Instituto Tecnológico de Buenos Aires

Maestría en Ingeniería de las Telecomunicaciones

TRABAJO DE TESIS

Título

El actual paradigma de los sistemas HMI/SCADA sobre las redes de telecomunicaciones en Argentina y sus futuros desafíos

Autor

Ing. Francisco Gulli Blanco

Director de Tesis

Ing. Enrique Minio

Buenos Aires, Abril de 2015

Presentación

Información del autor

Mi nombre es Francisco Gulli Blanco, tengo 34 años de edad y vivo en la ciudad de Buenos Aires, Argentina. Soy ingeniero electrónico graduado en el año 2008. Mi experiencia principal se centra en el área de la automatización, el control y las comunicaciones. He trabajado fundamentalmente en el rubro de la generación, distribución y transporte de energía eléctrica.

Epígrafe y agradecimientos

Este trabajo pretende dar testimonio del recorrido de un camino que inicié hace algunos años en donde a partir de mis experiencias profesionales en el ámbito de la automatización y las comunicaciones me planteé el desarrollo de un sistema de control que viniera a llenar un espacio vacío.

El tiempo pasó y el camino aún no terminó, sin embargo mirando hacia atrás puedo decir orgulloso que he logrado buenos resultados. Los desafíos a futuro aún son muchos, así como el trabajo necesario para superarlos. Sin embargo hoy más que nunca renuevo mi compromiso para no detenerme.

Quisiera aprovechar para agradecer profundamente a todos los que me han supervisado mi labor durante la elaboración de esta Tesis y el desarrollo del sistema, a todos los que han colaborado conmigo en incontables oportunidades y a mis maestros formales e informales de todos los días, los cuales acompañaron mi desarrollo tanto profesional como personal.

Finalmente quisiera agradecer a mis padres, amigos y a Sasha, mi compañera de vida.

Índice

Presentación....2
Información del autor....2
Epígrafe y agradecimientos....2
1. Capítulo I: Introducción....9
1.1. Presentación del tema en estudio....9
1.2. Propósito del presente trabajo....11
1.2.1. Panorama local....11
1.2.2. Hardware potente y de bajo costo....11
1.2.3. Nuevas herramientas de programación de código abierto....12
1.2.4. Redes de comunicaciones....12
1.2.5. Ataques informáticos y ciberseguridad....13
1.3. Metodología utilizada....13
1.4. Estructura del trabajo....14
1.4.1. Breve descripción....14
1.4.2. Descripción por capítulos....14
1.5. Limitaciones....15
2. Capítulo II: Contexto....17
2.1. Descripción de la tecnología....17
2.1.1. ¿Qué es SCADA?....17
2.1.2. Definición de SCADA....17
2.1.3. Procesos aplicables....18
2.2. Historia....19
2.2.1. Desarrollo desde la telemetría....19
2.2.2. Dependencia de las comunicaciones y las computadoras....21
2.3. Sistemas de Telecontrol....24
2.3.1. Esquema funcional de un Sistema de Telecontrol....26
2.3.2. Funciones de los Sistemas de Telecontrol....26
2.3.3. Unidad Terminal Remota....28
2.4. Sistemas de Transmisión de Datos....39
2.4.1. Rol de los Sistemas de Transmisión de Datos....39
2.4.2. Tipos de configuraciones....39
2.4.3. Modos de Transmisión....41

2.4.4. Tecnologías de enlaces punto a punto 42
2.4.5. Tecnologías de redes de área amplia (WAN) 44
2.4.6. MPLS, Internet y redes móviles 47
2.5. Generaciones de sistemas SCADA 49
2.5.1. Sistemas SCADA de primera generación 49
2.5.2. Sistemas SCADA de segunda generación 53
2.5.3. Sistemas SCADA de tercera generación 58
3. Capítulo III: Objetivo del trabajo 63
3.1. Enunciado 63
3.2. Desarrollo 63
4. Capítulo IV: Punto de partida 65
4.1. Identificación del problema 65
4.2. Evolución 65
4.3. Nuevos desafíos 66
5. Capítulo V: Esbozo de la Solución 69
5.1. Especificación de características requeridas 69
5.2. Diseño de arquitectura básica 70
Base de datos SQL 70
Módulo SCADA o Núcleo 70
Módulo de Configuración y Desarrollo 71
Módulo HMI 71
Sistema gráfico 71
5.3. Herramientas de programación a aplicar 72
5.4. Descripción de las herramientas 74
5.4.1. PostgreSQL 74
5.4.2. Qt 76
5.4.3. Javascript 80
5.4.4. jQuery 82
5.4.5. PHP 82
5.4.6. Especificación SVG 84
6. Capítulo VI: Conceptos del sistema 86
6.1. Elementos de adquisición de datos 86
6.1.1. Estación 86
6.1.2. Dispositivo 87

6.1.3. Canal de comunicación 87
6.1.4. Grupos de Escaneo 87
6.1.5. Relación entre elementos 88
6.2. Mapa de datos del Dispositivo 90
6.2.1. Registro de Entrada 16 Bits / Registro de Salida 16 Bits 90
6.2.2. Registro de Entrada 32 Bits / Registro de Salida 32 Bits 91
6.2.3. Entrada Digital/ Salida Digital 91
6.3. Sets de Límites 92
6.4. Sets de Texto 92
6.5. Sets de Conversión 93
6.6. Grupos de Alarma 93
6.7. Sets de Comandos 94
6.8. Puntos 94
7. Capítulo VII: Arquitectura del sistema 96
7.1. Diagrama funcional 96
7.2. Descripción de módulos 98
7.2.1. Módulo SCADA o Núcleo 98
7.2.2. Módulo de Configuración y Desarrollo 101
7.2.3. Módulo HMI 108
8. Capítulo VIII: Símbolos y pantallas SCADA 119
8.1. Características comunes 120
8.2. Descripción de tipos de símbolo 120
8.2.1. Tipo Bar 120
8.2.2. Tipo Gauge 123
8.2.3. Tipo Static 126
8.2.4. Tipo Status 127
8.2.5. Tipo Measure 129
8.3. Edición pantallas y de símbolos 131
8.3.1. Inkscape 131
8.4. Biblioteca de símbolos 133
8.4.1. Tipo Bar 134
8.4.2. Tipo Gauge 135
8.4.3. Tipo Measure 135
8.4.4. Tipo Static 136

8.4.5. Tipo Status 136
8.5. Edición de pantallas 137
9. Capítulo IX: Consideraciones de seguridad 138
9.1. El caso Stuxnet 138
9.2. Mitigación de riesgos 139
9.2.1. Segmentación de redes y uso de Firewalls 139
9.2.2. Mejoras 142
9.2.3. Consideraciones respecto al sistema SCADA expuesto 143
10. Capítulo X: Conceptos finales 145
10.1. Conclusiones 145
10.2. Futuros pasos 146
Supervisión por eventos 147
Gestión y mantenimiento a gran escala 147
Seguridad confianza y privacidad 147
Procesamiento de información en tiempo real 147
Simulación 148
Mejor interoperabilidad 148
11. Anexo: Modelo de datos del sistema 149
11.1. Diagrama de tablas y relaciones de la Base de Datos 150
11.2. Tabla de Puntos 151
11.2.1. Campos generales 151
11.2.2. Campos relacionados con la indicación del punto 152
11.2.3. Campos relacionados con la adquisición de datos 153
11.2.4. Campos relacionados con la ejecución de acciones 153
11.2.5. Campos relacionados con la naturaleza del punto 154
11.2.6. Otros campos 156
11.3. Tablas relacionadas con pantallas del sistema 157
11.3.1. Tabla tbl_display 157
11.3.2. Tabla tbl_display_point_assignment 157
11.3.3. Tabla tbl_display_point_type_name 158
11.4. Tablas relacionadas con la adquisición de datos 158
11.4.1. Tabla tbl_station 158
11.4.2. Tabla scd_device 159
11.4.3. Tabla scd_device_set 159

11.4.4. Tabla scd_device_type 160
11.4.5. Tabla scd_device_protocol 160
11.4.6. Tabla scd_comm_channel 160
11.4.7. Tabla scd_comm_channel_set 161
11.4.8. Tabla scd_comm_channel_type 161
11.4.9. Tabla scd_scan_group 162
11.4.10. Tabla scd_scan_group_set 163
11.4.11. Tabla scd_scan_group_data_type 163
11.5. Tablas de Sets 164
11.5.1. Tabla tbl_alarm_group 164
11.5.2. Tabla tbl_alarm_group_users 164
11.5.3. Tabla tbl_analog_limits_set 164
11.5.4. Tabla tbl_analog_conversion_set 165
11.5.5. Tabla tbl_text_set 166
11.5.6. Tabla tbl_text_set_name 166
11.6. Tablas de relacionadas a telemandos 166
11.6.1. Tabla scd_command_set 166
11.6.2. Tabla scd_command_set_name 167
11.6.3. Tabla scd_command_seq 168
11.6.4. Tabla scd_command_seq_name 168
11.7. Tablas relacionadas a vistas 169
11.7.1. Tabla adm_charts 169
11.7.2. Tabla adm_charts_points 169
11.7.3. Tabla adm_historian_view 169
11.7.4. Tabla adm_historian_view_points 170
11.8. Tablas de usuarios 170
11.8.1. Tabla adm_users 170
11.8.2. Tabla adm_user_displays 171
11.9. Tablas de listados del sistema 171
11.9.1. Tabla tbl_alarm_list y tbl_event_list 171
11.9.2. Tabla tbl_historian_list 172
11.10. Otras tablas 173
11.10.1. Tabla adm_crontab 173
12. Bibliografía 174

12.1. Libros 174
12.2. Artículos 174
12.3. Documentación en línea 175
12.4. Normas 176
13. Glosario de términos 177

1. Capítulo I: Introducción

1.1. Presentación del tema en estudio

Es notorio como en los últimos años Sudamérica en general y nuestro país en particular ha experimentado un crecimiento industrial sostenido, datos estadísticos de diversas fuentes respaldan esta afirmación.

De acuerdo al Instituto Nacional De Estadísticas y Censos (INDEC) entre el año 2005 y hasta la actualidad Argentina mantuvo tasas de crecimiento interanual del PIB (Producto Interno Bruto) que en promedio fueron del 5%.

Se ha experimentado el período de crecimiento más extenso de las últimas décadas, y si bien la crisis internacional ha desencadenado una contracción económica, diversos indicadores estiman que este proceso se ha revertido en los últimos meses esperándose un nuevo período de importante crecimiento a partir de 2016.

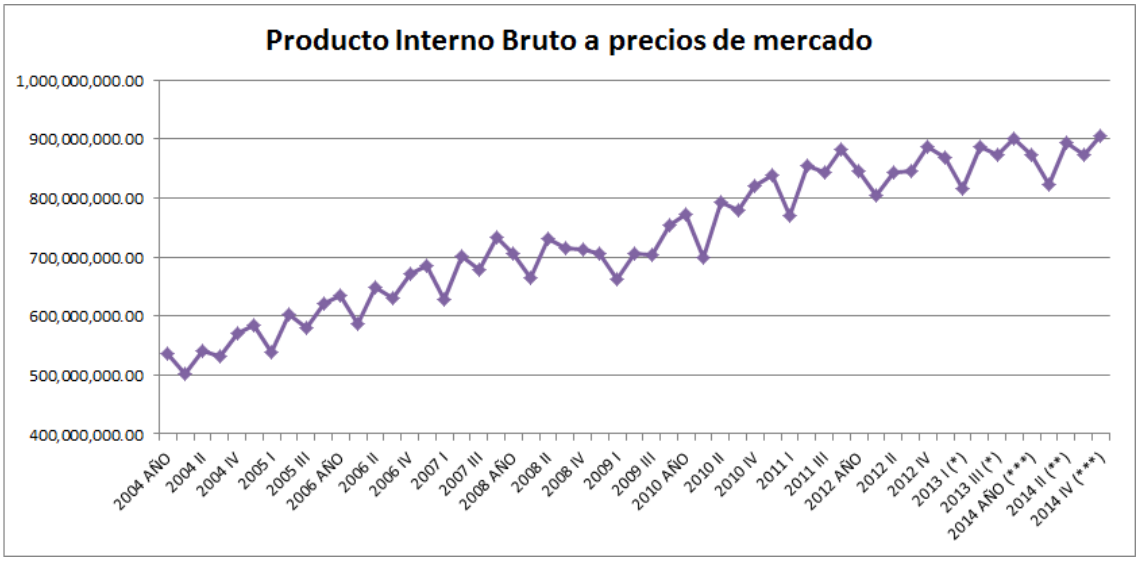


Figura 1 – Evolución del PIB en miles de pesos a precios de mercado en los últimos 10 años.

Fuente: INDEC http://www.indec.mecon.ar/nivel4_default.asp?id_tema_1=3&id_tema_2=9&id_tema_3=47

De acuerdo al Centro de Estudios para la Producción (CEP, un organismo dependiente de la Secretaría de la Industria, Comercio y de la Pequeña y Mediana Empresa) en el decenio 2003-2013 Argentina ha alcanzado un crecimiento del PIB industrial acumulado del 97% (Fuente: http://www.industria.gob.ar/cep/informes-y-estadisticas/industriales/).

Esto se dio en un contexto de crecimiento de la economía mundial donde es de público conocimiento que los precios internacionales de combustibles, minerales y alimentos alcanzaron records históricos.

Existe otro indicador determinado por el INDEC denominado Estimador Mensual Industrial (EMI), el mismo evalúa el desempeño del sector manufacturero en base a información

estadística proporcionada por diversos organismos públicos, cámaras empresariales y empresas líderes del sector. El cálculo se efectúa midiendo unidades físicas de producción en base al año 2006, el cual se considera con valor 100. En la Figura 2 puede verse la evolución de este índice en el período Enero-1994 – Enero-2015.

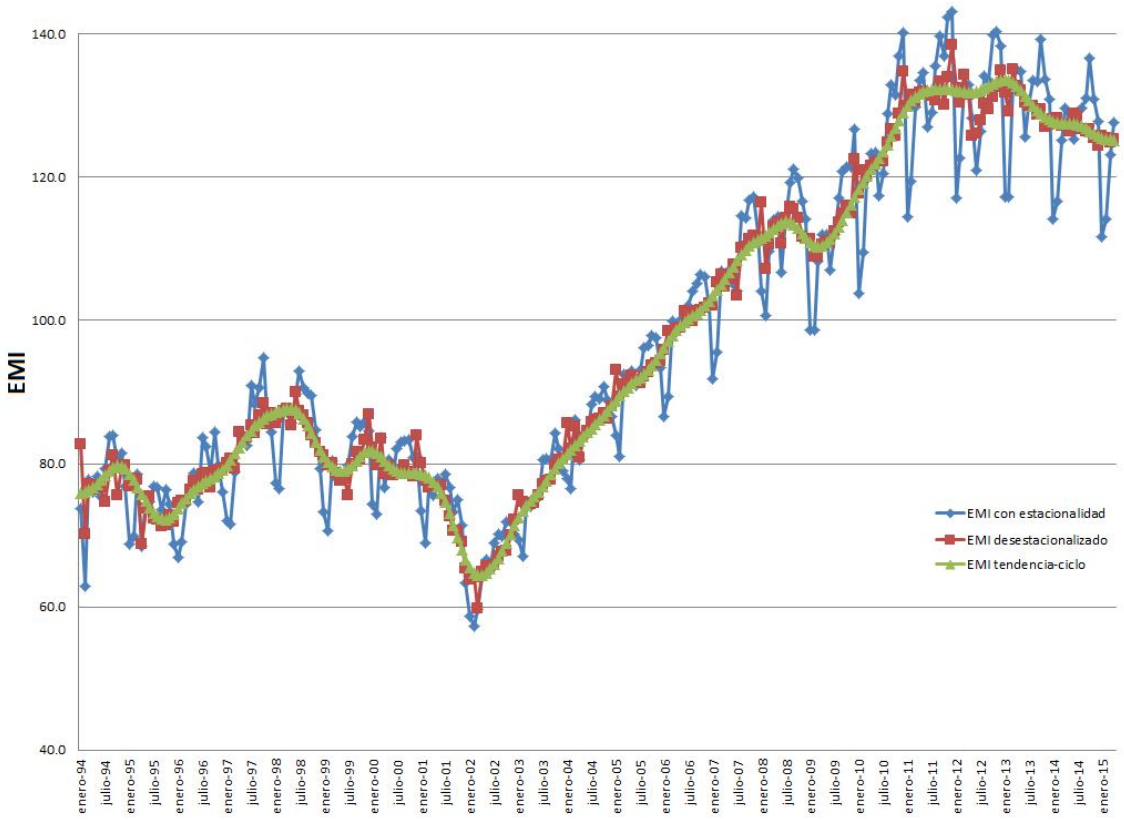


Figura 2 – Estimador Mensual Industrial Ene-94 – Ene-15

Fuente: INDEC http://www.indec.mecon.ar/nivel4_default.asp?id_tema_1=3&id_tema_2=6&id_tema_3=14

Si bien la velocidad de crecimiento se ha visto un tanto disminuida en los últimos tres años todo indica que las perspectivas de cara al futuro son buenas.

El reflejo en la realidad cotidiana de estos hechos es la creación, la consolidación y la ampliación de todo tipo de industrias destinadas a agregar valor a las más diversas de las materias primas. Asimismo existe un importante crecimiento en el sector de la generación, transporte y distribución de energía eléctrica así como de la exploración y explotación de hidrocarburos.

Estas nuevas industrias y empresas requieren apoyo para controlar sus actividades de producción, de manera de maximizar el uso de recursos, reducir tiempos, bajar costos, lograr economías de escala, ser compatibles con una política sustentable en términos medioambientales, etc.

En la actualidad no se concibe un proceso industrial o de supervisión sin el uso de sistemas o elementos computarizados para controlar o automatizar los diferentes elementos que lo componen (maquinarias, sensores, transductores, contactores, interruptores, seccionadores, válvulas, medidores, etc.)

Es fundamental entonces contar con una herramienta con la capacidad de centralizar la información adquirida de estos elementos para presentarla en tiempo real, de forma inteligible y práctica para el usuario u operario final, permitiéndole también actuar sobre las variables del proceso. Esta es la función principal de una Interfaz Hombre Máquina (HMI) y de un Sistema de Adquisición de Datos y Control de Supervisión (SCADA).

Los sistemas HMI/SCADA han estado presentes desde hace más de cincuenta años en la gran mayoría de las actividades de producción del mundo moderno, su crecimiento y desarrollo ha sido constante, su demostrada utilidad se encuentra más vigente que nunca.

Sin embargo se plantean nuevos desafíos de cara al futuro. El desarrollo incesante de tecnologías computacionales cada vez más poderosas y económicas, acompañado por nuevas herramientas de programación y las posibilidades que brindan las últimas redes de comunicaciones configuran un escenario ideal nunca antes visto. El mismo no debería ser desaprovechado por este tipo de sistemas.

1.2. Propósito del presente trabajo

El propósito del presente trabajo consiste en demostrar que en la actualidad están dadas las condiciones en nuestro país para la creación de un sistema SCADA de última generación que pueda aprovechar el contexto favorable tenga en cuenta los siguientes factores que en este apartado se desarrollan.

1.2.1. Panorama local

El mercado local de los sistemas de supervisión está dominado por soluciones de empresas internacionales. Se trata de herramientas complejas producto de décadas de desarrollo que tienen un muy alto costo y por lo tanto quedan restringidos a ser aplicados en grandes infraestructuras donde se realizan inversiones considerables de dinero.

Las pequeñas y medianas industrias están incapacitadas económicamente para acceder a estos sistemas y aunque no siempre requieren la complejidad que estos presentan no encuentran en el mercado una alternativa a su medida.

Sería un factor de cambio clave que estas organizaciones pudieran contar con herramientas de este tipo en el contexto antes descripto, pues son determinantes al adicionar un valor significativo en el desarrollo futuro de estas actividades.

En la actualidad no existen productos de origen nacional que brinden la funcionalidad de un sistema SCADA de manera completa.

1.2.2. Hardware potente y de bajo costo

Tradicionalmente los sistemas HMI/SCADA estuvieron ligados de manera obligada a costosísimas y voluminosas plataformas de hardware que tuvieran la capacidad de soportar aplicaciones de tales características. Esto incrementaba notablemente el precio total de la solución.

El constante avance de la tecnología de microprocesadores, memorias, placas madres y demás elementos de hardware y su incesante masificación pone al alcance de las empresas e industrias computadoras y servidores cada vez más potentes y a un costo muy accesible. Plataformas que antes eran prohibitivas por su precio para el sector de pequeñas empresas hoy ya no lo son.

En la actualidad un servidor básico, con características modestas en términos de performance y con un costo más que accesible es capaz de cumplir apropiadamente con los principales requerimientos de un sistema SCADA cuando años atrás la misma situación hubiera requerido una inversión millonaria.

1.2.3. Nuevas herramientas de programación de código abierto

Es notable como se han popularizado en los últimos años diversas plataformas, bibliotecas, herramientas y motores de base de datos de código abierto útiles para la construcción de aplicaciones de software, las cuales en general están soportadas por una comunidad mundial de usuarios que participan activamente en su desarrollo. Las mismas han sufrido un crecimiento tan importante que han equiparado en muchos casos en potencialidad y características a los productos comercializados por empresas como Microsoft, Oracle, etc.

De forma similar que con las plataformas de hardware esto pone al alcance de cualquier persona capacitada todo lo necesario para el desarrollo de un sistema HMI/SCADA, con todo el soporte de una comunidad mundial que genera permanentemente documentación de apoyo y de forma gratuita.

En el pasado gran parte de los sistemas HMI/SCADA dependían de licencias de terceros para su desarrollo y corrían en sistemas operativos privativos. Esto incrementaba su costo, hoy en día esta situación puede evitarse.

1.2.4. Redes de comunicaciones

Históricamente los sistemas SCADA de primera generación se comunicaron con los dispositivos desde donde se nutrían de información mediante vínculos punto a punto que eran o bien contratados o propiedad de la empresa que operaba el sistema. Se trataba de un sistema cerrado o auto-contenido con escasas posibilidades de interoperación con otros sistemas.

Los vínculos de comunicaciones eran principalmente implementados mediante radioenlaces o pares de cobre propios de la empresa o alquilados a compañías telefónicas. Se utilizaban módems, transmisores y repetidores para grandes distancias. Los vínculos satelitales, si bien eran menos comunes, también eran implementados. Las velocidades eran bajas y los cortes de comunicaciones frecuentes.

