPM:=M1;
    //END_IF
    Inicio_Ts:=FALSE;
END_IF

IF M1 >= 2000 THEN
    M1:=0.0; //y[n]
    M2:=0.0; //Y[n-1]
    M3:=0.0; //Y[n-2]
    M4:=0.0; //E[n-1]
    M5:= 0.0;//E[n-2]
    M6:=0.0;//E[n]
    Q0:=0.1103;
    Q1:=0.08166 ;
    Q2:=0.6326 ;
    Q3 := 1.6326;
END_IF
IF GVL.M_banda = FALSE THEN
    M1:=0.0; //y[n]
    M2:=0.0; //Y[n-1]
    M3:=0.0; //Y[n-2]
    M4:=0.0; //E[n-1]
    M5:= 0.0;//E[n-2]
    M6:=0.0;//E[n]
    Q0:=0.1103;
    Q1:=0.08166 ;
    Q2:=0.6326 ;

```

```
Q3 := 1.6326;  
END_IF
```

6. Conversiones previas al desarrollo del PID [PC007]

```
IF GVL.PID_ON = TRUE AND GVL.C_PID_ON = FALSE THEN  
    GVL.LED_C_PID_ON:=FALSE;  
    GVL.LED_PID_ON:=TRUE;  
    GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;  
    GVL.RPM:=((GVL.Bist_AI*0.049)-75.75); //Entrada ADC  
    GVL.AO_01:=REAL_TO_UINT(GVL.Out_RPM*3800)/115; //Salida PID  
  
ELSIF GVL.C_PID_ON = TRUE AND GVL.PID_ON = FALSE THEN  
    GVL.LED_C_PID_ON:=TRUE;  
    GVL.LED_PID_ON:=FALSE;  
    GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
```

```

GVL.RPM:=((GVL.Bist_AI*0.049)-75.75); //Entrada ADC
GVL.AO_01:=REAL_TO_UINT(GVL.Out_RPM*3800)/115; //Salida PID
Inicio_TS:=TRUE;

ELSE
    GVL.LED_C_PID_ON:=FALSE;
    GVL.LED_PID_ON:=FALSE;
    IF GVL.Out_Singal_M = TRUE THEN
        //      GVL.AO_01:=0;
        GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
        GVL.Bits_In_Signal:=REAL_TO_UINT(GVL.Bist_AI);
    END_IF

    GVL.Bist_AI := (0.7779*GVL.AI_01) + 1546.3;
    GVL.RPM:=((GVL.Bist_AI*0.049)-75.75); //Entrada ADC

END_IF

IF GVL.AO_01 >=4000 THEN
    GVL.AO_01 :=4000;
END_IF

```

7. Código Regulación en Codesys [PC007]

```

//Regulacion por numero de botellas utilizando el bloque PID
IF GVL.Mod_Vel_Dis = FALSE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON =
FALSE THEN
    IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE
THEN
        GVL.Set_point_RPM:=((GVL.Total_Botellas*3.75)+45);

        GVL.PID_ON:=TRUE;
    ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE
THEN
        GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
        GVL.PID_ON:=TRUE;
    ELSE
        GVL.PID_ON:=FALSE;
        GVL.AO_01:=0;
    END_IF

```

```

END_IF
//Regulacion por numero de botellas utilizando el controlador PID
IF GVL.Mod_Vel_Dis = FALSE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON =
TRUE THEN
    GVL.PID_ON:=FALSE;
    IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE
THEN
        GVL.Set_point_RPM:=((GVL.Total_Botellas*3.75)+45);
    ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE
THEN
        GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
    ELSE
        GVL.AO_01:=0;
    END_IF
END_IF
D_Aux:=REAL_TO_INT(GVL.Distancia);
//Regulacion por distancia con bloque PID
IF GVL.Mod_Vel_Dis = TRUE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON =
FALSE THEN
    IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE
THEN
        GVL.PID_ON:=TRUE;
        CASE D_Aux OF
            15..23:GVL.Set_point_RPM:=60;
            24..33:GVL.Set_point_RPM:=68;
            34..43:GVL.Set_point_RPM:=76;
            44..53:GVL.Set_point_RPM:=84;
            54..63:GVL.Set_point_RPM:=92;
            64..73:GVL.Set_point_RPM:=100;
            74..83:GVL.Set_point_RPM:=108;
            84..100:GVL.Set_point_RPM:=115;
        ELSE
            GVL.Set_point_RPM:=0;
            GVL.PID_ON:=FALSE;
            GVL.AO_01:=0;
        END_CASE
    ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE
THEN
        GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
        GVL.PID_ON:=TRUE;
    ELSE
        GVL.PID_ON:=FALSE;
        GVL.AO_01:=0;
    END_IF
END_IF
//Regulacion por distancia con controlador PID
IF GVL.Mod_Vel_Dis = TRUE AND GVL.Out_Singal_M = FALSE AND GVL.C_PID_ON =
TRUE THEN

```

```

GVL.PID_ON:=FALSE;
IF GVL.M_banda AND GVL.SP_Fijo = FALSE AND GVL.Mod_Manual = FALSE
THEN

    CASE D_Aux OF
        15..23:GVL.Set_point_RPM:=60;
        24..33:GVL.Set_point_RPM:=65;
        34..43:GVL.Set_point_RPM:=68;
        44..53:GVL.Set_point_RPM:=74;
        54..63:GVL.Set_point_RPM:=78;
        64..73:GVL.Set_point_RPM:=80;
        74..83:GVL.Set_point_RPM:=82;
        84..100:GVL.Set_point_RPM:=84;
    ELSE
        GVL.Set_point_RPM:=0;
        GVL.AO_01:=0;
    END_CASE
ELSIF GVL.M_banda AND GVL.SP_Fijo= TRUE AND GVL.Mod_Manual = TRUE
THEN
    GVL.Set_point_RPM:=UINT_TO_REAL(GVL.M_banda_M);
ELSE
    GVL.AO_01:=0;
END_IF

END_IF
IF GVL.Distancia > 100 THEN
    GVL.Distancia:=0;
END_IF

```