Esta topología de red de tipo estrella, en donde el sistema SCADA es el centro, es rígida, cerrada, con escasa resistencia a fallas y representa enormes costos. Por un lado gastos de mantenimiento para continuar operativa y también gastos de inversión si se pretende realizar una expansión.

En la actualidad con la aparición de Internet y las redes móviles se han logrado grandes avances que permitieron flexibilizar los esquemas de comunicación. En la mayoría de los casos ya no es necesario instalar vínculos punto a punto dedicados excepto en casos excepcionales donde sea preciso conectarse con un sitio remoto donde no llegue ninguna red de comunicaciones pública existente.

Por su parte, hoy cualquier punto con acceso a Internet o con cobertura de una red de comunicaciones móvil se transforma casi automáticamente en un punto de acceso remoto desde donde el sistema SCADA puede tomar información. La inversión es mínima, los anchos de banda más grandes y la confiabilidad es cada vez más alta.

Estas características hacen posible ampliar la densidad de información que puede supervisarse expandiendo las posibilidades de los sistemas SCADA.

1.2.5. Ataques informáticos y ciberseguridad

Dado que antes las comunicaciones se manejaban principalmente por vínculos en muchos casos propios o también contratados pero no al alcance de terceros la posibilidad de ataques informáticos era baja, por lo que históricamente esto nunca ha sido una preocupación.

Sin embargo en la medida que se han popularizado Internet y las redes móviles en donde los datos viajan por redes públicas es necesario implementar medidas que brinden algún grado de protección contra este tipo ataques, ahora plausibles.

Por lo antes mencionado los protocolos de telecontrol más utilizados en la actualidad para el transporte de información SCADA, por ejemplo Modbus o DNP3, no incorporan nativamente ningún tipo de encriptación o protección de seguridad o de integridad. Mientras tanto se ha visto como en la actualidad los ciberataques son una realidad preocupante que no puede ser pasada por alto.

Es menester contemplar este factor desarrollando contramedidas para evitar una posible afectación de las infraestructuras críticas a las que los sistemas SCADA se encuentren conectados.

1.3. Metodología utilizada

El presente trabajo se propone ser en principio una Tesis de investigación que proponga avances a una tecnología conocida, en nuestro caso los sistemas HMI/SCADA. La meta de esta Tesis será entonces realizar un análisis pormenorizado de la situación actual de esta tecnología, sus antecedentes y sus posibilidades tratando de determinar el sentido en el que debería evolucionar la tecnología de aquí en adelante. Más tarde tomando esta determinación como punto de partida se expondrá el desarrollo de un nuevo producto HMI/SCADA cuya pretensión será seguir los lineamientos evolutivos planteados.

1.4. Estructura del trabajo

1.4.1. Breve descripción

Durante el desarrollo del presente documento se realizará una investigación tecnológica de la historia de los sistemas HMI/SCADA desde sus orígenes tratando de identificar dentro del contexto actual cuáles son sus desafíos de cara al futuro. Identificados los mismos se planteará un objetivo que buscará resolver los problemas que la tecnología presenta mediante la proposición de mejoras y progresos.

A continuación se expone en detalle un sistema SCADA que pretende funcionar como ejemplo de la implementación de las mejoras y progresos que se le plantean a la tecnología. Finalmente se sacarán las conclusiones necesarias tratando de analizar si los planteos han resultado de utilidad.

1.4.2. Descripción por capítulos

Capítulo I: Introducción. Se presentan los sistemas HMI/SCADA y se describe el propósito del trabajo. Se analiza el panorama argentino de los sistemas de automatización y las oportunidades. Se explica la metodología utilizada y se describe la estructura del trabajo. Se enumeran las limitaciones del trabajo.

Capítulo II: Contexto. En este capítulo se definen y se explican en detalle en qué consisten los sistemas HMI/SCADA, su historia, se analizan los procesos en los que pueden aplicarse, los elementos que conforman un Sistema de Telecontrol y su esquema funcional. Se describe luego al elemento principal de adquisición de datos utilizado por los sistemas SCADA, la RTU (Remote Terminal Unit - Unidad Terminal Remota). Más tarde se analizan los Sistemas de Transmisión de Datos con los que los sistemas se comunican entre sí y con las RTUs, se estudia su rol, sus configuraciones más comunes y las tecnologías involucradas. Finalmente se exponen las distintas generaciones de los SCADAs. Se llega a la actualidad de este tipo de sistemas.

Capítulo III: Objetivo del trabajo. Se expone y se desarrolla el objetivo del trabajo, el desarrollo local de un nuevo sistema SCADA multiplataforma, basado en web de bajo costo que utilice herramientas de código abierto.

Capítulo IV: Punto de partida. Se identifican las falencias que los sistemas SCADA presentan a la hora de encarar los nuevos desafíos que el panorama actual propone, se analiza cómo debería ser el crecimiento y la evolución de cara al futuro.

Capítulo V: Esbozo de la solución. Se proponen una serie de mejoras y características concretas con las que debería contar el nuevo sistema SCADA. Se describe una arquitectura básica, explicando sus partes y fundamentando la elección de herramientas de programación para desarrollarlas. A continuación se analizan estas herramientas.

Capítulo VI: Conceptos del sistema. Se definen una serie de conceptos forjados durante el desarrollo del sistema, relacionados con diferentes áreas, funciones y características para ayudar a comprender globalmente el comportamiento del mismo.

Capítulo VII: Arquitectura del sistema. Se expone un diagrama funcional y se describen en detalle los tres módulos principales del sistema: SCADA o núcleo, de Configuración y Desarrollo y HMI. Se habla de sus funciones y su principio de funcionamiento.

Capítulo VIII: Símbolos y pantallas SCADA. Este capítulo detalla el manejo gráfico de archivos vectoriales que realiza en sistema para representar procesos industriales. Se exponen los diferentes tipos de símbolos y sus características y la herramienta de edición de pantallas.

Capítulo IX: Consideraciones de seguridad. Se plantean los riesgos de seguridad informática a los que los sistemas de control y automatización se encuentran expuestos.

Capítulo X: Análisis de resultados. Como cierre se evalúa el camino recorrido durante el trabajo, se analizan los resultados y se sacan conclusiones.

Anexo I: Modelo de datos del sistema. Exposición detallada del contenido de la base de datos del sistema, sus tablas, relaciones y significado y función de cada campo.

Bibliografía. Detalle de los libros, artículos, normas y documentación utilizada para la preparación del presente trabajo.

Glosario de términos. Listado de términos relativos al tema en cuestión utilizados a lo largo de esta tesis.

1.5. Limitaciones

El presente trabajo pretende analizar los desafíos de la tecnología de los sistemas HMI/SCADA principalmente centrados en actividades industriales, como la producción de bienes manufacturados o el procesamiento de materias primas; en la supervisión de grandes infraestructuras, como los procesos de distribución de agua potable, gasoductos, generación, transporte y distribución de energía eléctrica, tratamiento y transporte de petróleo, etc.

En función de esto se espera especificar y diseñar un sistema que se aproxime, al menos en un principio, a los requerimientos que este tipo de aplicaciones requieren, apuntando primero al mercado de las pequeñas y medianas empresas, aspirando con el tiempo a cubrir otro tipo de proyectos con necesidades y requerimientos más específicos y rigurosos.

Se deja como futuras líneas de investigación las posibilidades de mejora que este tipo de sistemas presentan para otras áreas de aplicación, que no son las relacionadas con actividades industriales o de producción. Se citan a continuación algunos ejemplos:

- ✓ Telemedicina.
- ✓ Sistemas de Seguridad y control de accesos.
- ✓ Control de tráfico vehicular.
- ✓ Agricultura de precisión.
- ✓ Meteorología.
- ✓ Domótica.

Estas áreas también muestran avances significativos, plantean nuevos desafíos y brindan nuevas posibilidades de desarrollo de la mano de las nuevas tecnologías de hardware,

software y comunicaciones. Son campos de investigación también valiosos para estudiar el rol que los sistemas HMI/SCADA deberían tomar en los mismos.

2. Capítulo II: Contexto

2.1. Descripción de la tecnología

2.1.1. ¿Qué es SCADA?

SCADA es un concepto que abarca una serie de tecnologías de control, estas tecnologías en conjunto permiten a un usuario de un determinado sistema colectar datos a distancia de una o más infraestructuras lejanas, enviando a su vez instrucciones de control.

Su aplicación hace innecesario que los operadores sean asignados a quedarse o visitar frecuentemente locaciones remotas donde las infraestructuras operan. El sistema incluye una interfaz para el operador y la manipulación de datos relativos a la aplicación, entre otros factores.

Su existencia se remonta a la creación de los sistemas de control. Los primeros sistemas SCADA utilizaron adquisición de datos por medio de paneles de medidores, luces y registradores gráficos. Los operadores utilizando perillas de control ejercían el control de supervisión. Estos dispositivos, de cinco décadas de antigüedad, eran y todavía son usados en algunos casos para ejecutar el control y la adquisición de datos de plantas, fábricas e infraestructuras de transporte y generación de energía.

2.1.2. Definición de SCADA

SCADA es un acrónimo que está formado por las primeras letras de las palabras de la frase "Supervisory Control And Data Acquisition", o Control de Supervisión y Adquisición de Datos.

Un sistema SCADA permite al operador en una ubicación central de un proceso distribuido geográficamente, como una refinería de petróleo, sistemas de tuberías, irrigación o complejos de generación hidroeléctrica, entre otros, cambiar ajustes en dispositivos distantes, abrir o cerrar válvulas o interruptores, monitorear alarmas, y recolectar información de control. Cuando las dimensiones del proceso se vuelven grandes se pueden apreciar los beneficios que los sistemas SCADA ofrecen al reducir el costo de las visitas rutinarias de personal de operación para monitorear instalaciones. El valor de estos beneficios será aún más grande si estas instalaciones son muy distantes y requieren un gran esfuerzo y gasto para ser visitadas.

La tecnología SCADA se refiere a la combinación de control, telemetría y adquisición de datos. El término SCADA engloba la recolección de la información, su transferencia al centro de control, la ejecución de los controles y análisis necesarios y la presentación de la información en un cierto número de pantallas de operación. Las acciones de control necesarias son enviadas de vuelta al proceso.

2.1.3. Procesos aplicables

La tecnología SCADA es más apropiada para procesos que se distribuyen sobre grandes áreas; que sean relativamente simples de controlar y monitorear; y que requieran una intervención frecuente, regular o inmediata. Los siguientes ejemplos de procesos ayudan a visualizar el rango de aplicaciones para las que un sistema SCADA es apropiado:

- ✓ Grupos de estaciones de pequeños generadores hidroeléctricos que son encendidos o apagados en respuesta a la demanda y están generalmente ubicados en sitios remotos, pueden ser controlados abriendo o cerrando las válvulas que permiten el pasaje de agua a la turbina. Deben ser monitoreados continuamente, y necesitan responder relativamente rápido a la demanda de la red eléctrica.
- ✓ Infraestructuras de producción de gas y petróleo incluyendo pozos, sistemas de recolección, equipamiento de medición de fluidos, y bombas. Estos equipos están diseminados sobre grandes áreas, requieren controles relativamente simples, como encender y apagar motores, necesitan recolectar información de telemetría regularmente, y responder eficazmente a las condiciones del resto de la instalación.
- ✓ Las tuberías para petróleo y gas, químicos o agua tienen elementos que se ubican en distancias variables del centro de control. Los mismos pueden ser controlados abriendo o cerrando válvulas, o arrancando y parando bombas, y deben ser capaces de responder rápidamente a las condiciones del mercado y a derrames de materiales sensibles para el medioambiente.
- ✓ Los sistemas de transporte de energía eléctrica, que cubren distancias de miles de kilómetros, pueden ser controlados abriendo o cerrando interruptores y deben responder instantáneamente a cambios en la carga de las líneas de transmisión.
- ✓ Los sistemas de irrigación a menudo cubren áreas de cientos de kilómetros. Requieren ser controlados abriendo o cerrando válvulas y necesitan la recolección de valores de telemetría para el agua que se suministra a los usuarios.

Estos ejemplos son sólo algunos casos de procesos en donde los sistemas SCADA han sido correctamente instalados y han representado una enorme evolución en la operación de estos sistemas. Asimismo a medida que la tecnología involucrada ha madurado la complejidad de estos sistemas lo ha hecho de la misma forma.

Las señales típicas a ser recopiladas de los sitios remotos incluyen alarmas, indicaciones de estado, valores analógicos, y valores totalizados de medidores. Sin embargo un amplio rango de información puede ser recolectada. De forma similar, las señales enviadas desde el sistema SCADA, ubicado en un centro de control, a los sitios remotos está usualmente limitada a cambios discretos de bits o valores analógicos relacionados con un dispositivo del proceso. Un ejemplo de un cambio de un bit sería la instrucción que ordena a un motor parar. Por su parte un ejemplo de un valor analógico sería la orden de cambiar el punto de ajuste de una válvula al 70%.

2.2. Historia

2.2.1. Desarrollo desde la telemetría

En los primeros dos tercios del siglo XX, la ingeniería de desarrollo de aviones y cohetes así como también la investigación del clima y otros parámetros geofísicos requirió que simples paquetes de datos sean recolectados de equipamientos ubicados en lugares donde era difícil o imposible utilizar observadores. Por ejemplo, en los primeros tiempos de la investigación de la aviación, los diseños tenían espacio para el piloto, pero muy poco lugar quedaba disponible para los ingenieros de diseño y testeo para acompañar el vehículo y monitorear los cientos de sensores instalados para evaluar las tensiones y esfuerzos del fuselaje y el motor.

De manera similar, los primeros cohetes no tenían espacio ni siquiera para los pilotos, y como en general todas las primeras experiencias terminaban muy repentinamente con explosiones y fallos, hubiera sido muy difícil en cualquier caso conseguir ingenieros o técnicos que voluntariamente aceptaran volar junto con los instrumentos. El hecho de que la mayoría de estos primeros viajes no solo terminaran rápido sino más pronto que lo esperado hizo doblemente necesario encontrar una forma de obtener la información de vuelo mientras estaba todavía disponible.

La tecnología fue aplicada primero a la predicción del tiempo, los científicos descubrieron que eran necesarios grandes volúmenes de datos para realizar pronósticos certeros. Sin embargo una muy pequeña cantidad de datos estaba disponible, desde donde las poblaciones estaban localizadas. Las oficinas de correo remotas, faros, barcos y estaciones meteorológicas especialmente establecidas podrían ser tripulados y transmitir los datos a una estación central, ya sea por teléfono, telégrafo o radio. Pero eso solo contaba para la información de superficie, y la meteorología se compone de mucho más que efectos superficiales. Para tener un mejor manejo del clima, los científicos pensaron que la solución era desarrollar perfiles de información meteorológica a través de la atmósfera. Se usaron globos meteorológicos de bajo costo, con pequeños instrumentos montados en ellos que medían parámetros de interés y proveían la información deseada. ¿Pero cómo podía esta información ser recolectada?

La respuesta a este problema surgió del método de comunicación que estaba siendo usado en otra industria para lidiar con una preocupación de seguridad. Desde hacía algún tiempo, los sistemas ferroviarios habían usado comunicaciones cableadas para monitorear la posición de su material rodante y el estado de los cambios de vía.

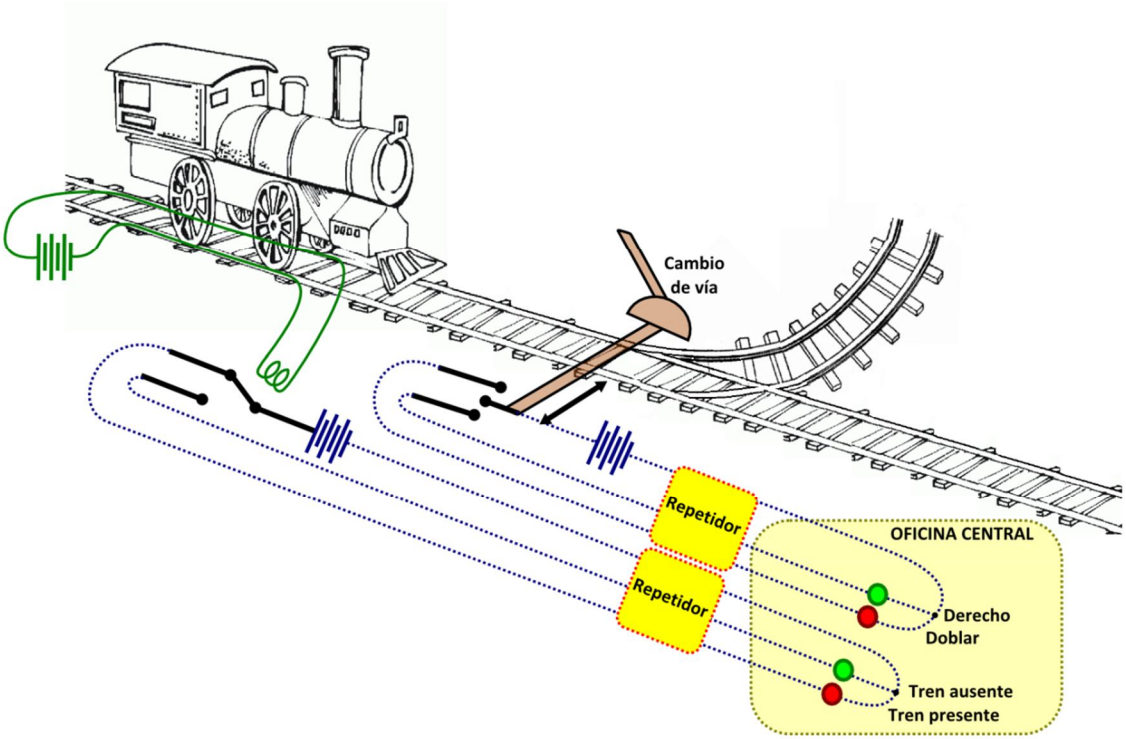


Figura 3 – Sistema de telemetría de control ferroviario

La Figura 3 muestra cómo se lograba esto. Se empleaban sólo interruptores de posición controlados eléctricamente, cable y luces de estado. Cuando las distancias eran muy largas se utilizaban repetidores para compensar las caídas de tensión en los cableados. Las conexiones remotas se concentraban en una oficina central en donde se instalaban varios tableros con luces indicadoras y registradores gráficos para representar el estado de las señales provenientes de sitios lejanos.

Este modo de comunicación primitivo, llamado desde entonces telemetría, permitía monitorear cuestiones que sucedían en locaciones remotas y permitía a los controladores de tráfico ferroviario planificar los viajes de manera más eficiente y segura. Con esta información disponible era posible enviar mediante líneas telegráficas instrucciones a los operadores situados en puestos lejanos para que cambiaran manualmente la posición de las vías.

El sistema estaba por supuesto limitado a infraestructuras fijas donde los cables podían ser tendidos entre la fuente de la señal y la oficina donde funcionaba el monitoreo. Pero donde estas condiciones se cumplieran la telemetría funcionaba bien.

Este rudimentario tipo de sistema SCADA es el más simple, pues las señales provenientes de campo se conectan directamente a los medidores, interruptores y luces de los tableros. Si se deseaba expandir parcialmente el sistema agregando nuevas señales e indicadores se podía realizar sin mayores complicaciones y era relativamente económico.

Sin embargo este tipo de sistema se vuelve inmanejable cuando se pretende instalar cientos de señales debido al volumen de cable a emplearse. Por su parte la cantidad de información es mínima y rudimentaria, no es posible el registro histórico de datos ni el monitoreo remoto de

alarmas y eventos, además se requería un operador visualizando los indicadores y medidores las 24 horas del día atento a posibles cambios que requirieran intervención.

Figura 4 – Operador utilizando un antiguo sistema de telemetría para control de una red ferroviaria

Fuente: http://www.michiganrailroads.com/RRHX/Pictures/Photos/061-070/Photo068C.htm

Sistemas de este tipo aún están en uso para la adquisición de datos de plantas, fábricas e infraestructuras de generación de energía.

2.2.2. Dependencia de las comunicaciones y las computadoras

2.2.2.1. Telemetría por radio

Al mismo tiempo que surgió la necesidad de comunicarse con elementos móviles, la tecnología de la radio también avanzaba. Mientras no se esperara obtener mucha información, la radio podía enviarla. Y mientras que no se tuviera que transmitir por mucho tiempo, la batería podía ser lo suficientemente pequeña para ser práctica. Entonces la tecnología de radio telemetría nacía. La Figura 5 muestra un sistema simple en donde un sensor de temperatura y un barómetro pueden conectarse a una radio para transmitir información de temperatura y presión en función de la altitud.

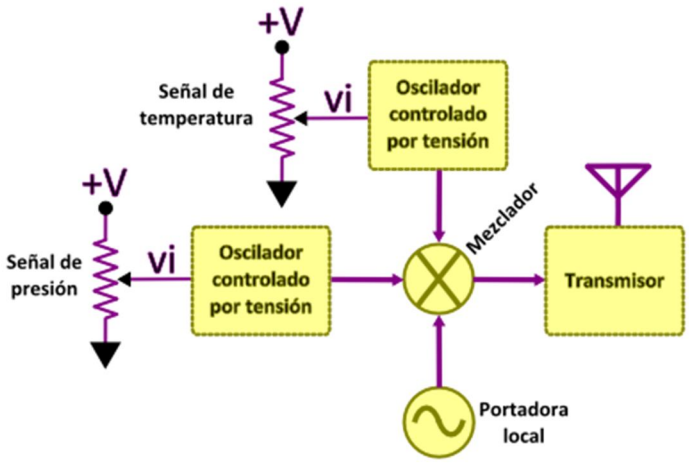


Figura 5 – Transmisión simple de señales de telemetría por radio

La telemetría por radio evolucionó con el tiempo mejorando la confiabilidad del sistema, incrementando en consecuencia la densidad de datos que podía ser transmitida. Se desarrollaron métodos de control y corrección de errores. Los equipamientos se hicieron más pequeños. Pero, en general, la telemetría por radio continuó por un largo tiempo un sistema de una sola vía, los datos eran recolectados del sitio remoto y transmitidos a la locación central.

2.2.2.2. Telemetría por cable

La telemetría por cable también maduró durante este período. En vez de concentrarse en mejorar la confiabilidad del medio, los ingenieros que trabajaban en este campo desarrollaron el concepto de comunicación de dos vías, que permitió a los cambios de vía de los trenes no solo ser monitoreados sino también controlados. Las empresas de energía eléctrica y las de transporte de fluidos por tuberías tenían infraestructuras que eran comparables con las de las empresas ferroviarias: grandes inversiones de capital que requerían control para operar más eficientemente.

Sin embargo los interruptores y válvulas que debían ser controladas estaban localizados en lugares lejanos e inconvenientes. Las empresas de producción de gas y petróleo tenían sus instalaciones ubicadas en lugares poco hospitalarios y poco desarrollados donde era difícil establecer viviendas para personal de operación. Todas estas industrias respondieron al mismo problema con la misma solución: investigaron formas de monitorear y controlar funciones simples remotamente usando cables y señales eléctricas para reducir los costos operativos de sus infraestructuras. Para el comienzo de la década del sesenta, el monitoreo remoto y el control de supervisión de varios procesos industriales era una tecnología en desarrollo.

Científicos e ingenieros reconocieron las mejoras y las incorporaron a la tecnología de telemetría por radio logrando un sistema de operación de dos vías. La radio podía ser localizada donde fuera necesario, en cualquier lugar de la superficie del planeta.

Habitualmente, la combinación de cable y radio era el método más efectivo para recolectar y distribuir señales. Junto con la evolución de la radio bajaron los costos de instalación y más infraestructuras pudieron pasar la evaluación de factibilidad. La instalación de cableado enterrado hacia áreas remotas podía ser extremadamente costoso. Los enlaces de radio son relativamente inmunes a las condiciones del terreno mientras haya línea de vista entre el transmisor y el receptor. Durante la década del sesenta la radio fue usada más y más. Para mediados de los setenta la radio se convirtió en el medio de comunicación más elegido para los nuevos sistemas de telemetría de dos vías a infraestructuras de ubicación fija.

2.2.2.3. Evolución de las computadoras

Mientras la radio evolucionaba, otra tecnología electrónica se desarrollaba. Las computadoras digitales hicieron su debut en sistemas de control de monitoreo remoto y supervisión a principios de los sesenta. La creciente flexibilidad que ofrecían era muy atractiva para los diseñadores de estos sistemas. Desde el principio los sistemas no computacionales tenían estaciones centrales que habían crecido en complejidad requiriendo hasta varios miles de relés. Las computadoras de tamaño mediano estuvieron disponibles para 1965, las condiciones estaban dadas para una explosión de sistemas que permitieran una masiva centralización del control. Las comunicaciones en gran escala, que habitualmente requerían largas líneas telefónicas dedicadas en combinación con vínculos privados de radio, proveyeron el camino a ese gran número de infraestructuras de campo. En general, el total de las instalaciones de campo de una empresa se conectaban a una computadora que se ubicaba a miles de kilómetros de distancia.

A principios de los años setenta el término SCADA era forjado y la palabra telemetría dejó de ser tan usada para describir estos sistemas de dos vías. La radio se volvió tan confiable que eventualmente reemplazó los sistemas de cable enterrado. Los roedores y las retroexcavadoras no podían hacer daño a las señales de radio pero sí al cableado enterrado. El control de estabilidad de frecuencia de las radios mejoró significativamente, requiriendo menos mantenimiento. La reducida complejidad del equipamiento y el diseño industrial mejorado de los radios permitieron realizar servicios de mantenimiento en campo con técnicos menos calificados y con instrumentos menos complicados.

La tecnología SCADA maduró lentamente durante la última parte de los años setenta. Las mejoras en el software mejoraron las interfaces hombre-máquina. Se desarrollaron generadores de reportes para contar con la información deseada cuando era requerido. Los sistemas crecieron. Como era el caso de la mayoría de las tecnologías industriales, el desarrollo de potentes minicomputadoras tuvo un profundo impacto en el desarrollo de la tecnología SCADA. Ahora las instalaciones más pequeñas podrían ser consideradas para operación remota. Quizás el cambio más profundo, sin embargo, fue que la minicomputadora se volvió tan económica que ya no era necesario centralizar el sistema. Por supuesto la centralización todavía podía ser llevada a cabo donde tuviera un sentido operacional, pero ahora los costos del hardware ya no eran un determinante tan importante. Las industrias eléctricas y de transporte de fluidos por tuberías mantuvieron su filosofía centralizada, las empresas de producción de petróleo y gas pasaron a ser más descentralizadas para volver a poner el control de cada infraestructura en manos del especialista en operación del lugar.

A medida que los sistemas proliferaron, los espectros de radio se volvieron cada vez más utilizados y se volvió cada vez más difícil encontrar frecuencias libres para usar. Mientras esto pasaba otros métodos de comunicación por radio fueron desarrollados, como las comunicaciones por satélite y celulares. Los costos de estos tipos de comunicaciones cayeron hasta llegar a ser, con pocas modificaciones, la tecnología elegida en muchos sistemas SCADA.

Esta situación se mantuvo de esta forma durante gran parte del siglo XX. Más tarde la llegada de Internet y las redes móviles ha cambiado radicalmente el panorama de los sistemas SCADA.

2.3. Sistemas de Telecontrol

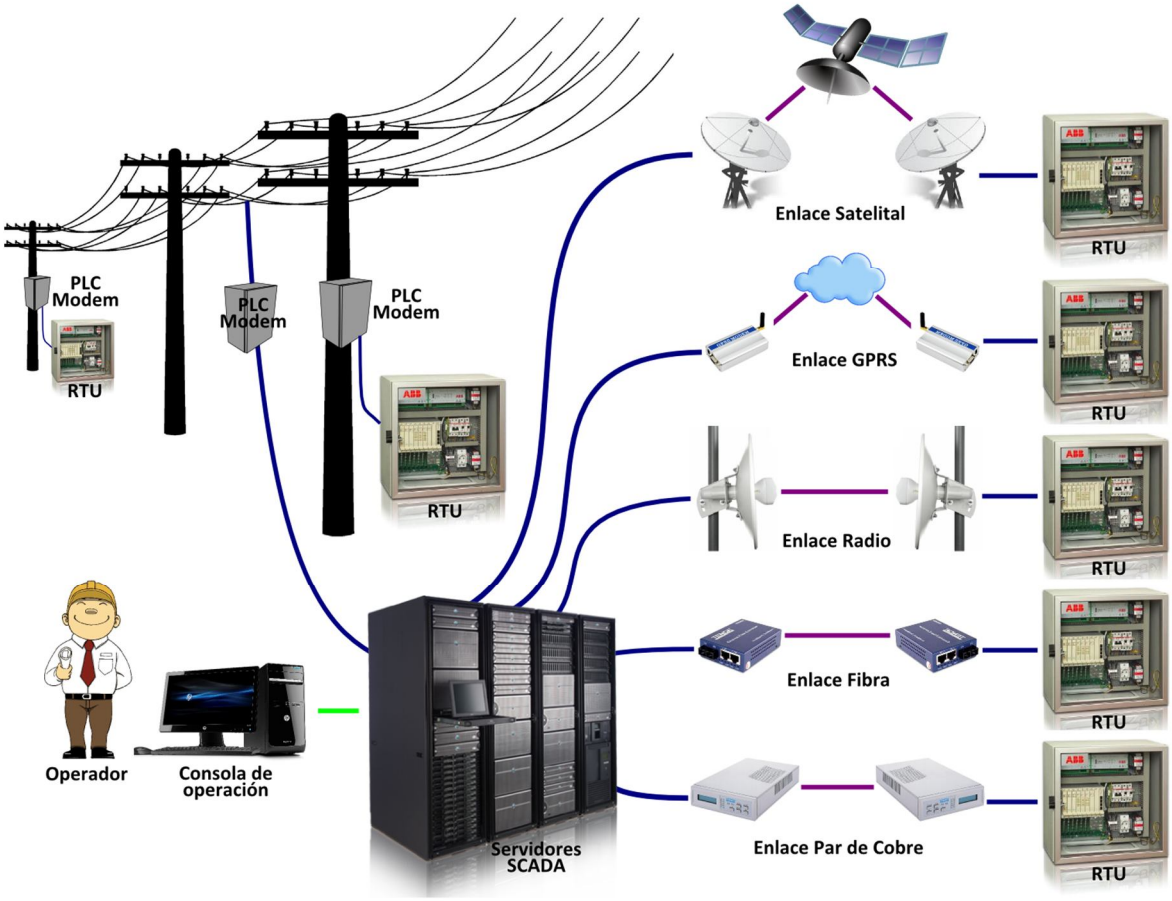


Figura 6 – Esquema de un Sistema de Telecontrol

La Figura 6 muestra los componentes principales de un sistema de telecontrol. En el centro del mismo se ubica el SCADA y los servidores que soportan sus funciones. En la esquina inferior izquierda se encuentra el operador que accede al sistema utilizando un dispositivo interfaz que suele llamarse consola de operación. La consola funciona como la ventana del operador al proceso. Cuenta con una pantalla que muestra datos en tiempo real del proceso y dispositivos de entrada, como teclado, mouse, dispositivos táctiles para que el operador pueda enviar comandos de vuelta al proceso. Cuando se produce una condición de alarma a través de ella se oirá una señal.

La consola de operación se conecta a uno o más servidores encargados de soportar las funciones del sistema. El sistema puede monitorear y controlar el campo aunque el operador no esté presente. Lo hace mediante un planificador que puede ser programado para repetir instrucciones a intervalos determinados. Por ejemplo, puede ser programado para requerir una actualización de cada unidad terminal remota (RTU o Remote Terminal Unit) cada 30 segundos.

Algunas de las características comunes de los sistemas SCADA se enumeran a continuación.

- ✓ **Interfaz de usuario**
  Herramienta mediante la cual el operador del sistema interactúa con el proceso en análisis. Nexo entre el hombre y la máquina.
- ✓ **Pantallas de operación**
  Representaciones pictóricas de los procesos o situaciones de interés de los usuarios del sistema que son alteradas gráficamente y en forma dinámica en función de los cambios del proceso o situación a la cual están asociadas.
- ✓ **Alarmas**
  Situaciones de distinto grado de criticidad que hayan ocurrido dentro del ámbito de supervisión del Sistema de Telecontrol y que requieren atención de un operador.
- ✓ **Gráficos de tendencias**
  Representación de la evolución en el tiempo de parámetros del proceso que el sistema supervisa.
- ✓ **Escalabilidad**
  Posibilidades de crecimiento de la capacidad del sistema para acompañar una expansión o un aumento de la complejidad del proceso bajo supervisión.
- ✓ **Uso de motores de base de datos**
  Herramienta de software utilizada para el almacenamiento sistemático de toda la información recolectada por el sistema del proceso industrial en análisis para su posterior uso.
- ✓ **Interfaz de comunicaciones para RTU y PLCs**
  Puertos de conexión mediante los cuales el sistema se comunica con dispositivos para interrogarlos acerca del estado actual de sus entradas y salidas.
- ✓ **Almacenamiento histórico**
  Capacidad del sistema de registrar periódicamente en la base de datos el estado actual de todos los estados, mediciones y demás informaciones recolectadas para su posterior análisis.
- ✓ **Disponibilidad y confiabilidad**
  Porcentaje de tiempo donde el sistema está operativo y listo para ser utilizado en toda su capacidad.
- ✓ **Redundancia y tolerancia a falla**
  Característica mediante la cual se dispone de varios equipos o procesos para mejorar la performance de operación o para reducir el riesgo de indisponibilidad o mal funcionamiento.
- ✓ **Procesamiento distribuido del tipo cliente/servidor**
  Modelo de comunicación en donde se distribuyen las tareas informáticas entre las que proveen recursos o servicios (servidores) y las que los demandan (clientes). Las tareas

del tipo servidor y las del tipo cliente pueden ejecutarse o no en la misma computadora.

2.3.1. Esquema funcional de un Sistema de Telecontrol

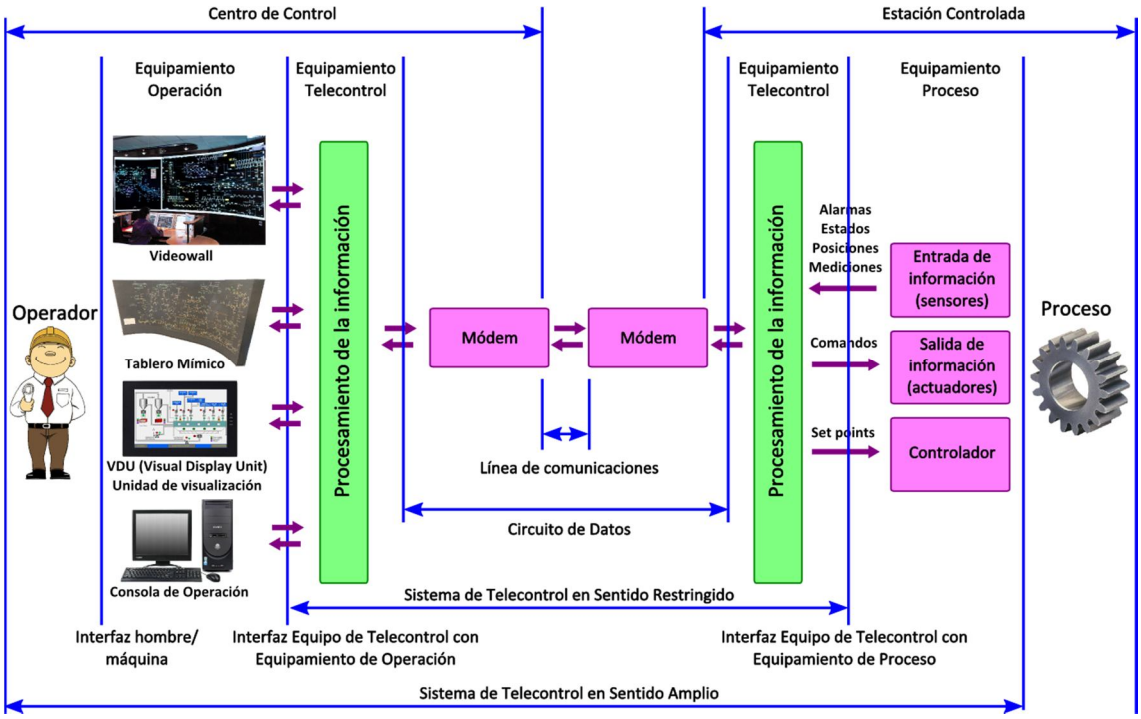


Figura 7 – Esquema funcional de un Sistema de Telecontrol (configuración punto a punto)

Una estructura típica punto a punto se muestra en la Figura 7. En ella se pueden observar los módulos funcionales básicos de equipamiento o subsistemas que son:

- ✓ Adquisición de datos mediante Entradas y Salidas de información del proceso.
- ✓ Procesamiento de la información en la Estación Controlada.
- ✓ Transmisión de datos a través de líneas de comunicaciones.
- ✓ Procesamiento de la información en el Centro de Control.
- ✓ Entradas y salidas para el Operador a través de Interfaz Hombre-Máquina (HMI).

Un sistema puede, desde el punto de vista operacional, gobernar el proceso en su conjunto o bien dividirse en diferentes niveles de responsabilidad o en subsistemas parcial o totalmente independientes. Los sistemas de control locales son usualmente independientes del Sistema de Telecontrol general, pero cierto tipo de información fluye en ambos sentidos. El uso de microprocesadores en todos los niveles del sistema, permite diseñar arquitecturas con inteligencia distribuida, de modo de pre-procesar la información evitando así el envío de datos superfluos o redundantes.

2.3.2. Funciones de los Sistemas de Telecontrol

Las funciones de los Sistemas de Telecontrol pueden dividirse en las siguientes:

2.3.2.1. Funciones de aplicación

Son la que cubren las necesidades del proceso para los cuales se aplica el sistema, ocupándose de los tipos de información que emana el proceso desde y hacia el operador. Esta información es transferida al Sistema de Telecontrol por medio de señales y manipuladas dentro del sistema en forma de datos. Pueden dividirse en dos categorías:

Funciones básicas

Las mismas se ocupan de todos los tipos de información desde y hacia el proceso y el operador. Las principales son: Monitoreo (señalización a distancia, telemetría, conteo a distancia) y Comando y Control (comandos a distancia, regulación a distancia).

Funciones de proceso extendidas

Se derivan de las funciones básicas actuando sobre las entradas y/o salidas por medio de funciones de proceso operacional. Estas funciones pueden ser ejecutadas por los ordenadores centrales o por sistemas separados. Algunos ejemplos son: indicación de límites, interpretación automática de alarmas, estimación de estados en tiempo real, registros cronológico de eventos, etc.

2.3.2.2. Funciones de procesamiento operacional

Aseguran la correcta adquisición de datos y proveen una apropiada presentación de los datos. Algunas funciones típicas son: adaptación de señales entrada/salida con interfaces cruzadas hacia el operador y hacia el proceso, supresión de rebotes de contactos, cálculo de valores de medidas, etc. La tecnología actual permite la implementación de estas funciones donde la información es adquirida, lo que da lugar al pre-procesamiento de la misma, lo que reduce la cantidad de datos a ser transmitidos previniendo posibles sobrecargas en los canales de comunicación y ordenadores en situaciones de emergencia.

La Interface Hombre-Máquina se define como la frontera entre el terminal del sistema de procesamiento de datos y el operador. Los requerimientos básicos para establecer una comunicación óptima es proveer a los operadores y al personal de mantenimiento con una información adecuada y confiable del estado de la red y del sistema de telecontrol propio y hacer uso de facilidades que permitan intervenciones posibles en el proceso. Los tipos de información mencionada son típicamente ejecutados por los siguientes componentes: Tablero mímico, Consola de Operación con sus correspondientes periféricos, Equipos de registro (impresoras, plotters, etc.), unidades de visualización, videowalls, sistemas de señalización acústica, etc.

2.3.2.3. Funciones de transporte de datos, redes y enlaces físicos

Cubren todas las funciones involucradas en el manejo de transferencias de información eficiente y confiable entre estaciones. Las típicas funciones son: Organización de prioridades para transferencias de información espontáneas y cíclicas, Estrategias de recuperación de errores, Generación y monitoreo de códigos de detección de errores, Generación y monitoreo

de estructuras de sincronización, respuesta y detección de errores de tamaño en la estructura de los mensajes, etc.

2.3.3. Unidad Terminal Remota

Una Unidad Terminal Remota o UTR (RTU por su sigla en inglés) es un dispositivo independiente de adquisición de datos y control. Es un elemento fundamental de los Sistemas de Telecontrol.

Su tarea principal es controlar y adquirir datos de equipamiento de un cierto proceso en un sitio remoto para luego transferirlos mediante un Modem y un transmisor hacia un Centro de Control donde opera un sistema SCADA. Posee una Unidad Central de Procesamiento (o CPU en inglés) que permite definición local de lógicas de control. Las RTUs también pueden comunicarse entre sí, por lo que pueden encadenarse extendiendo su área de operación.

Los servidores SCADA deben comunicarse con las RTUs que están localizadas en sitios lejanos respecto del centro de control. Un sistema SCADA puede tener desde una RTU hasta cientos de ellas. Las RTUs recolectan datos de campo enviándolos al Centro de Control a través del sistema de comunicaciones.

El sistema de comunicaciones define el camino para la comunicación entre la estación maestra y los sitios remotos. Este puede ser cableado, por fibra óptica, radio, líneas telefónicas, microondas o incluso vía satélite. Se usan protocolos específicos y métodos de corrección de errores para optimizar la transferencia de datos. Más adelante nos referiremos a estos aspectos con mayor profundidad.

El Centro de Control, en su condición de estación maestra, recolecta datos de varias RTUs y generalmente brinda al operador una interfaz para presentar la información y controlar los sitios remotos. En grandes sistemas de telemetría, también existen estaciones sub-maestras recolectan datos de sitios remotos y la reenvían a la estación maestra de control.

La información oportuna y precisa permite la optimización de la operación de la planta y el proceso. A menudo los beneficios permiten una operación más eficiente, confiable y, lo más importante, segura. El resultado es un costo de operación más bajo comparado con los primeros sistemas no automatizados.

La RTU se conforma de una serie de partes con distintas funciones, en general encontramos una o varias fuentes de alimentación, una o varias unidades centrales de procesamiento (CPU), entradas y salidas analógicas y digitales, módulos de comunicaciones de varios tipos, etc. La configuración puede ser monolítica (en un solo gabinete o caja) o modular mediante placas que se enchufan sobre un bus central de datos interno. Se detallan más adelante en profundidad las funciones y características de estas partes.

En la siguiente imagen vemos una RTU de la marca ABB, comercializada como RTU560. En este caso particular vemos que la configuración es modular, lo que significa que el dispositivo puede crecer en funciones y capacidad de acuerdo al requerimiento del usuario. Se pueden agregar entradas o salidas, fuentes de alimentación, etc. La posibilidad de expansión es la

ventaja principal de las RTUs de este tipo por sobre las monolíticas, pero su costo es mayor en comparación.

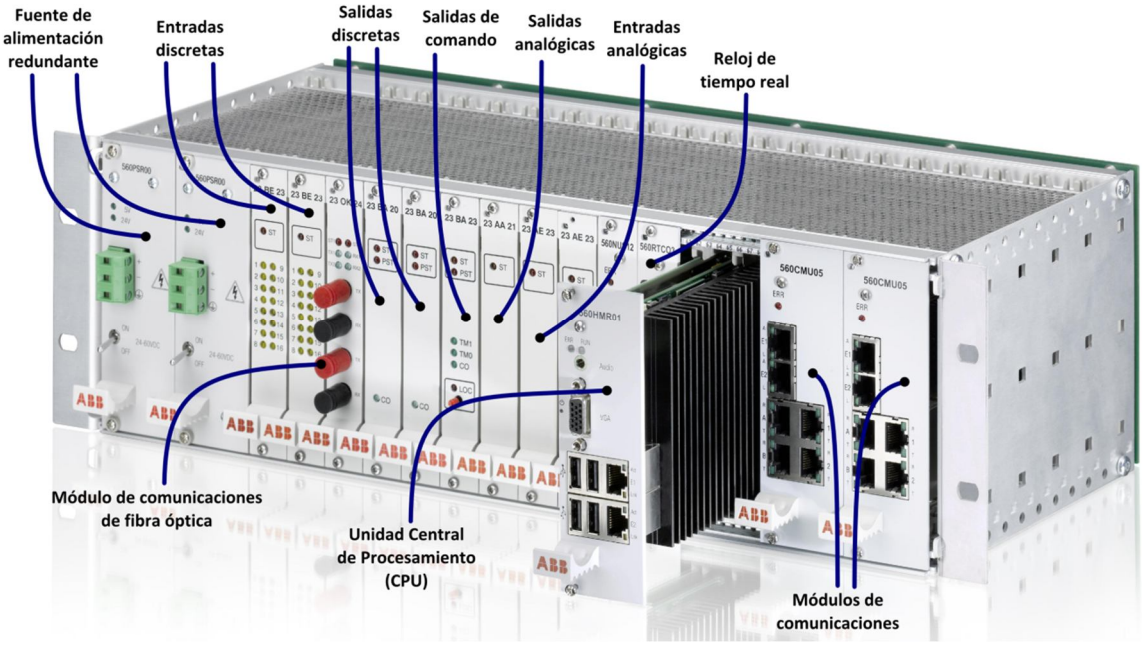


Figura 8 – Imagen de la RTU560 de la marca ABB

Fuente: http://www.abb.ch/cawp/seitp202/ce837fda50b470ccc1257b810021baa3.aspx

Cada RTU tiene la capacidad de entender el mensaje que le fue dirigido, decodificarlo, actuar en consecuencia, y responder de ser necesario. Luego quedará a la espera del siguiente mensaje. El procedimiento de actuación requiere evaluar la posición actual del dispositivo de campo, compararla con la posición requerida, enviar una señal eléctrica al dispositivo que le ordena cambiar de estado, chequear una serie de entradas para verificar que la orden fue obedecida y enviar un mensaje de respuesta al servidor SCADA para confirmar que el procedimiento fue completado.

La RTU provee una interfaz a los sensores analógicos y digitales de campo situados en cada sitio remoto. La conexión de la RTU con los dispositivos de campo se realiza mediante conductores eléctricos. Usualmente la RTU provee la energía necesaria para alimentar los sensores y los actuadores de baja potencia.

Las señales recibidas desde los sensores y enviadas a los actuadores son tensiones o corrientes. Por ejemplo para una entrada digital la presencia o ausencia de una tensión indicará el estado de un interruptor, para una entrada analógica la variación de corriente entre 4 y 20mA proveniente de un sensor de temperatura indicará el valor actual de temperatura del mismo. Esto se conoce como cableados “duros” ya que no corresponden a líneas de comunicaciones.

Originalmente existía solo una RTU por cada sitio lejano o infraestructura que requería ser supervisada desde el centro de control. La misma era responsable de todos los sensores y

actuadores presentes en esa infraestructura. Actualmente la tendencia es descentralizar cada vez más esa situación, incorporando mayor cantidad de sub-RTUs más pequeñas y más cercanas a cada parte del proceso. Esto ha disminuido significativamente la cantidad de cableados, ya que donde antes se requería al menos un par de cables por cada sensor y actuador ahora puede agruparse una parte de ellos bajo la supervisión de una sub-RTU.

Luego las comunicaciones con cada sub-RTU se concentran en un Gateway que a su vez se conecta al centro de control.

Se presenta a continuación un diagrama funcional de una RTU genérica, luego se explica con mayor detalle cada una de sus secciones.

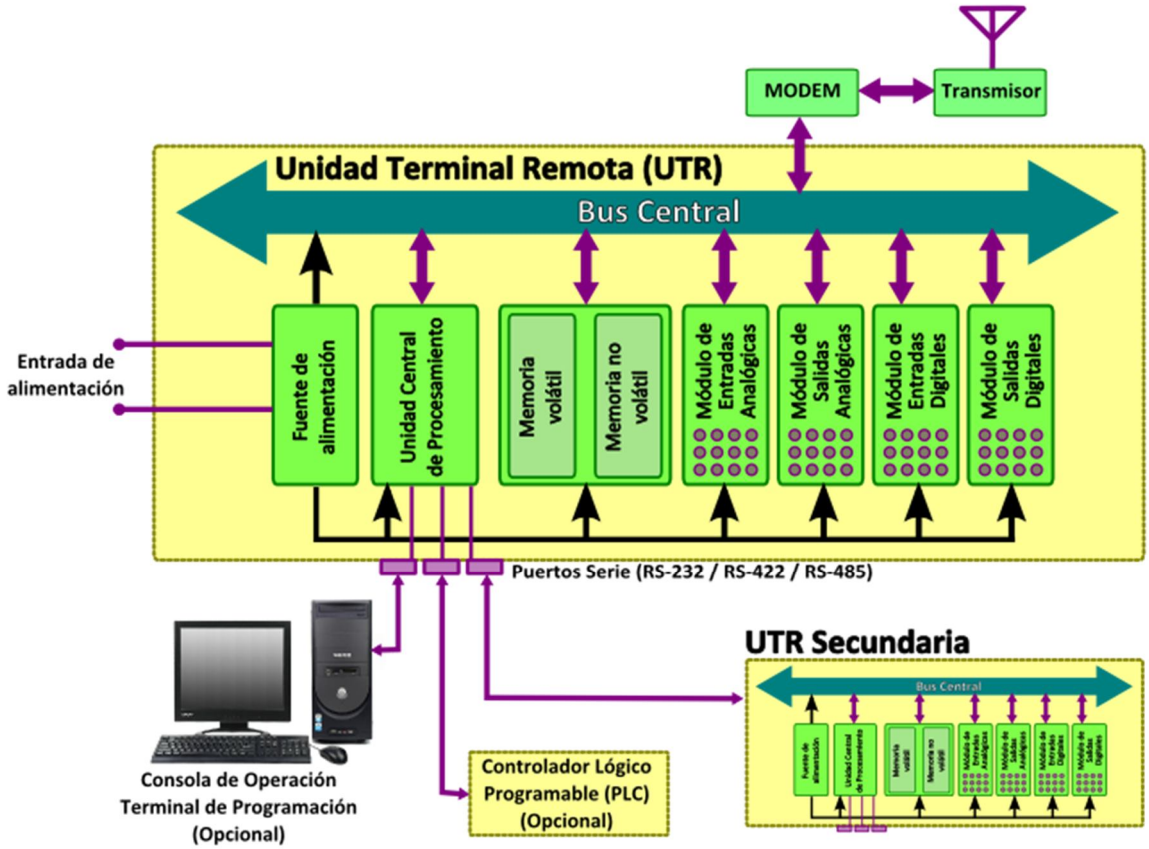


Figura 9 – Diagrama funcional de una RTU

2.3.3.1. Unidad Central de Procesamiento (CPU)

Posee en general un microprocesador de 16 o 32 bits, memoria EPROM/Flash para el almacenamiento de la configuración, memoria RAM para la ejecución y puertos de comunicaciones del tipo RS-232/RS-422/RS-485, Ethernet, Fibra óptica para conectarse a herramientas de diagnóstico, a consolas de operación y al centro de control mediante un vínculo de comunicaciones.

En general poseen LEDs y displays para facilitar la búsqueda de fallas y el diagnóstico de problemas (falla de CPU, falla de módulos, etc.).

Además incorporan un reloj de tiempo real con diferentes grados de precisión el cual es usado para el etiquetado de eventos.

Un watchdog verifica que la RTU se encuentre en operación normal. De producirse un timeout se indica el error y opcionalmente puede resetear la RTU.

2.3.3.2. Módulos de entradas analógicas

Hay cinco componentes principales que conforman un módulo de entradas analógicas:

- ✓ Multiplexor de entrada.
- ✓ Amplificador de señal de entrada.
- ✓ Circuito de muestreo y retención.
- ✓ Conversor analógico-digital.
- ✓ Interfaz hacia el bus de comunicaciones interno de la RTU.

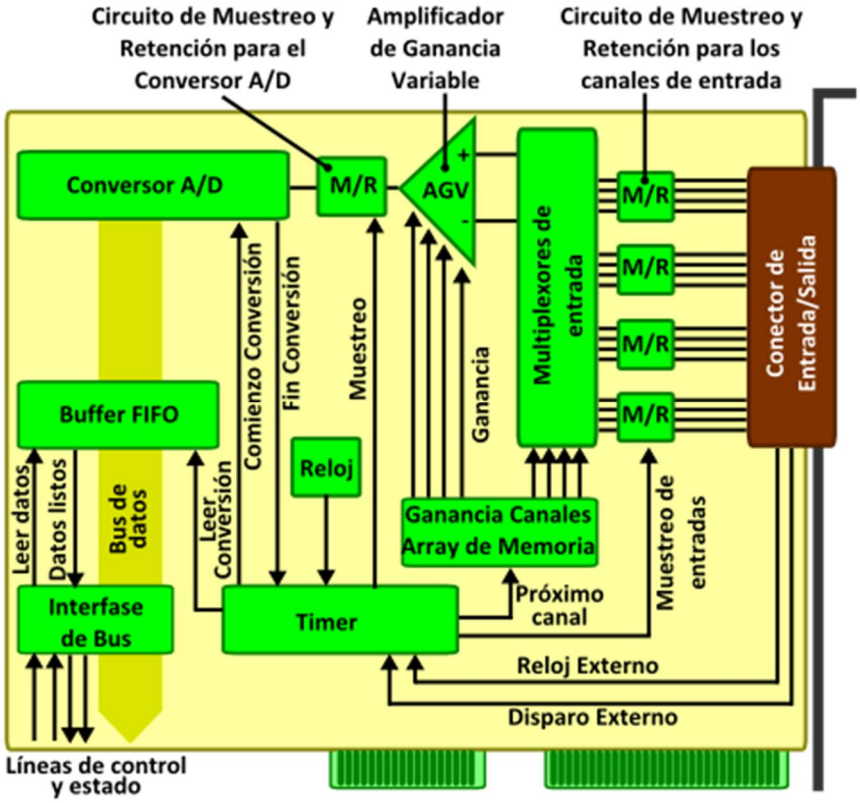


Figura 10 – Diagrama en bloques de un módulo de Entradas Analógicas

Multiplexor de entrada

Un multiplexor es un dispositivo que muestrea varias entradas analógicas en orden y las vincula a la salida en secuencia. La salida generalmente se conecta a un conversor analógico/digital. Esto elimina la necesidad de convertir cada canal de entrada.

Amplificador

Cuando tensiones de bajo nivel requieren ser digitalizadas, éstas necesitan ser amplificadas para equiparar el rango de entrada del conversor analógico digital, de otra forma se perdería precisión en la conversión. Para esta tarea se suelen usar amplificadores diferenciales aprovechando su característica de rechazo al ruido.

Un amplificador diferencial ideal responde solo a las diferencias de tensión entre sus dos terminales de entrada, siendo su salida proporcional a la diferencia entre estas, sin importar la señal de modo común presente en ambos terminales. Desafortunadamente en la realidad las tensiones de modo común si producen errores de salida. Se define entonces el parámetro conocido como Rechazo al Modo Común o CMRR (Common Mode Rejection Ratio):

$$CMRR = 20 \log (Vcm / Vdiff) [dB]$$

Donde Vcm es la tensión aplicada a ambos terminales de entrada y Vdiff es la tensión de salida (tensión de error) cuando se aplica Vcm. Se considera un valor apropiado para este parámetro 80dB o mayor.

Otro factor a considerar es la especificación de deriva térmica. Si un amplificador está configurado para dar una tensión de salida cero a una determinada temperatura, la salida cambiará a lo largo del tiempo si hay un cambio de temperatura. Este parámetro se mide generalmente en PPM/unidad de tiempo y PPM/ºC.

Circuito de Muestreo y Retención

La mayoría de los conversores analógicos/digitales requieren un tiempo fijo donde la señal de entrada debe permanecer constante para poder realizar la conversión. Esto se conoce como tiempo de apertura. Si la señal de entrada cambiara durante este intervalo, la conversión devolvería un valor incorrecto. Para evitarlo se utiliza un circuito de muestreo y retención en la entrada del conversor analógico/digital. Este muestrea la señal de salida del multiplexor y la mantiene constante durante el tiempo de apertura del conversor.

Conversor Analógico/Digital

Es el corazón del módulo, su función es medir una señal analógica de entrada y entregar como resultado un código digital correspondiente a la tensión de entrada. Existen varias formas de implementar conversores de este tipo, a continuación se detallan como ejemplo dos de ellas:

- ✓ Con integrador o de doble rampa: Son usados para aplicaciones de muy baja frecuencia (como máximo algunos cientos de Hertz). Pueden tener una muy alta precisión, por ejemplo 22 bits. Utilizan un capacitor que se carga con la señal de entrada durante un tiempo fijo. Luego se calcula cuanto demora el capacitor el descargarse. Esta demora es proporcional a la tensión de entrada.
- ✓ Por aproximaciones sucesivas: Permiten tasas de muestreo mucho más altas hasta cientos de kHz con precisión aceptable, por ejemplo 12 bits. El algoritmo de conversión es similar a una búsqueda binaria, donde se comienza comparando la entrada con una tensión de referencia correspondiente a la mitad del rango. Si se

encuentra en la primera mitad el primer dígito será cero y se repite la comparación utilizando la primera mitad del rango. Si se hubiera encontrado en la segunda mitad, el primer dígito sería uno. Se continúa de la misma forma, dividiendo por la mitad hasta completar todos los dígitos.

Métodos de conexión

Hay dos métodos de conectar señales de entrada a la placa de adquisición de datos: Entrada común y entrada diferencial. En general las entradas diferenciales son usadas para máxima inmunidad al ruido.

- ✓ **Entradas comunes:** Las placas con esta configuración tienen un solo cable de entrada para cada señal, las tierras de cada señal de entrada se conectan todas a un mismo terminal. Esta configuración pierde el rechazo al modo común y es muy sensible al ruido. No es recomendable usarlo con cables de conexión largos. La ventaja es que permite una mayor cantidad de entradas, y una conexión más simple.

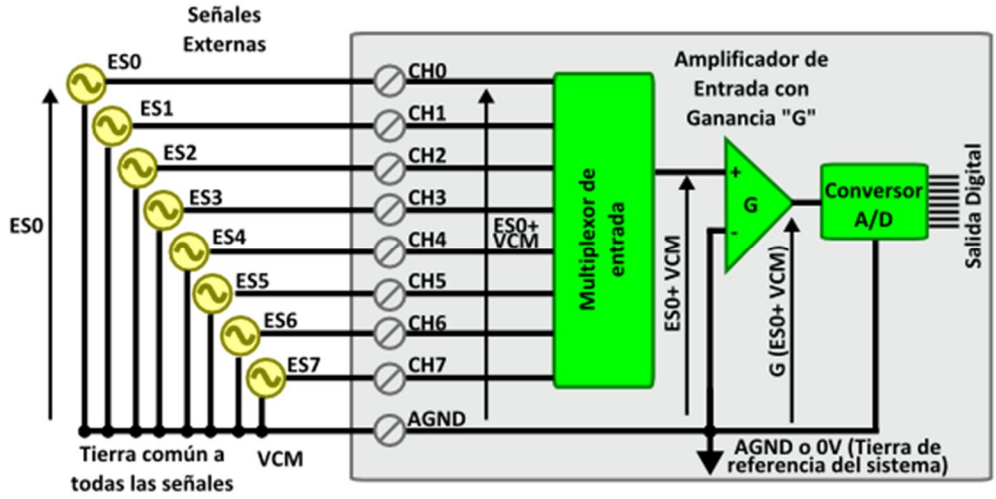


Figura 11 – Método de conexión mediante entradas comunes

- ✓ **Entradas diferenciales:** Permiten una gran inmunidad al ruido. También se utiliza cuando las tierras de las señales de entrada son distintas y no pueden ser unidas. Se requiere en este caso dos multiplexores de entrada y se tiene la mitad de los canales respecto del modo de entrada común.

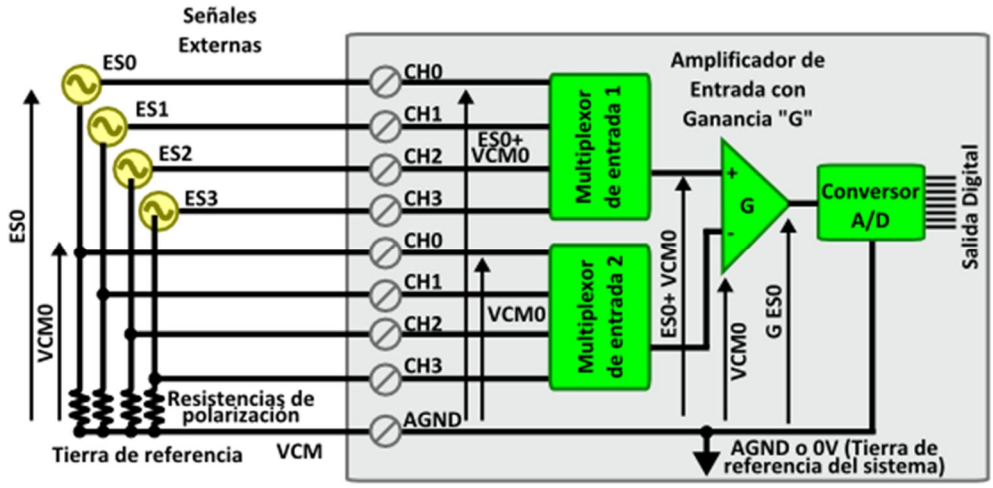


Figura 12 – Método de conexión mediante entradas diferenciales

Especificaciones típicas de módulos de entradas analógicas

Los módulos típicos tienen las siguientes características:

- ✓ Cantidad de entradas: 8, 16, 32, etc.
- ✓ Resolución: 8 bits, 12 bits, etc.
- ✓ Rango: 4-20 mA, +/-10V, 0-10V.
- ✓ Resistencia de entrada: 240 kΩ a 1 MΩ.
- ✓ Tasa de conversión: 10 microsegundos a 30 milisegundos.
- ✓ Entradas generalmente con terminal común.

2.3.3.3. Módulos de salidas analógicas

Especificaciones típicas de módulos de salidas analógicas

- ✓ Cantidad de salidas: 4, 8, 16, etc.
- ✓ Resolución: 8 bits, 12 bits, etc.
- ✓ Rango: 4-20 mA, +/-10V, 0-10V.
- ✓ Tasa de conversión: 10 microsegundos a 30 milisegundos.

Se debe considerar que la resistencia de carga del módulo no sea inferior a la especificada (típicamente 50 kΩ) o la corriente a través de la misma será excesiva. En general se prefiere brindar una tensión de salida en lugar de una corriente ya que el requerimiento de energía es menor.

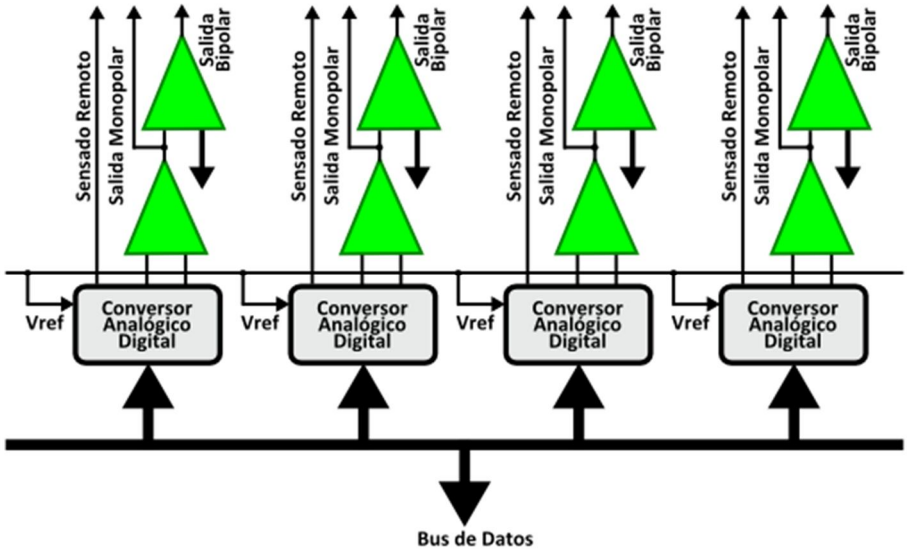


Figura 13 – Módulo de salidas analógicas

2.3.3.4. Módulos de entradas digitales o discretas

Se utilizan para indicar valores como estados y señales de alarmas.

Las señales de estado provenientes de una válvula comprenden dos interruptores de fin de carrera. El cierre de uno de ellos indica el estado abierto de la válvula y el otro el cerrado. Cuando ambos están abiertos esto puede indicar que la válvula está en tránsito, esto es pasando de estado abierto a cerrado o viceversa. Si ambos están cerrados es una situación de error.

Una señalización de alarma podría tratarse de un interruptor que indique que se ha excedido un nivel. En la lógica de alarmas es importante que la RTU sea capaz de distinguir la primer alarma de las siguientes que pudieran ser espurias (debido por ejemplo a un tableteo del interruptor de nivel).

Los módulos pueden ser de 8, 16, 32 entradas por placa. Se pueden instalar varios módulos si el requerimiento de entradas es mayor. Estas pueden ser normal abierto o normal cerrado. Se suele utilizar las del tipo normal cerrado para las señalizaciones de alarma.

La aislación óptica utilizando optoacopladores es una buena práctica para lidiar con picos de corriente inducidos en el cableado. Un circuito típico se muestra a continuación.

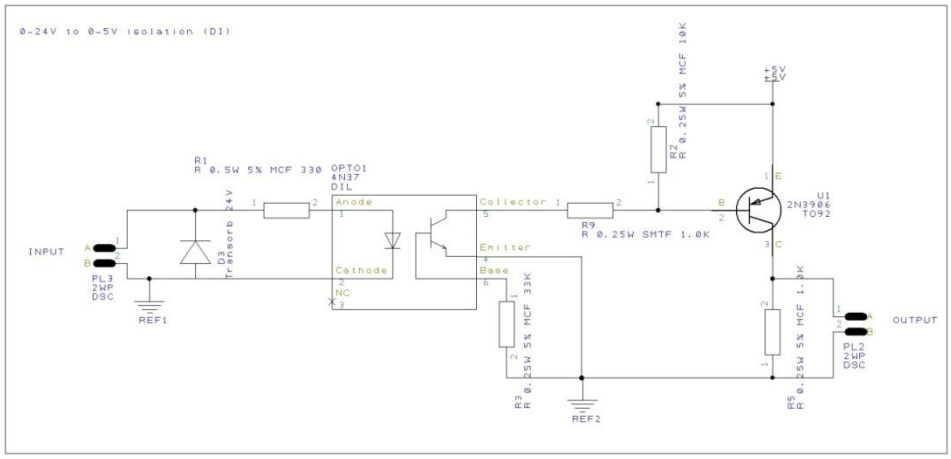


Figura 14 – Circuito típico de una Entrada Digital

Fuente: http://www.avrfreaks.net/forum/signals-isolation-di-0-24v-ai-0-10v-ai-4-20ma-0-5v

Existen dos enfoques principales de configuración del módulo de entrada:

- ✓ Modo "Sink": Los contactos son energizados externamente, cuando se produce el cierre de un contacto se energiza la entrada.
- ✓ Modo "Source": Se energiza el módulo, cuando se produce el cierre de un contacto conecta la entrada a tierra.

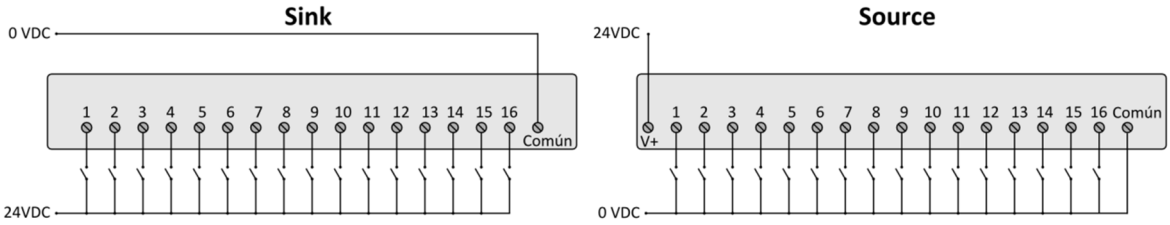


Figura 15 – Modos de conexión tipo "Sink" y "Source"

Especificaciones típicas de módulos de entradas digitales

- ✓ 16 entradas por módulo.
- ✓ LEDs de indicación de estado actual asociados a cada entrada.
- ✓ Tensión de entrada de activación: 110/240 VAC y 12, 24, 48 VDC.
- ✓ Aislación óptica para cada entrada.

2.3.3.5. Módulos de salidas digitales o discretas

Un módulo de salidas digitales maneja una tensión de salida para cada canal del mismo mediante el uso de alguno de los siguientes dispositivos:

- ✓ TRIAC: Usado para corriente alterna. Se utiliza típicamente un varistor para reducir el efecto de transitorios eléctricos.
- ✓ Relés.

- ✓ Salida TTL.

Especificaciones típicas de los módulos de salidas digitales

- ✓ Cantidad de salidas: 8, 16, 32, etc.
- ✓ Tensión de trabajo: 110/220 VAC, 24VDC. 0,5 a 2 A.
- ✓ LEDs de indicación de estado actual asociados a cada salida.
- ✓ Aislación óptica o contacto seco de relé para cada salida.

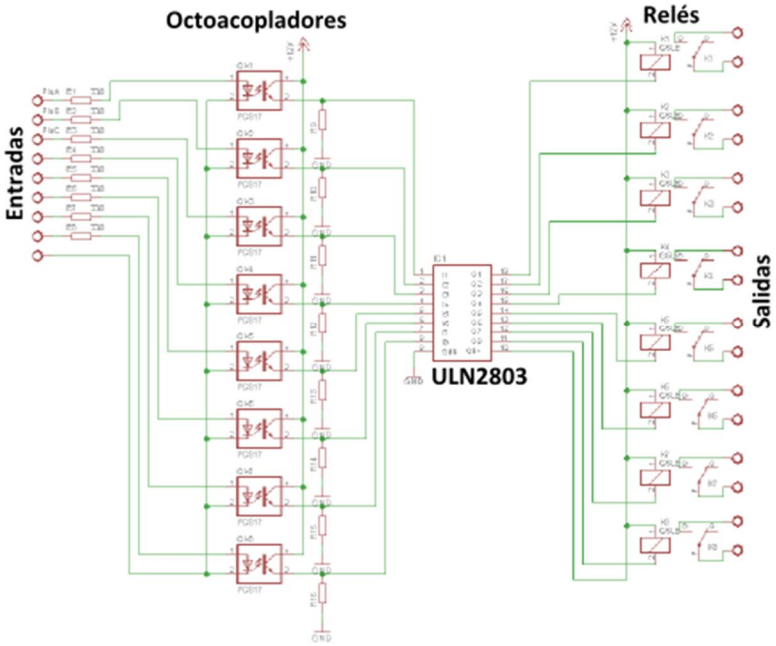


Figura 16 – Circuito típico de un módulo de Salidas Digitales

La **Figura 16** representa uno de los circuitos posibles que podría utilizarse como módulo de salidas digitales. El mismo posee optoacopladores para proveer aislación galvánica entre la parte lógica y la de potencia. A la salida de éstos se coloca un driver (en este caso se utiliza el circuito integrado ULN2803, aunque existen otras opciones) que permite manejar los bobinados de los relés de salida, los cuales ante su energización conectan o desconectan las cargas de salida.

2.3.3.6. Contador o acumulador

Hay muchas aplicaciones donde se requieren entradas de pulso. Las señales de entrada pulsantes son normalmente contactos secos en donde la energía es provista desde la fuente de la RTU en lugar de desde la fuente del pulso.

La Figura 17 muestra un diagrama de un contador digital. La aislación óptica es útil para minimizar los efectos del ruido externo y proveer aislación galvánica de la entrada. El tamaño del acumulador es importante considerando el número de pulsos a ser contados, de otra forma la cuenta ciclará a cero.

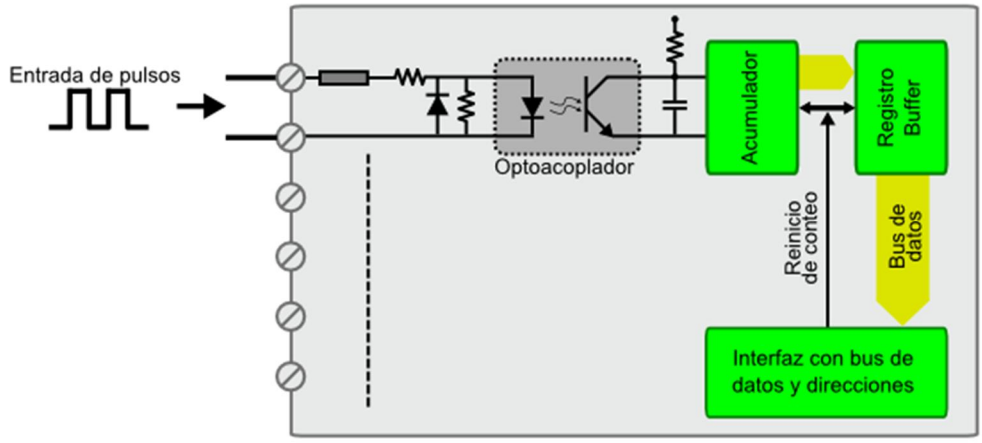


Figura 17 – Módulo contador

Eléctricamente el circuito es similar al de una entrada digital, pero a su salida se utiliza un acumulador de pulsos de entrada para contar los mismos.

Especificaciones típicas de los contadores

- ✓ Cantidad de contadores: 4, 8, etc.
- ✓ Tamaño de cada contador: Registros de 16, 32 bits.
- ✓ Frecuencia de conteo: Hasta 20 kHz.
- ✓ Ciclo de trabajo preferido: 50%.

2.3.3.7. Módulos de entradas y salidas digitales

Es común encontrar módulos que combinan en una misma placa entradas y salidas digitales. Los mismos son de los tipos ya descriptos. Una posible combinación podría ser:

- ✓ 4 entradas analógicas (resolución 8 bits).
- ✓ 2 entradas digitales.
- ✓ 1 salida digital.
- ✓ 2 salidas analógicas (resolución 8 bits).

2.3.3.8. Interfaces de comunicaciones

Las RTUs modernas deben ser lo suficientemente flexibles para manejar muchos medios de comunicación como puede ser:

- ✓ Interfaces seriales de cobre RS-232 / RS-422 / RS-485.
- ✓ Interfaces ópticas seriales.
- ✓ Ethernet sobre cobre / fibra óptica.

2.3.3.9. Fuente de alimentación

La RTU debe ser capaz de operar con tensiones de entrada tanto de corriente alterna (por ejemplo 110/240 VAC +/-10%) o de corriente continua (por ejemplo 12-96 VDC +/- 10%). Se suele proveer algún tipo de fuente ininterrumpida de energía (UPS) provisto de baterías que

permitan la operación ante cortes de suministro eléctrico por al menos 12 horas. Todo este equipamiento se suele instalar dentro del gabinete de la RTU. Se suelen enviar a la estación maestra datos relativos al estado de la batería, como su nivel de tensión, etc.

2.4. Sistemas de Transmisión de Datos

Se detallan a continuación las distintas arquitecturas o configuraciones de comunicaciones con las que se vinculan los sistemas SCADA ubicados en Centros de Control con sus propias RTUs, con otros Centros o con RTUs de otros Centros.

2.4.1. Rol de los Sistemas de Transmisión de Datos

Existen múltiples maneras de transmitir datos, pero debido a que el sistema de telecontrol debe operar en tiempo real, se imponen condiciones a los canales de comunicaciones. Los requerimientos típicos de la transmisión de datos en telecontrol son:

- ✓ Muy alta disponibilidad.
- ✓ Muy alta integridad de datos.
- ✓ Tiempos de transferencia cortos que permitan respuesta a los eventos en tiempo real.
- ✓ Alta eficiencia en la transferencia de información.

Además se debe tener en cuenta que los equipos de transmisión en ocasiones se ubican en las cercanías de infraestructura industrial por lo que muchas veces se encuentran sometidos a altos niveles de interferencia.

2.4.2. Tipos de configuraciones

Las configuraciones de redes de transmisión de datos pueden ser descompuestas en elementos con diferentes funcionalidades que constituyen la base de todas las configuraciones de red compuestas. Se representan a continuación los distintos esquemas de transferencia de datos apropiados para propósitos de telecontrol utilizando los siguientes símbolos:

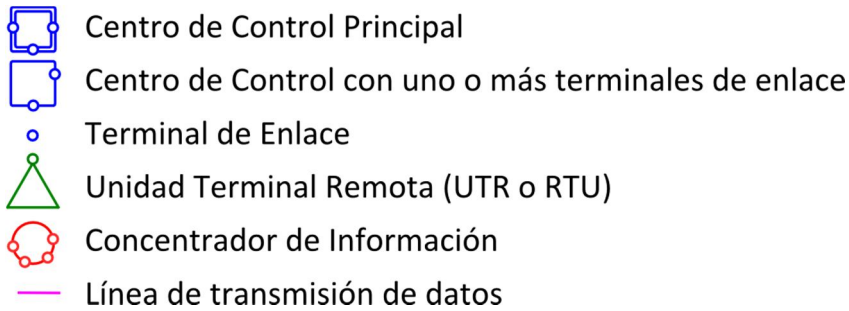


Figura 18 – Símbolos de esquemas de transferencia de datos

- ✓ **Configuración múltiple punto-a-punto:** el Centro de Control está conectado a las RTUs con un terminal de enlace por cada una de ellas. En cualquier instante todas las RTUs pueden transmitir datos hacia el Centro de Control, pudiendo hacerlo simultáneamente.

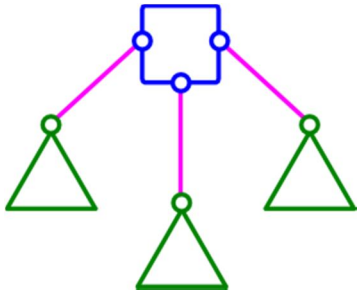


Figura 19 – Configuración múltiple punto a punto

- ✓ **Configuración estrella multipunto**: el Centro de Control está conectado a más de una RTU por un terminal de enlace común. En cualquier instante solo una RTU puede transmitir datos hacia el Centro de Control. Este puede transmitir mensajes a una o varias RTU simultáneamente.

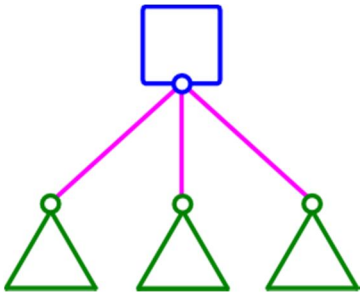


Figura 20 – Configuración estrella multipunto

- ✓ **Configuración multidrop:** el Centro de Control está conectado a más de una RTU a través de un camino común. Las restricciones impuestas al intercambio de datos son similares a la configuración estrella multipunto.

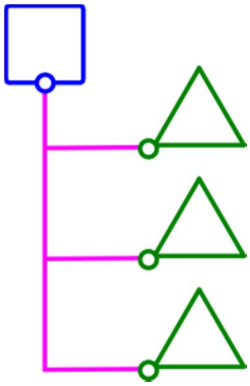


Figura 21 – Configuración multidrop

- ✓ **Configuración anillo multipunto**: el camino de comunicación entre todas las RTU forma un anillo. Este método mejora la disponibilidad del camino de comunicación. Si el camino se interrumpe en algún lugar, se mantiene la comunicación total ya que cada UTR puede dialogar con el Centro de Control por dos lados a través del anillo.

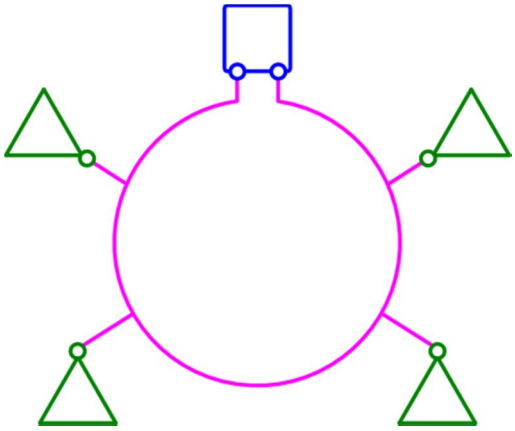


Figura 22 – Configuración anillo multipunto

- ✓ **Configuración compuesta**: Las configuraciones mencionadas anteriormente pueden ser combinadas de múltiples maneras. La más importante es la configuración mallada en la cual se establece comunicación entre cualquier par de estaciones. Pueden insertarse concentradores de información, los cuales recogen e intercambian datos entre los distintos puntos que conectan. Los sistemas multirouting (configuración anillo multipunto y configuración compuesta) tienen más de un camino para alcanzar a una determinada RTU e incrementan la confiabilidad y disponibilidad de la transmisión de datos.

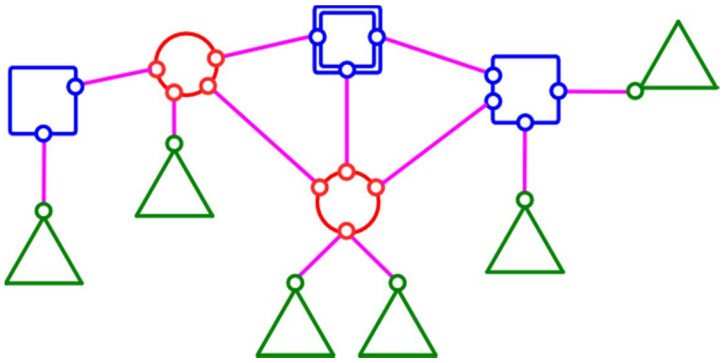


Figura 23 – Configuración compuesta

2.4.3. Modos de Transmisión

Hay tres modos básicos de iniciación de una transmisión de datos en telecontrol:

- ✓ **Transmisión iniciada por eventos o transmisión espontánea**:
  En este caso la transmisión se inicia cuando ocurre un evento (por ejemplo cambia el estado de un equipo, actúa una alarma, etc.). Este método es el que mejor se adapta a los requerimientos de tiempo real.
- ✓ **Transmisión a demanda**
  El Centro de Control requiere actualizar el estado de una cierta RTU y como respuesta esta envía la información actual de su estado.
- ✓ **Transmisión periódica**
  Este modo se utiliza frecuentemente para la transmisión de los valores medidos y de información binaria desde una RTU hacia el Centro de Control.

La transmisión iniciada por eventos y la transmisión a demanda se transmiten solo una vez por lo que es importante que tenga una alta integridad de datos. Para la transmisión periódica, una pérdida de información se corrige durante el requerimiento siguiente, por lo que se requiere una integridad de datos menor que los casos anteriores. Estos inicios de transmisión pueden combinarse de modo de obtener el sistema más apropiado.

2.4.4. Tecnologías de enlaces punto a punto

Una de las primeras tecnologías usadas como medio de comunicación consistía, y consiste, en la utilización de líneas fijas. Estos consisten en pares de cobre, propiedad de la empresa o alquilados a una telefónica, que recorre la distancia física que separa el centro de control de las RTUs.

En los extremos de estas líneas se instalan módems industriales del tipo serial pudiendo ser independientes o modulares. Estos últimos vienen en formato de placa y se enchufan en un rack apropiado. Sus velocidades son relativamente bajas, por ejemplo 1200, 2400, 4800, 9600 baudios por segundo. Sin embargo en los sistemas SCADA la cantidad de información que se transmite hacia y desde las RTUs es en general razonablemente pequeña, por lo tanto no tiene un gran requerimiento en cuanto a ancho de banda.

Figura 24 – Ejemplos de módems seriales utilizados para comunicación mediante pares de cobre

Fuentes:

http://faculty.mercer.edu/carter_k/resume.html

http://www.raymar-telenetics.com/analog-digital-modems/analog-modems/dial-leased-line-products/v32_v34/

http://www.arcelect.com/telenetics_dsp9612rm_modem_card_for_RM16M_racks.htm

http://www.asia.ru/ru/ProductInfo/63833.html

Por otro lado también es común encontrar vínculos de radio. Desde el punto de vista del sistema SCADA los radio-módems funcionan de forma similar a los módems para pares de cobre, es decir se conectan tradicionalmente mediante un puerto serie. Aunque cambia el medio de transmisión, la forma de modulación de este tipo de enlaces puede ser por ejemplo FSK.

Figura 25 – Ejemplos de equipamiento de radio usado para transporte de datos SCADA

Fuentes:

http://www.hamradio.com/detail.cfm?pid=H0-000125

http://www.motorolasolutions.com/en_xp/products/mototrbo.html

http://www.radioindustries.com.au/motorola-dr-3000.html

https://www.icom.co.jp/world/products/land_mobile/repeater/ic-fr5100/

La combinación de cobre y radio por tramos también es posible, especialmente para distancias lejanas donde es más complicado resolver el vínculo en un solo tramo.

Estos módems tenían antiguamente puertos seriales en general V.24 o RS232 del tipo DCE. Por otro lado el hardware sobre el que corrían los sistemas SCADA antiguos poseía una gran cantidad de puertos RS232 DTE necesarios para la conexión de estos equipos.

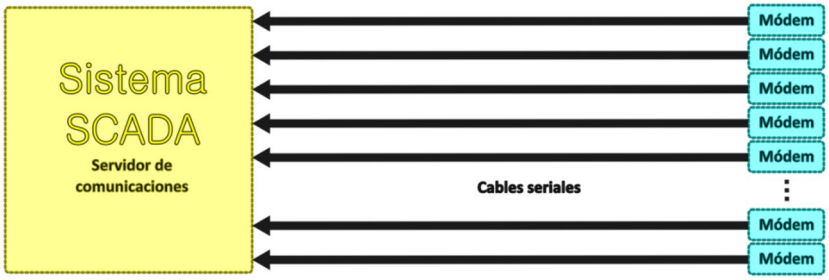


Figura 26 – Esquema de conexión tradicional de Módems a sistema SCADA

Actualmente los módems han evolucionado tecnológicamente. Si bien el medio de transmisión sigue siendo el mismo (cobre o radio) se han mejorado notablemente las características de los transmisores y receptores con nuevas y más complejas técnicas de modulación y demodulación que han permitido aumentar los anchos de banda y la confiabilidad.

Por su parte es común que incluyan como puerto de comunicación un puerto Ethernet en lugar de uno serial. Esto ha flexibilizado el esquema de conexión simplificando notablemente los cableados. Donde antes era necesario tender un cable de comunicaciones para cada módem ahora se puede instalar un Switch Ethernet concentrando una gran cantidad de equipos en el mismo, reduciendo la cantidad de vínculos necesarios a uno o a lo sumo dos si se desea redundancia.

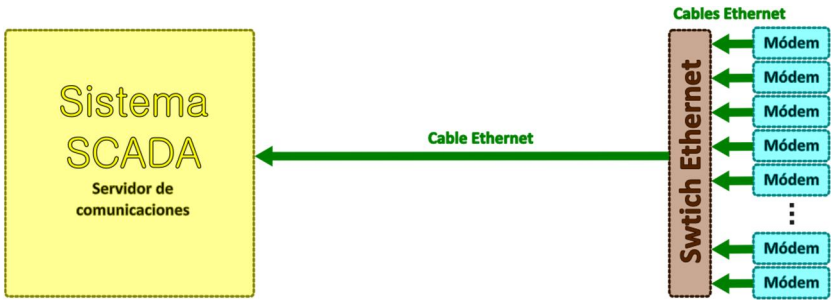


Figura 27 – Conexión de Módems con puerto Ethernet a sistema SCADA

En los casos en que se sigue manteniendo equipamiento de comunicaciones viejo conectado a sistemas SCADA más moderno se suele utilizar un tipo de dispositivo conocido como Terminal Server. Este dispositivo posee una gran cantidad de puertos seriales y un puerto Ethernet para su conexión al sistema SCADA. Luego en el servidor de comunicaciones se generan puertos serie virtuales que representan a los del Terminal Server. De forma lógica el sistema funciona de igual manera, aunque físicamente el esquema es mucho más flexible y reducido en tamaño.

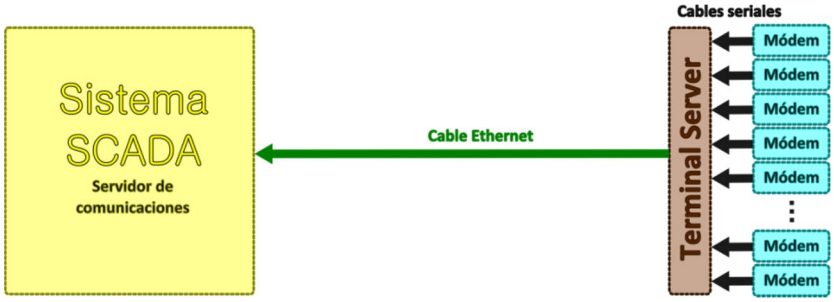


Figura 28 – Conexión de Módems seriales a sistema SCADA mediante Terminal Servers

Otras opciones de comunicación punto a punto son los vínculos satelitales o de fibra óptica. En cualquier caso existe un dispositivo en ambos extremos que presenta un puerto de conexión (serie o Ethernet) con el que sistema SCADA pueda comunicarse.

Este tipo de esquema de comunicaciones plantea un sistema cerrado o auto-contenido formado por el SCADA, las comunicaciones y los dispositivos de control.

2.4.5. Tecnologías de redes de área amplia (WAN)

Otra de las infraestructuras de comunicaciones usadas tradicionalmente por los sistemas SCADA han sido los servicios de redes de transporte WAN ofrecidos por las empresas de telecomunicaciones.

2.4.5.1. Redes X.25

Las redes X.25 fueron quizás las primeras redes transporte, basada en la tecnología de conmutación de paquetes, utilizadas para el envío de datos SCADA entre otras aplicaciones. El protocolo fue definido por el CCITT (actualmente ITU) antes del modelo OSI, aunque sus tres

capas equivalen a las tres capas inferiores de este modelo. El primer borrador se emitió en el año 1974, y se editaron revisiones en 1976, 1978, 1980 y 1984.

Alcanzó su mayor popularidad a fines de la década del setenta y durante la del ochenta. Durante la década del noventa comenzó a ser reemplazado paulatinamente por Frame Relay.

Define la interfaz entre un DTE (Data Terminal Equipment o Equipo Terminal de Datos) y un DCE (Data Communication Equipment o Equipo de Comunicaciones de Datos) para el acceso a redes públicas de conmutación de paquetes. El DCE actúa como un nodo de acceso a la red.

Los paquetes son de duración variable y ocupan el canal hasta que la transmisión termine. X.25 no define la arquitectura interna ni de operación de la red pública ni define algoritmos de encaminamiento, lo que queda a criterio del proveedor pero que debe ser transparente al usuario. La norma define mecanismos de verificación de las tramas de información muy robustos y apropiados para la baja calidad de los medios de transmisión reinantes en la época en que este protocolo fue popular.

El modelo X.25 se basaba en establecer circuitos virtuales confiables a través de redes públicas compartidas, por lo que conceptualmente se asimilaba a las redes de telefonía tradicionales. Los circuitos virtuales se establecen desde nodos DCE fuente a nodos DCE destino llevando un registro del flujo de paquetes en cada conexión. Varios circuitos virtuales pueden operar simultáneamente a través de una línea de transmisión mediante la técnica de multiplexado estadístico por distribución de tiempo. Se denominan circuitos virtuales porque emulan la existencia de una conexión física permanente entre la fuente y el destino aunque este no es el caso por tratarse de una red de conmutación de paquetes. Los circuitos pueden ser conmutados (SVC), cuyos paquetes siguen diferentes trayectorias a través de la red, y permanentes (PVC) cuyos paquetes siguen siempre la misma ruta.

En el esquema de la **Figura 29** podemos ver un ejemplo de comunicación entre centros y RTUs mediante una red de conmutación de paquetes con acceso X.25.

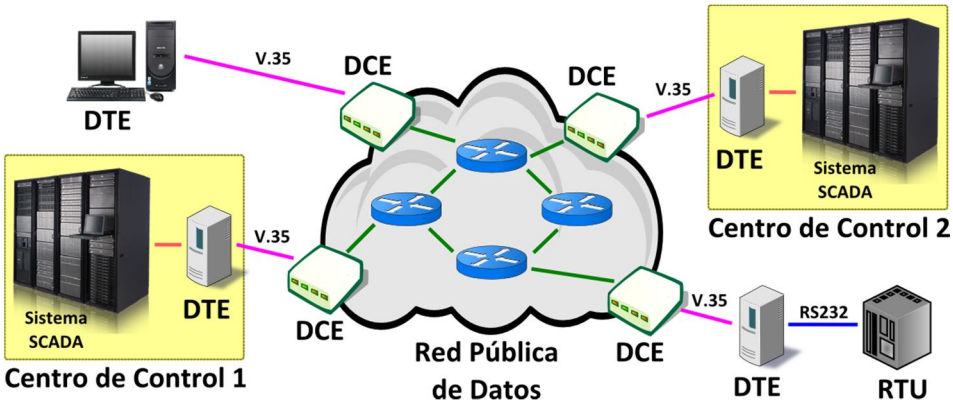


Figura 29 – Esquema de transporte SCADA sobre red X.25

El protocolo se compone de una capa física, una capa de link y una capa de paquetes. La capa física especifica la interfaz de comunicaciones, estableciendo características eléctricas, mecánicas, funcionales y de procedimiento, puede ser X.21 o V.35 con velocidades de hasta

19,2kbps y 48kbps respectivamente. La capa de link se implementaba mediante el protocolo LAPB (Link Access Procedure Balanced) que es un subconjunto del protocolo HDLC (High Level Data Link Control), el mismo es orientado a bit, asegura la secuencia de los paquetes y controla errores.

Si bien el equipamiento relacionado con este protocolo ha sido dado de baja por las empresas de telecomunicaciones debido a su antigüedad, el servicio se sigue ofreciendo para las organizaciones que lo requieran. Es así que se han desarrollado protocolos especiales como XOT. El mismo fue creado por Cisco para el encapsulamiento de paquetes X.25 y su transporte por redes TCP/IP reemplazando al protocolo LAPB.

Esta red ha sido históricamente usada por los Sistemas de Telecontrol para el transporte de datos entre centros y también en algunos casos para la comunicación con RTUs.

2.4.5.2. Frame Relay

Este protocolo reemplazó a X.25 durante la década del noventa permitiendo mayor capacidad de transporte, se lo considera su sucesor. En la medida en que los medios de transmisión evolucionaron con los vínculos de fibra óptica, fue posible transmitir paquetes de datos de forma eficiente y confiable, de esta forma ya no fue necesaria la robusta verificación de integridad de tramas que poseía el protocolo X.25.

En X.25 se lleva a cabo un control de flujo y control de errores usando números de secuencia y ventanas deslizantes, este no es el caso de Frame Relay que deja esta tarea a cargo del equipamiento que se conecte a la red. Sin embargo sí se realiza la verificación de trama como en X.25, aunque si ésta contiene un error no se pide retransmisión automática sino que simplemente se la descarta. Esto simplifica notablemente el protocolo y reduce el retardo de transmisión.

Frame Relay es entonces un protocolo simplificado para el transporte de información de alta velocidad, posee sólo dos capas equivalentes a los dos primeros niveles del modelo OSI. Al eliminar el nivel de red (nivel tres) los protocolos que funcionan en éste se transfieren a través de Frame Relay de forma transparente, aumentando la velocidad que puede ir de 9.6Mbps hasta 52Mbps. Se concentra en la entrega rápida, en el orden y el lugar correcto de los datos, descartando los incorrectos. Elimina la mayor parte de la complejidad característica de otros protocolos como HDLC.

Se maneja al igual que X.25 con el concepto de circuitos virtuales pudiendo ser estos permanentes (PVC) o temporales (SVC). También al igual que en X.25 los paquetes son de longitud variable y el canal es ocupado hasta que se termine la transmisión. Asimismo contempla un sistema de administración de la congestión a cargo de la red y de las estaciones de usuario las que pueden limitar cuando es necesario el flujo de datos hacia la red.

Al igual que X.25, Frame Relay es utilizado para el transporte de datos SCADA en configuraciones similares a las detalladas en el punto anterior.

2.4.5.3. ATM

Las redes ATM (Asynchronous Transfer Mode o Modo de Transferencia Asíncrono) son las sucesoras de la tecnología Frame Relay. Es una tecnología de conmutación de banda ancha orientada a conexión y con capacidad de multiplexado.

Es de alguna manera similar a la conmutación de paquetes usada por X.25 y Frame Relay. Permite que muchas conexiones lógicas sean multiplexadas sobre una única interfaz física. A diferencia de X.25 y Frame Relay que enviaban paquetes de longitud variable, la tecnología ATM se rige por la transmisión de celdas con una longitud fija de 53 bytes. Es apropiada para la transmisión de aplicaciones sincrónicas y asincrónicas con distintos grados de calidad y servicios. Tiene capacidades mínimas de control de error y flujo lo que reduce su overhead, el header de cada celda es de solo 5 bytes, restando 48 para payload.

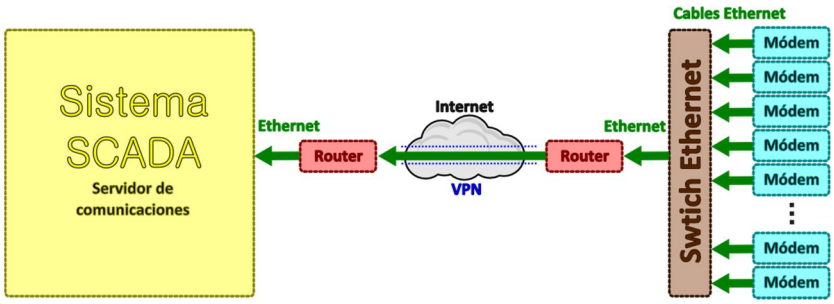
Al usar celdas de tamaño fijo hace más simple el procesamiento que requiere cada nodo de la red, reduciendo la latencia, aumentando la velocidad y haciéndola apropiada para altas tasas de transmisión de datos.

Si bien la capacidad y el ancho de banda de las redes ATM exceden ampliamente los requerimientos para transmisión de datos SCADA muchas empresas han utilizado esta tecnología para comunicar sus equipamientos de telecontrol a veces revendiendo la capacidad excedente para recuperar parte de la inversión.

2.4.6. MPLS, Internet y redes móviles

Con la aparición de redes de transporte más modernas, se ha modificado la arquitectura de comunicaciones de los sistemas SCADA. El crecimiento y la popularización de Internet y de las redes móviles han facilitado en gran medida la conexión entre los dispositivos de adquisición de datos lejanos como las RTUs y los centros de control. Hoy en día es muy simple contar con un acceso a la web en infraestructuras lejanas mediante proveedores de servicio de Internet o GPRS.

Así mediante la implementación de redes privadas virtuales (VPN) es posible encaminar las comunicaciones SCADA por este medio de forma segura. Esto puede llevarse a cabo incluso con equipamiento de comunicaciones antiguo realizando las adaptaciones correspondientes.



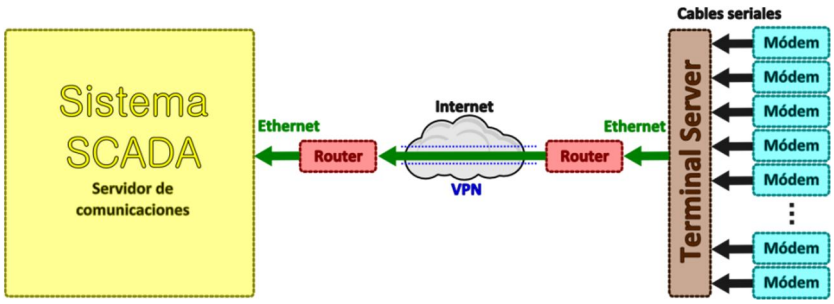


Figura 30 – Encaminamiento de comunicaciones SCADA a través de Internet

De esta forma logramos una gran flexibilidad, la conexión de sitios mediante VPNs puede realizarse a través conexiones tradicionales y sin calidad de servicio (entrega de mejor esfuerzo) u optar por redes de transporte modernas de tecnología MPLS (Multiprotocol Label Switching) que aseguran calidad de servicio. Esto debe decidirse en base a la criticidad operativa de la infraestructura o instalación que se desea supervisar.

MPLS es una tecnología de transporte de datos que está reemplazando paulatinamente a las redes ATM y Frame Relay. Se definió en la publicación RFC3031 de la IETF. Las redes MPLS proporcionan un transporte escalable e independiente del protocolo, trabaja agregando a los paquetes un encabezado que contiene una o más labels o etiquetas en una capa que se ubica entre los niveles dos y tres del modelo OSI. Luego los paquetes se encaminan entre los nodos de la red a través de diferentes rutas en base exclusivamente a estas etiquetas sin necesidad de procesamiento de direcciones de red extensas o de tablas de ruteo. Las redes MPLS pueden ser usadas para transportar tráfico de muchos tipos, ya sean paquetes IP así como también tramas ATM o Ethernet. También son apropiadas para transportar VPNs SCADA.

Otro fenómeno a considerar es la masificación a todo nivel de Internet. Esto hace que hoy por hoy se hable de la "Internet de las Cosas" (Internet of Things) concepto que trata de explicar el fenómeno que se manifiesta en el surgimiento constante de dispositivos electrónicos de todo tipo que vienen de fábrica listos para ser conectados a la web. Esto amplía el panorama de los sistemas SCADA ya que se ha pasado de un modelo muy cerrado, con redes de comunicaciones propias y pocas infraestructuras desde donde se tomaba información, a uno mucho más abierto en donde tenemos gran cantidad de posibilidades de comunicación más económicas junto a una masa crítica de dispositivos electrónicos de todo tipo ya preparados y diseñados para ser supervisados desde sitios remotos. La inteligencia que antes se concentraba fuertemente en un Centro de Control en la figura del SCADA y del hardware sobre el que se soportaba, se ha distribuido a lo largo de toda la arquitectura del sistema.

Actualmente un sistema SCADA debería estar preparado para poder funcionar como parte integrante de un sistema superior y con claras posibilidades de interoperación con las otras partes. Ahora sería posible aprovechar infraestructuras de comunicación y control existentes, incluso de otras empresas o personas, para montar fácilmente una estructura de supervisión poderosa y empleando mínimos recursos.

2.5. Generaciones de sistemas SCADA

Desde el punto de vista de la evolución histórica de los sistemas SCADA podemos clasificar los mismos en tres generaciones claramente identificables.

2.5.1. Sistemas SCADA de primera generación

Los sistemas SCADA de primera generación son los más antiguos en la línea de tiempo y por lo tanto los de tecnología más primitiva. Su aparición se remonta a fines de la década del sesenta y principios de la de los setenta.

Estos sistemas contaban con características muy reducidas de procesamiento y memoria y por lo tanto sus capacidades eran acotadas. Sus medios de almacenamiento eran o bien unidades de cinta magnética o discos duros de enormes dimensiones y unos pocos megabytes de almacenamiento. Por su parte su memoria podía ser tan voluminosa como 512 kBytes.

La carga de programas se realizaba mediante tarjetas perforadas, la interfaz de usuario era en general una terminal teletipo. Su estructura interna se componía de cientos de chips con compuertas lógicas interconectados entre sí mediante cableados tipo “wire wrap”. La enorme cantidad de chips de tecnología TTL generaban una gran cantidad de calor que muchas veces no llegaba a ser disipado eficientemente, haciendo el sistema vulnerable a fallas frecuentes de sus componentes.

Las comunicaciones con las RTUs (también de tecnología similar) se realizaban mediante protocolos y puertos de comunicación propietarios. Se utilizaban pares de cobre, propios o alquilados a empresas de telefonía y vínculos de radio.

Se puede decir que se trataba de sistemas monolíticos dado que se construían como un solo gran módulo donde se concentraban todas las funciones típicas de este tipo de sistemas. Los paneles del sistema eran colocados juntos, en un solo espacio físico y no podían ser separados demasiado debido a la escasa inmunidad al ruido que poseían los buses de comunicaciones internos.

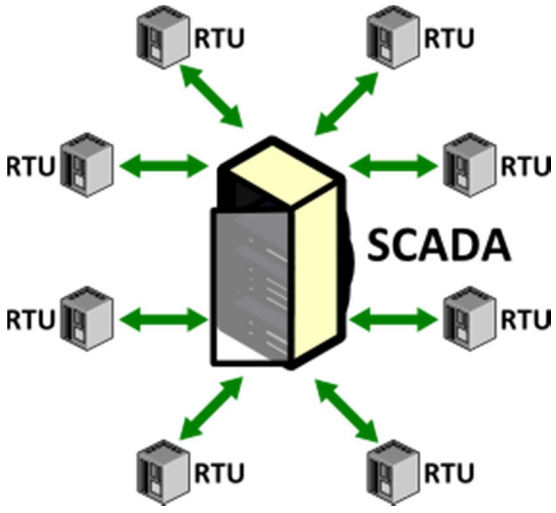


Figura 31 – Esquema de un sistema SCADA de primera generación

Podemos citar como ejemplo de la tecnología utilizada en esta generación de sistemas a Modcomp, una empresa estadounidense fabricante de minicomputadoras en la década del 70 creadas para ser empleadas en aplicaciones de tiempo real por lo que fueron usadas para algunos de los primeros sistemas SCADA.

Estas minicomputadoras también fueron utilizadas para otras interesantes aplicaciones, por ejemplo durante la fase final del programa espacial Apolo y durante las primeras etapas del programa del trasbordador espacial de la NASA.

En Argentina este computador fue parte del primer sistema SCADA del Centro de Movimiento de Energía (CME), centro de control de la red de transmisión y subtransmisión eléctrica de la empresa Segba (Servicios eléctricos del Gran Buenos Aires), empresa que luego fuera privatizada durante la década del noventa.

Allí estuvo en operación desde fines de la década del setenta hasta principios de los noventa de forma ininterrumpida. Las imágenes que se presentan a continuación corresponden a este computador instalado en el CME. Son puntualmente del modelo MODCOMP VI, hoy se encuentra en desuso pero aún se preservan algunas partes con fines históricos.

El MODCOMP IV era un computador de 32 bits con 512 kBytes de memoria, 240 registros de propósito general, unidad de coma flotante y una unidad de gestión de memoria con 1024 registros organizados en 4 páginas de 256 registros cada una. Utilizaba un sistema operativo propietario especialmente diseñado para este computador y que no podía ser usado en otras plataformas.

El sistema se componía de este computador y periféricos y accesorios como lectoras de cintas, discos rígidos, consolas de operación, interfaces de comunicaciones, impresoras de alarmas, teletipos, etc. Para este tipo de aplicaciones Modcomp estaba asociada con TRW Controls, responsables de la integración del sistema de Segba.

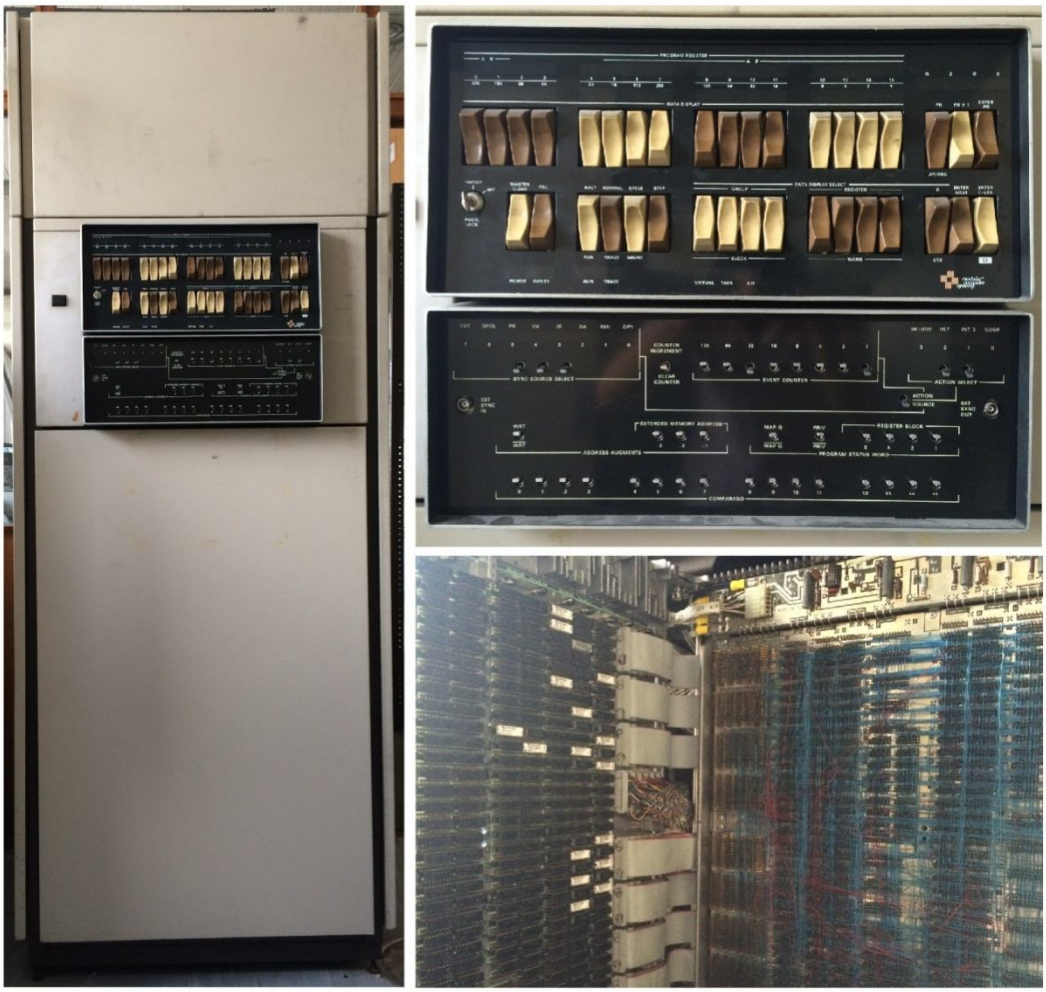


Figura 32 – Imágenes del minicomputador MODCOMP IV perteneciente al CME (Segba)

En la imagen puede verse el panel donde se ubicaba el computador y las placas de circuitos integrados cuyos terminales se interconectaban mediante la tecnología “wire-wrap” que se ubican en la parte inferior.

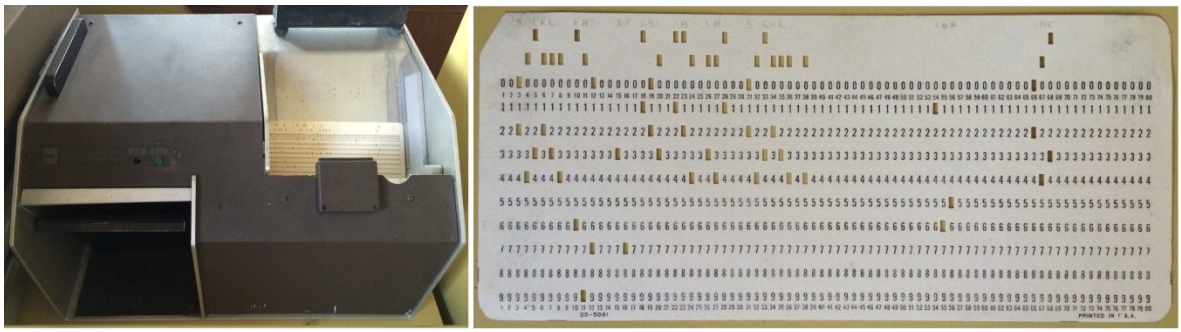


Figura 33 – Equipo de lectura de tarjetas perforadas

Las tarjetas perforadas eran la forma de cargar programas o aplicaciones en el sistema, existía un lector especial que iba leyendo las tarjetas secuencialmente hasta cargar todo el código.

Figura 34 – Disco rígido con capacidad para almacenar 25MB

El almacenamiento de datos se realizaba o bien en unidades de cinta magnética o también en un disco rígido con capacidad de 25MB, que consistía en varios discos apilados como el que puede verse en la imagen. Se disponían de varias de estas unidades las cuales podían ser intercambiadas dentro del tambor que las alojaba.

Figura 35 – Consola de operación y teletipo

La comunicación con las RTUs se llevaba a cabo mediante un protocolo de telecontrol de TRW conocido como TRW 9550. Era un protocolo básico, orientado a bit lo que significa que no se usaban caracteres o bytes (paquetes con longitud múltiplo de 8 bits) sino que se transmitían secuencias de bits de longitud arbitraria. Existían módems de comunicaciones especiales para este tipo de transmisión que no cumplían con la norma EIA232 por ser anteriores a esta.

La interacción con el usuario era para el caso de los operadores del sistema a través de consolas de operación con una interfaz gráfica rudimentaria en donde podían verse los

esquemas unifilares de las subestaciones eléctricas (imagen izquierda). Por su parte el personal de sistemas también las utilizaba con las herramientas de programación y configuración y para la creación o modificación de pantallas del sistema.

También existían teletipos (imagen derecha) que imprimían en papel carbónico mensajes de salida del sistema como alarmas y otras informaciones.

Figura 36 – Personal de Segba en Florida, EEUU junto a ingenieros de MODCOMP durante los ensayos de prueba del sistema en la década del 70

2.5.2. Sistemas SCADA de segunda generación

La segunda generación de sistemas SCADA surge a fines de la década del ochenta y principios de la del noventa. La evolución desde la generación anterior determinó que poco a poco se empezara a favorecer una estructura más distribuida, modular con partes identificables por sus funciones, en detrimento de la estructura monolítica que presentaban los sistemas anteriores.

El sistema se compone de una cierta cantidad de servidores o computadores, con una potencia computacional mucho mayor respecto a la generación anterior, que se comunican entre sí mediante redes de área local (LAN). Cada servidor tiene una función específica, por ejemplo comunicaciones, base de datos, aplicaciones, consolas de operación, etc.

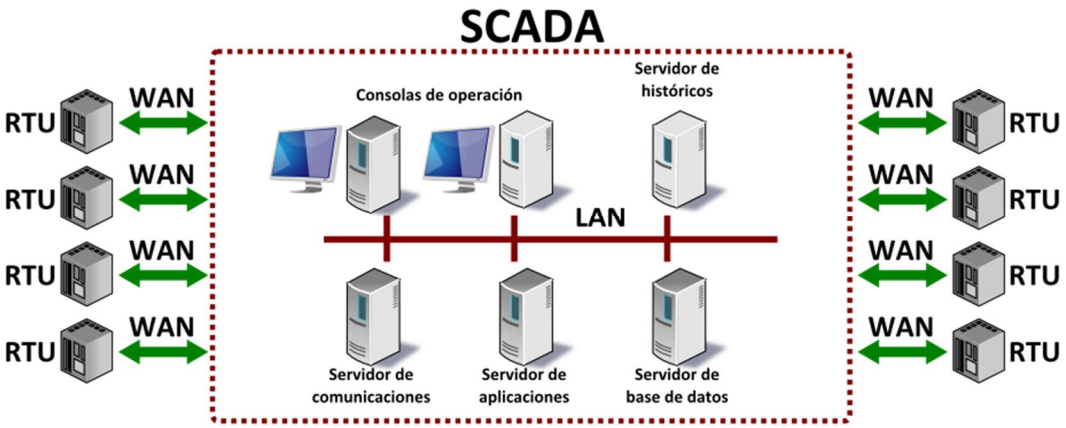


Figura 37 – Sistema SCADA de segunda generación

Ahora los módulos del sistema se encuentran claramente diferenciados, pueden estar separados por mayores distancias, por su parte las RTUs también evolucionan permitiendo conexiones de datos más estables. Surgen sistemas operativos, protocolos y tecnologías abiertas, lo que permite que por primera vez comience a ser posible la interoperabilidad e integración parcial entre sistemas de distintas empresas.

Las comunicaciones con las RTUs son del tipo WAN utilizando redes de transporte como ATM, X.25 o Frame Relay, es decir redes de conmutación orientadas a conexión con escasa flexibilidad. También se siguen usando vínculos punto a punto sobre pares de cobre propios o alquilados y radioenlaces.

A modo de ejemplo se exponen a continuación algunas imágenes del segundo sistema SCADA que en operación en el Centro de Movimiento de Energía desde mediados de la década del noventa hasta 2011.

Figura 38 – Vista general de los paneles del sistema

Este sistema era integrado por Siemens y su denominación era EMPOWER Spectrum.

Sus servidores eran IBM de la familia RS/6000, con procesadores de unos 120MHz de frecuencia de reloj, en promedio unos 128MB de memoria RAM y uno o más discos rígidos de 2GB de almacenamiento. Fue la primera línea de equipos con procesadores PowerPC, utilizaban el sistema operativo AIX, basado en UNIX. Sobre este sistema operativo corrían las aplicaciones de adquisición de datos, almacenamiento de datos históricos, comunicaciones con RTUs, estadísticas desarrolladas por Siemens y que conformaban el sistema SCADA.

La comunicación entre servidores se realizaba mediante una red LAN de 100Mbps, a través de la misma las aplicaciones residentes en los servidores intercambian información.

El servidor de comunicaciones contaba con una interfaz propietaria que consistía en un conjunto de placas idénticas basadas en el microprocesador Zilog Z80 y con dos puertos seriales tipo RS232 DTE cada una.

A partir de este sistema el protocolo de telecontrol utilizado para comunicarse con las remotas fue DNP3 (Distributed Network Protocol). El mismo fue creado por la empresa Westronic, luego Harris, ahora propiedad de General Electric. Es orientado a byte, a diferencia del TRW9550 del sistema anterior, robusto y con muchas más funcionalidades. Cumple parcialmente con el modelo OSI implementado en tres capas: enlace (Data Link), aplicación y una tercera pseudo-capa de transporte. Utiliza CRC para la detección de errores. Se utiliza fundamentalmente en el sector eléctrico aunque también es usado en las áreas de petróleo y

gas. No fueron considerados aspectos de seguridad en su diseño, lo que lo hace vulnerable para su transporte sin medidas de seguridad a través de redes públicas.

Las placas se enchufan a un backplane de conexión que se conecta a su vez por un lado al servidor de comunicaciones mediante una interfaz propietaria y por el otro a un panel de conectores DB-25 donde mediante cables seriales se llega a los Módems utilizados para comunicarse con las RTUs.

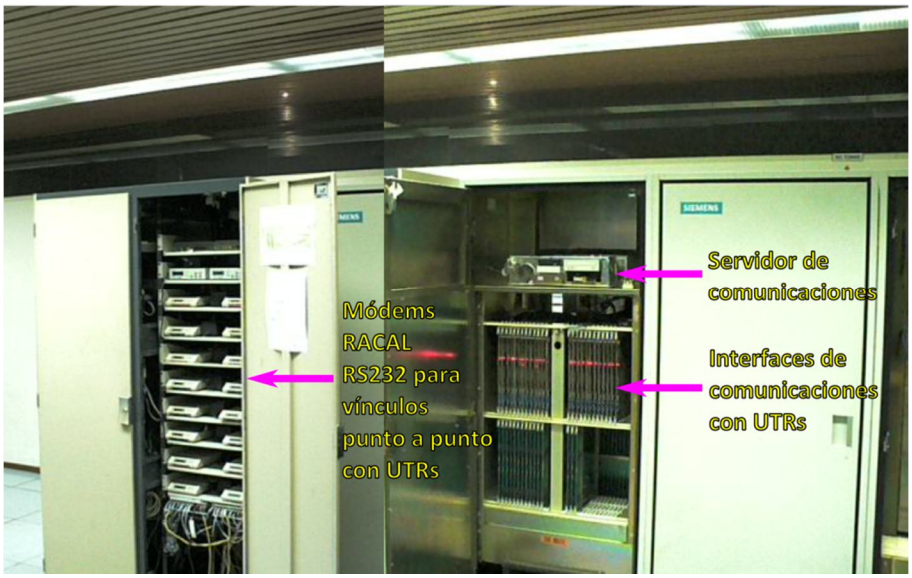


Figura 39 – Vista de servidor, interfaces y módems de comunicaciones con RTUs

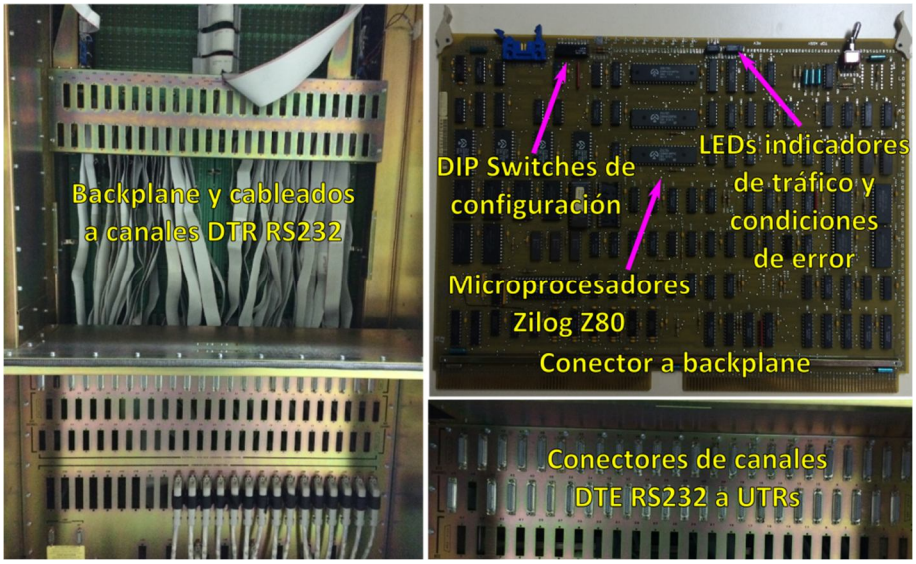


Figura 40 – Vista trasera de interfaces de comunicaciones y detalle de placa con microprocesadores Zilog Z80

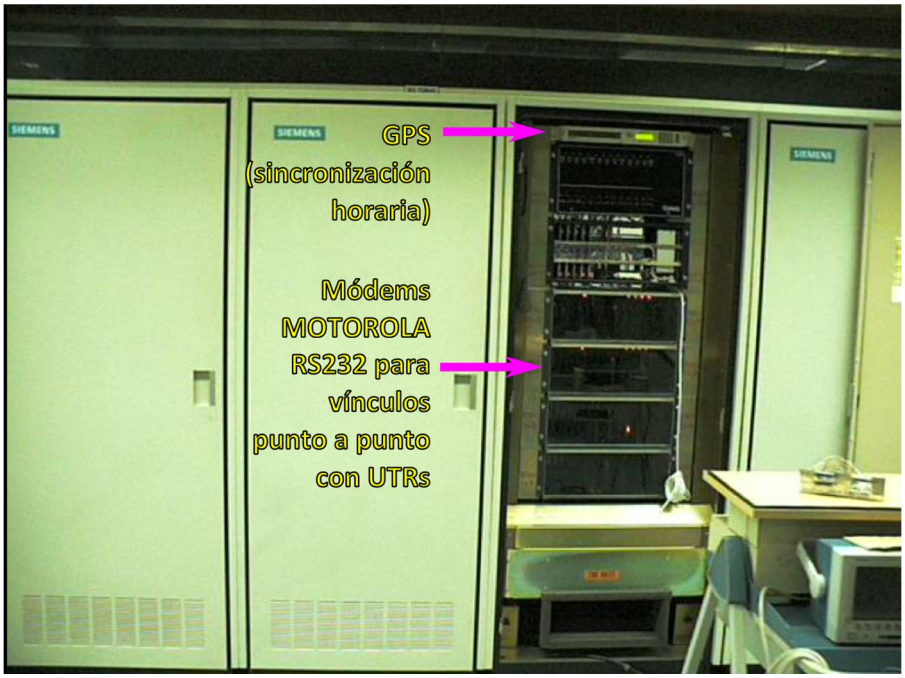


Figura 41 – Otros módems de comunicaciones con RTUs y GPS de sincronización horaria

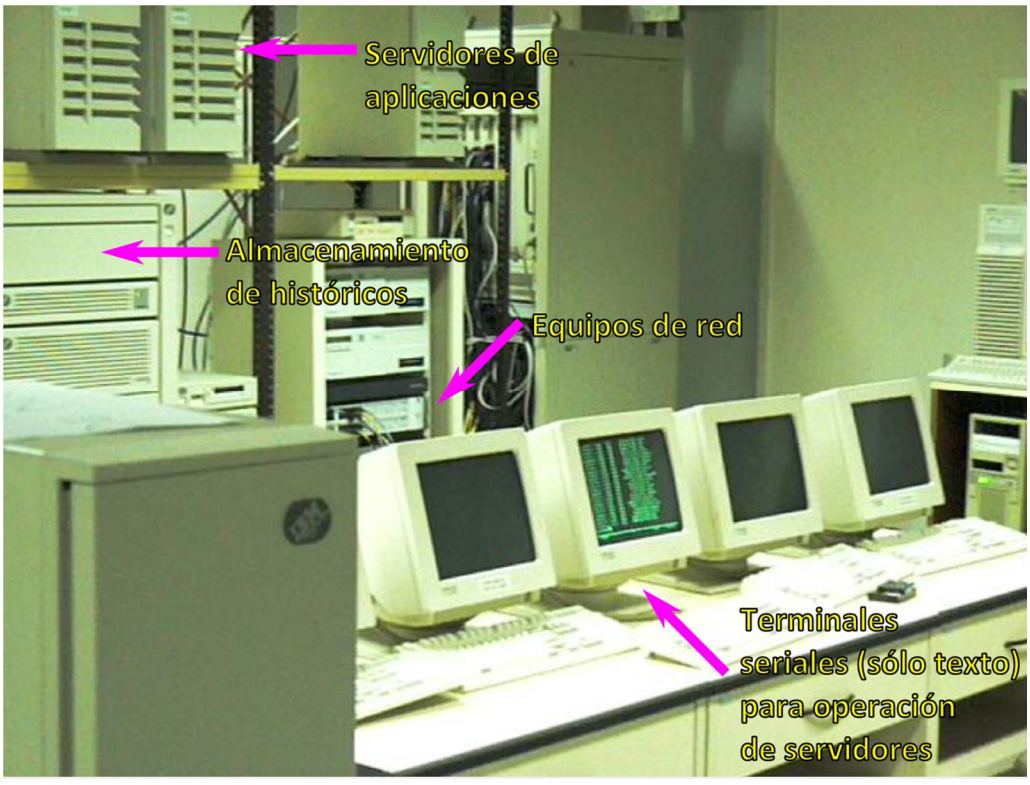


Figura 42 – Consolas de operación de servidores y otros servidores y equipamientos

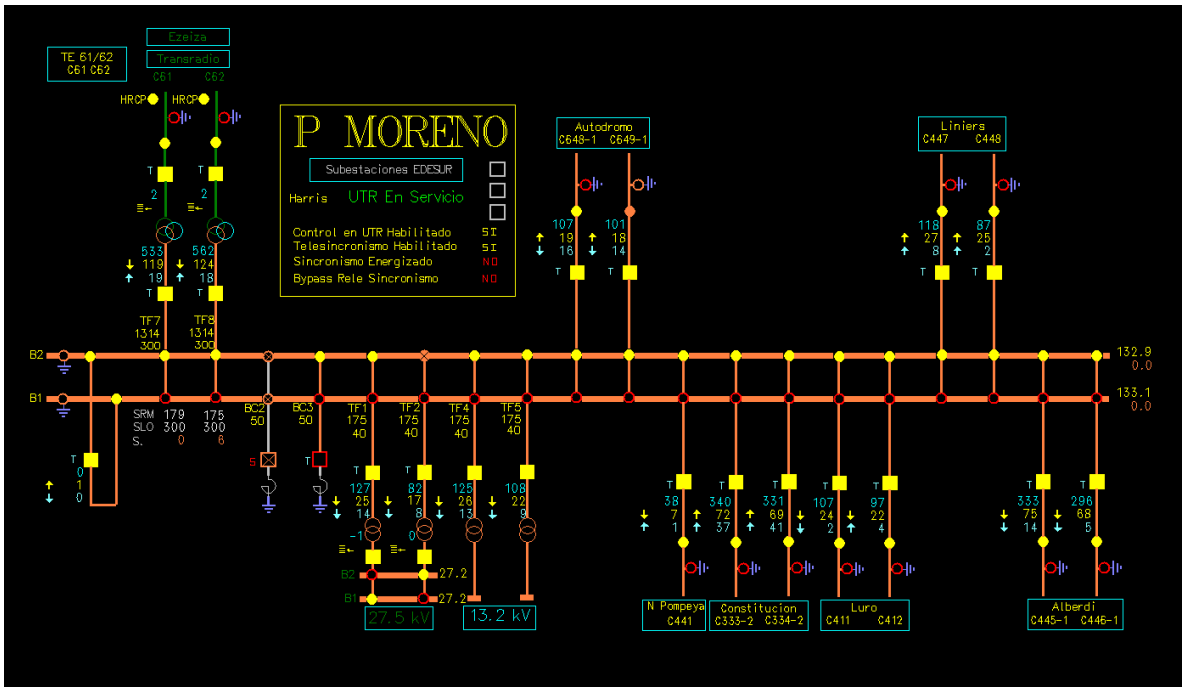


Figura 43 – Ejemplo de una pantalla del sistema representando el esquema unifilar de una subestación

2.5.3. Sistemas SCADA de tercera generación

La tercera generación de sistemas SCADA permite un esquema en donde los elementos se encuentran aún más distribuidos que en la generación anterior. Esta generación comienza a ser posible con la aparición de Internet y de redes de transporte de tecnología más moderna como MPLS. Con esta filosofía queda aún más lejos el concepto de los sistemas más antiguos donde la inteligencia se concentraba en un sola locación física, en un centro de control y en un SCADA monolítico para pasar a una concepción en donde los módulos del sistema ya no necesitan estar juntos.

Así por ejemplo podríamos tener un centro de control maestro o principal, donde todas las funciones típicas de un sistema SCADA están representadas, pero al mismo tiempo dar la posibilidad a los usuarios de conectarse desde consolas de operación remotas, tener servidores de almacenamiento de datos en la nube, u otros servicios remotos. Esta arquitectura facilita incluso tener centros de control de respaldo o sistemas SCADA de backup en ubicaciones lejanas siempre listos para operar en caso de falla del SCADA maestro.

Los elementos del sistema ya no están vinculados con conexiones rígidas sino que las mismas cuentan con la flexibilidad necesaria para ubicarlos donde sea más cómodo atendiendo a otros factores importantes como el punto de vista operativo o de seguridad.

En cuanto a las comunicaciones con las RTUs es cada vez menos común tener vínculos mediante pares de cobre o radio excepto que estos sean a sitios inaccesibles donde no exista otra tecnología de comunicación disponible. Se pondera, en cambio, el uso de redes de transporte modernas. Para aplicaciones en las que no se requiera calidad de servicio se pueden utilizar incluso enlaces a Internet convencionales.

Por su parte es posible aún comunicarse con RTUs de tecnología vieja mediante el uso de equipamiento especial que encamina las conexiones seriales de las mismas a través de sockets TCP/IP.

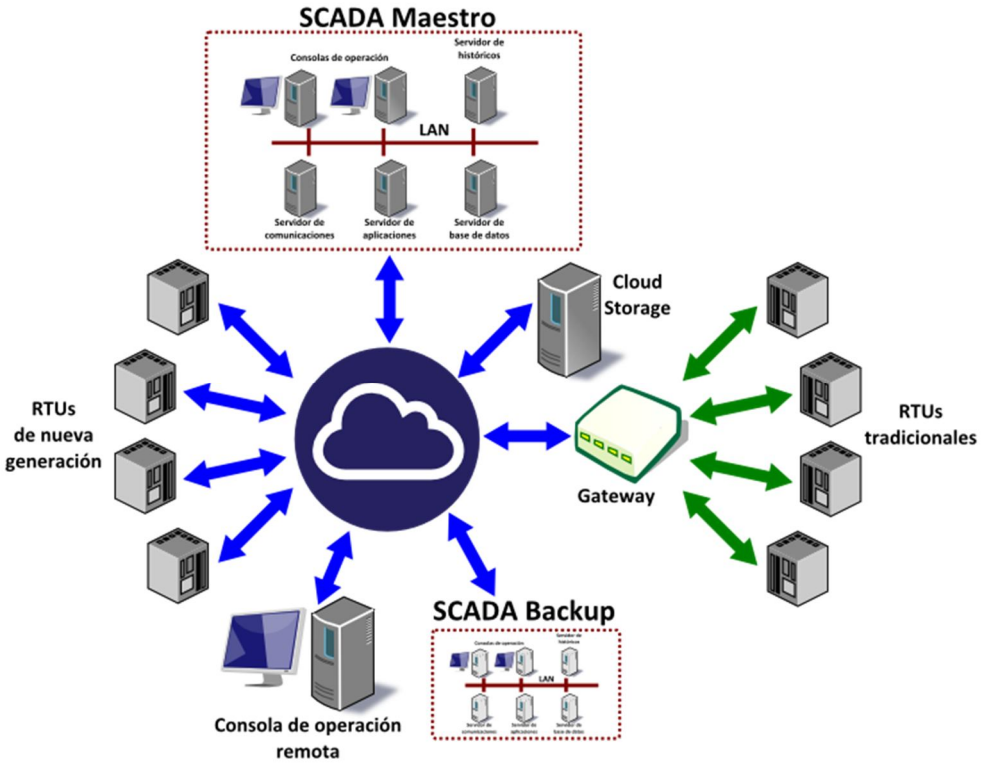


Figura 44 – Sistema SCADA de tercera generación

El sistema SCADA en operación actualmente en Centro de Movimiento de Energía de Buenos Aires se encuentra dentro de esta categoría.

Se trata de un sistema integrado por la empresa estadounidense General Electric denominado XA/21.El mismo es un sistema de control distribuido, de alta performance, especialmente diseñado para su aplicación en la supervisión de redes eléctricas. Su arquitectura soporta la expansión incremental de los nodos del sistema y su funcionalidad. Nuevos canales de comunicaciones para RTUs o consolas de operación pueden ser agregadas fácilmente según se requiera.

Su interfaz de usuario es basada en Web, posee un sistema de la alarmas de tiempo real con diferentes prioridades, filtrados, cálculos en tiempo real, definición de áreas de responsabilidad, etc.

Posee alta estabilidad y disponibilidad con redundancia de los nodos críticos. Es modular a nivel software y hardware, con capacidad potencial de manejar hasta más de un millón de puntos.

Figura 45 – Vista de paneles del sistema General Electric XA/21

La integración para este centro de control se hizo sobre servidores IBM PowerPC P520, con sistema operativo AIX 7. Cuentan con procesadores dual core de 4.2GHz, memoria RAM de 8GB, capacidad de almacenamiento local de aproximadamente 600GB e interfaces Ethernet de alta velocidad.

Estos servidores son de alta disponibilidad y alojan las aplicaciones críticas del sistema. Son por ejemplo los servidores de comunicaciones, SCADA, base de datos, de desarrollo, de entrenamiento. Existen también servidores HP con Red Hat Linux para el controlador de dominio y la zona desmilitarizada.

El equipamiento de conectividad es Cisco, y accesos de comunicaciones para RTUs seriales mediante dispositivos Digi PortServer TS16 mediante el protocolo DNP3, también pueden conectarse remotas por TCP/IP utilizando una versión especial de DNP3. Admite también los protocolos de control IEC-60870-5-101 e IEC-60870-5-104.

Utiliza Oracle Enterprise como motor de base de datos y equipamiento NetApp FAS2020 para el almacenamiento de información histórica.

El sistema cuenta con posibilidades de ser accedido desde el exterior, por ejemplo desde la red corporativa de la empresa. Para ello se utiliza un firewall apropiadamente configurado que permite el acceso a una zona desmilitarizada para consultar información histórica y otros registros.

Figura 46 – Consolas de operación del sistema

Las consolas de operación son HP y corren como sistema operativo Windows 7. La aplicación correspondiente a la interfaz de usuario está desarrollada en Java, puede funcionar tanto de forma local como remota.

Cada consola de operación posee cuatro monitores sobre la que los usuarios del sistema despliegan los listados de alarmas y las pantallas de información que desean visualizar.

Existe un Tablero Mímico que esquematiza la red eléctrica de alta tensión en la zona de influencia del centro de control. Posee aproximadamente 2600 indicadores lumínicos que representan el estado de una gran cantidad de equipos de las subestaciones pertenecientes a la red eléctrica.

El mismo ha estado en funcionamiento desde fines de la década del setenta, aunque ha sufrido varias reformas con el objetivo de mantenerlo actualizado. Originalmente utilizaba lámparas incandescentes de gran consumo, actualmente se ha modernizado para utilizar LEDs de alta luminosidad y bajo consumo.

El Tablero se comunica con el sistema SCADA XA/21 a través de una conexión serial, y lo mismo ha ocurrido con los dos sistemas anteriores, ahora en desuso.

Figura 47 – Imágenes de la integración del sistema durante 2010/2011 en testfloor de General Electric, Melbourne, FL, EEUU.

3. Capítulo III: Objetivo del trabajo

3.1. Enunciado

Identificar en función de la actualidad de los sistemas HMI/SCADA los problemas y desafíos futuros que estas plataformas presentan. Especificar y diseñar un nuevo sistema que utiliza herramientas de programación de código abierto, multiplataforma, basado en web y de bajo costo para el mercado argentino, validando así el propósito del presente trabajo.

3.2. Desarrollo

La tecnología de la automatización y del control ha evolucionado de manera sostenida desde la década de 1960 con la aparición masiva de las computadoras digitales. Desde entonces la mayoría de las tareas simples y repetitivas de manufactura han sido automatizadas mediante máquinas, hecho que ha revolucionado la industria. Los sistemas HMI/SCADA, contenidos dentro de este fenómeno, han jugado un rol importante dentro de esta revolución.

Desde entonces, y mayoritariamente debido a razones históricas relacionadas con la complejidad de estos sistemas, pocas han sido las empresas con la capacidad de desarrollar y proveer este tipo de tecnologías. Esa es la razón por la cual su implementación haya quedado relegada a grandes empresas capaces de enfrentar inversiones importantes para mejorar y modernizar sus procesos de producción.

Paralelamente desde fines de la década de 1980s la cultura del software libre ha permitido el surgimiento constante de herramientas de código abierto de todo tipo, desde entornos de programación hasta motores de base de datos, que desde hace algunos años han evolucionado hasta alcanzar el nivel de alternativas comerciales en cuanto a cualidades y potencialidad.

De la misma forma la constante evolución de las tecnologías de comunicación y su consecuente abaratamiento abre un nuevo panorama para el crecimiento de los sistemas SCADA que haciendo uso de la infraestructura existente pueden conectarse con sistemas de telemetría remotos a un menor costo.

Esto genera el clima ideal y una gran oportunidad para el desarrollo de nuevas tecnologías de automatización, puntualmente de sistemas HMI/SCADA, aprovechando el sustento que estas herramientas brindan.

Los grandes proveedores tradicionales de soluciones de automatización como ABB, Siemens AG, General Electric, Rockwell Automation, etc., parecen atados al viejo paradigma, con productos más bien adaptados a las nuevas tecnologías pero que arrastran conceptos antiguos producto de la inercia tecnológica debido a sus grandes estructuras empresarias.

Queda en evidencia la necesidad de contar con un sistema basado en estas nuevas herramientas de manera de lograr un servicio de aspecto fresco y moderno, versátil, integrado plenamente a la Web 2.0 y a las redes de comunicaciones modernas.

Se busca primero entonces hacer una clara identificación de las falencias comunes que presentan los sistemas actuales, analizando luego las tendencias del sector del control y la automatización e intentar así determinar qué posibilidades y funcionalidades que se les requerirían en un futuro cercano a este tipo de plataformas.

A continuación se realizará una especificación teniendo en cuenta el análisis anterior para posteriormente pasar al diseño del sistema propiamente dicho seleccionando un conjunto de herramientas, bibliotecas y plataformas de código abierto para llevar adelante esta tarea.

4. Capítulo IV: Punto de partida

4.1. Identificación del problema

En la actualidad es una realidad evidente que los procesos industriales así como muchos sistemas modernos dependen cada vez más de los sistemas SCADA y de las tecnologías de control y automatización para llevar adelante funcionalidades complejas. Como ya se ha mencionado los ejemplos típicos en donde este tipo de sistemas son utilizadas cada vez con mayor presencia incluyen la generación, transporte y distribución de energía eléctrica, los procesos de refinamiento de petróleo, los sistemas de almacenamiento y distribución de agua potable, todo tipo de industrias manufactureras, entre muchas otras aplicaciones.

Esto se debe a que las infraestructuras dependen cada vez más del monitoreo del mundo real, la oportuna evaluación de los datos adquiridos y el control, esto plantea nuevos desafíos. Los dispositivos de control y las magnitudes que necesitan ser monitoreadas y controladas se ha incrementado a una gran velocidad. Acompañando este fenómeno también se han incrementado en la misma medida los requerimientos de monitoreo y análisis de alta performance así como también la necesidad de una gestión cada vez más eficiente.

Las arquitecturas de los sistemas SCADA tradicionales fueron diseñadas para ambientes industriales cerrados, con infraestructuras de comunicaciones y control propias o dentro de ambientes estrictamente controlados.

Sin embargo con el crecimiento constante de Internet, las nuevas redes de comunicaciones y la estandarización e interoperabilidad de tecnologías existe un gran potencial para mejorar su funcionalidad y minimizar los costos de integración si se los incorpora en un enfoque colaborativo con otros sistemas empresarios y los servicios de gran escala del mundo real. En la medida que los ambientes se vuelven más complejos y brindan nuevas posibilidades de comunicación y control, ya no es viable ni eficiente diseñar sistemas auto-contenidos.

El nuevo enfoque indica que estos deben estar preparados para integrarse a un sistema de gran escala junto a otros sistemas que desempeñen otras funciones. Es necesario considerar cuales deben ser los próximos pasos a tener en cuenta en la evolución de la actual generación de sistemas SCADA.

4.2. Evolución

En las últimas décadas ha habido un cambio notable en la arquitectura de los sistemas SCADA. Como ya se ha expuesto en la primera generación se tenían sistemas monolíticos conectados mediante vínculos punto a punto a las RTUs en el lugar de trabajo. En una segunda generación se favoreció un esquema con mayor distribución donde redes de área local (LAN) eran usadas para la interconexión de los componentes. Este enfoque era más rentable que la primera generación y permitía procesamiento distribuido e intercambio de información en tiempo real entre las entidades del sistema basado en protocolos propietarios. La aparición de la tercera

generación se encamina hacia el uso de la red como tal, usando no solamente WAN sino también Internet y usando protocolos abiertos.

Mirando hacia adelante es un hecho que las empresas modernas necesitan ser ágiles y soportar el proceso de toma de decisiones a diversos niveles. Para que esto funcione, la información crítica necesita estar disponible en el punto exacto y de forma oportuna.

En las infraestructuras futuras una enorme cantidad de datos son generados por los dispositivos del mundo real. Estos necesitan ser integrados, procesados dentro de un contexto específico y comunicados bajo demanda y a tiempo.

4.3. Nuevos desafíos

Los nuevos sistemas SCADA actuales se enfrentan con la necesidad de procesar una mucha mayor cantidad de información distribuida y en tiempo real. Comienza a ser importante que las decisiones operativas se tomen tomando en cuenta los datos internos y externos e información adquirida. Además estos datos, análisis y resultados deben quedar expuestos al mundo exterior como servicios para que otros sistemas y personas puedan consultarlos.

El cambio más importante en la infraestructura es una mayor tendencia a la distribución, aprovechándose de redes de comunicaciones con calidad de servicio y servidores de alta disponibilidad. Además como el volumen de información manejado es mucho mayor y teniendo en cuenta este panorama distribuido en donde los módulos tienen una ubicación flexible, se requieren mayores anchos de banda.

Las partes del sistema interaccionan entre sí y se relacionan con otros sistemas de la organización mediante servicios. A través de ellos es que se exponen todas sus funcionalidades.

Se citan a continuación otros tipos de sistemas con los que es cada vez más importante establecer una interacción.

- ✓ **ERP**: Enterprise Resource Planning o Planificación de Recursos Empresarios son sistemas orientados a producir información útil para las áreas gerenciales de la organización sobre aspectos operativos, contables, de existencias o stock y de logística entre otros.
- ✓ **MES**: Manufacturing Execution System o Sistema de Ejecución de Manufactura son plataformas diseñadas para mejorar las actividades operativas mediante la generación de informes de producción, mantenimiento, entre otros.

Estos sistemas se ven muy beneficiados con la información fresca y en tiempo real proveniente de un sistema SCADA asociado a procesos de producción. La siguiente integración queda bien representada con la jerarquía por capas que se representa mediante la Figura 48 conocida como pirámide de la automatización:

Figura 48 – Pirámide de la automatización

Fuente: http://processengineering.theengineer.co.uk/pictures/460xAny/2/1/9/2044219_No2-pyramid.jpg

En la base tenemos los sensores, actuadores y demás elementos de planta, sobre ellos se encuentran los PLCs, las RTUs y demás dispositivos control y adquisición de datos desde estos sensores y actuadores. Una capa más arriba en la pirámide se encuentran los sistemas SCADA. Encima de éste se ubica el MES y en la capa superior el ERP.

También debería existir una interacción fluida con otros SCADAs, del mismo u otro fabricante, y diversas fuentes de información, es decir otras bases de datos, gateways o mediadores de servicio a dispositivos de control antiguos, etc.

En la Figura 49 se presenta un esquema de lo que podría ser la arquitectura de los próximos sistemas SCADA.

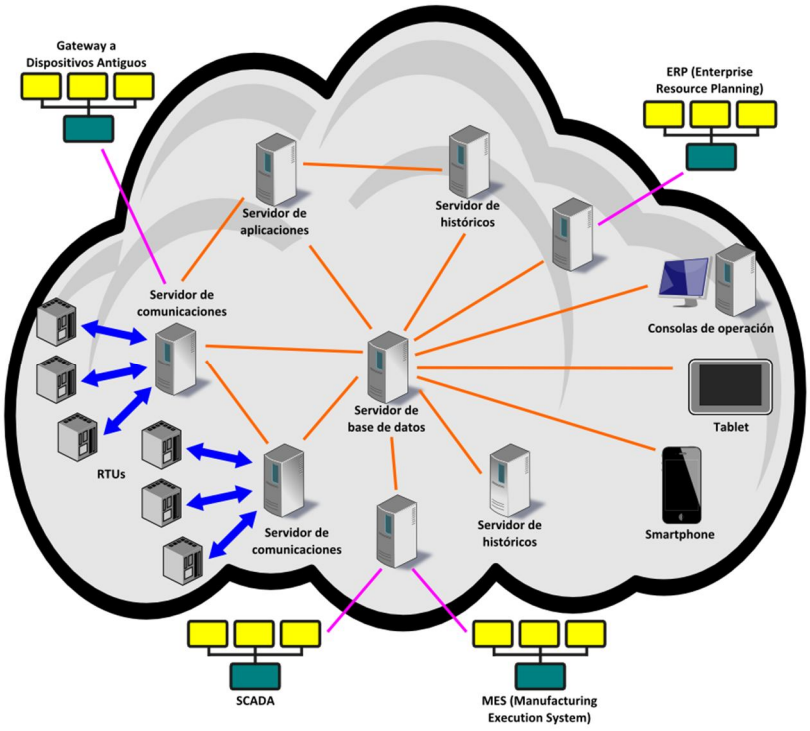


Figura 49 – Visión de una nueva generación de sistemas SCADA

Por su parte la interfaz hombre-máquina ya no está relacionada con una ubicación estática, sino que es accesible desde cualquier lugar y en cualquier momento mediante cualquier dispositivo, consolas de operación tradicionales, computadoras de escritorio, tabletas, smarthphones y demás dispositivos inteligentes con capacidades de conectividad. Además su funcionalidad no es monolítica, sino compuesta por una combinación de aplicaciones accedidas a través de servicios alojadas en los dispositivos y en la nube.

El monitoreo de la información así como el control de los procesos vinculados se hace de forma colaborativa y es llevada a cabo por varios entes de forma simultánea. En la actualidad existen disponibles servicios en la nube que pueden realizar análisis de alta performance de los datos monitoreados. Por su parte sistemas de soporte de decisiones pueden analizar datos en tiempo real que vengan no solo de los dispositivos sino desde procesos del negocio interconectados.

En cuanto a la funcionalidad de la RTU está ahora representada por mediadores de servicio o incluso en dispositivos inteligentes que ahora reportan directamente sobre la red los datos recolectados. Dependiendo de las capacidades ofrecidas por los dispositivos, la información puede ser también procesada previamente antes de ser enviada, cumpliendo los requerimientos establecidos por el sistema de supervisión y sin tener que transmitir información no necesaria o redundante.

Una infraestructura de comunicaciones conectando a todos los componentes es todavía la columna vertebral, sin embargo está mucho más diversificada sobre varios canales cableados e inalámbricos, brindando calidad de servicio dependiendo de las necesidades dinámicas de la aplicación. La introducción de las tecnologías de Internet en todas partes y la interacción basada en servicios facilita la integración y la interoperabilidad, llevando a disminuir el costo de operación y el rápido despliegue de soluciones altamente personalizadas.

Es esperable que la próxima generación de sistemas SCADA forme parte de un conjunto de personas, sistemas, dispositivos y procesos que necesiten colaborar para cumplir los objetivos. Mediante el uso de indicadores complejos de performance será posible evaluar en tiempo real el estado de cualquier capa, evaluar alternativas, y ajustar recursos para alcanzar una performance óptima.

Se espera que los sistemas futuros sean capaces de colaborar e intercambiar información de entre capas de manera transparente, fluida, oportuna y mediante tecnologías abiertas. Se conforma así un sistema de sistemas que recorre la organización de punta a punta. La infraestructura de control debería ser fácilmente actualizable y con sentido de evolución flexible en cuanto a los requerimientos del negocio, haciendo posible su integración futura con otras tecnologías a largo plazo.

Por su parte la siguiente generación de sistemas SCADA podría no ser de naturaleza física. Esto implica que podrían residir solo en el mundo virtual, en el sentido que van a estar compuestos de varios dispositivos del mundo real, servicios en los dispositivos y en la red, e interacciones impulsadas por colaboración, que compondrán un sistema de sistemas distribuido, ágil, colaborativo y complejo.

5. Capítulo V: Esbozo de la Solución

Durante el capítulo II se ha realizado un recorrido histórico de la evolución de los sistemas HMI/SCADA hasta llegar a su actualidad. En el capítulo III se planteó como objetivo el desarrollo de un nuevo sistema que se adapte a los requerimientos actuales del sector. A continuación en el capítulo IV se identificaron las falencias comunes de este tipo de sistemas, se analizó su evolución y los nuevos desafíos. Se pretende ahora en el presente capítulo proponer una serie de mejoras concretas con las que debería contar un nuevo sistema SCADA. Teniendo en cuenta todo esto se procederá a exponer en que consiste el sistema desarrollado.

5.1. Especificación de características requeridas

- ✓ **Multiplataforma**
  El sistema debe estar disponible en varios sistemas operativos. Esto implica en consecuencia que podrá funcionar en varias plataformas de hardware. De esta forma no se restringe al usuario a tener que usar un determinado sistema operativo, pudiendo elegir opciones de código abierto, como alguna de las distribuciones de Linux, y de esta forma no tener que incurrir en la compra de licencias de software.
- ✓ **Basado en herramientas de programación de código abierto**
  Se ponderará la utilización de herramientas de programación de código abierto, de probada eficacia y soportadas por comunidades de usuarios grandes. Se evitará el uso de plataformas y bibliotecas privativas o que requieran la compra de licencias adicionales.
- ✓ **Interfaz de usuario basada en Web 2.0**
  El acceso al sistema deberá ser basado en Web, lo que significa que no deberá requerir la instalación de software especial, será suficiente con que el usuario ingrese a un navegador de Internet e introduzca una dirección para conectarse y comenzar a operar.
  Para lograrlo se plantea un modelo cliente-servidor. El cliente maneja la interfaz de usuario. El servidor procesa las peticiones del cliente y accede a la base de datos en función de éstas.
- ✓ **Modular**
  Debe ser escalable y poder crecer de acuerdo a las necesidades del usuario, pudiendo éste seleccionar que partes del sistema le son útiles para su aplicación en particular. Los módulos deben poder trabajar en conjunto, en un mismo servidor o plataforma, o por separado, contando con las facilidades de comunicación necesarias para poder utilizarse a través de Internet en la nube.
- ✓ **Interoperable con la mayoría de las redes de comunicaciones**
  En cuanto a la comunicación del sistema con los dispositivos y elementos desde los que se nutre de información es importante que el mismo tenga flexibilidad para poder operar sin problemas tanto con vínculos punto a punto tradicionales como por Internet y redes móviles.
- ✓ **Seguro**

Teniendo en cuenta que el sistema enviará y recibirá información a través de redes públicas se deberá poner especial énfasis en la protección de esta información contra ciberataques que pudieran poner en riesgo su seguridad y su integridad.

5.2. Diseño de arquitectura básica

Se define una arquitectura básica de operación que consta de cinco módulos o secciones fundamentales comunicadas entre sí como puede verse en la **Figura 50**.

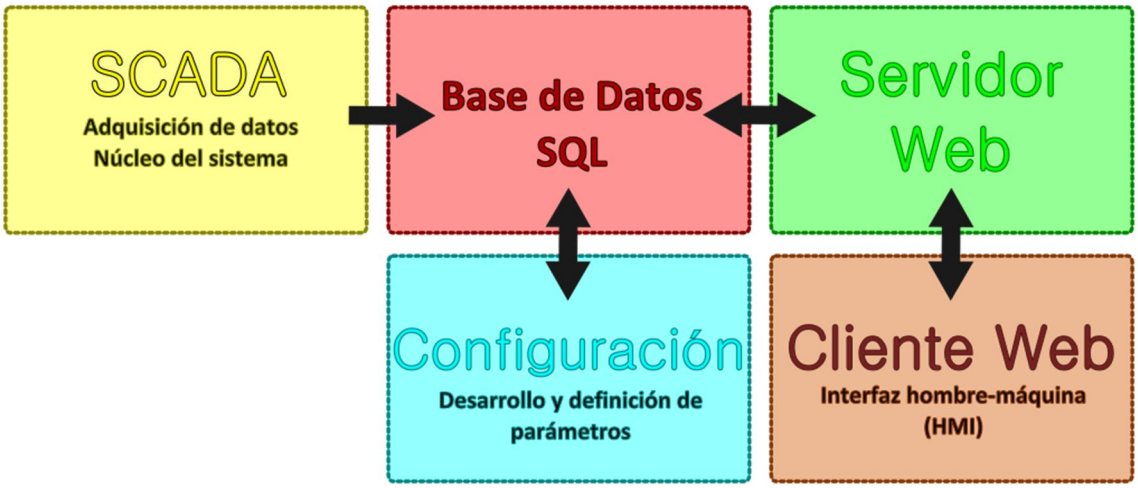


Figura 50 – Arquitectura básica del sistema

Base de datos SQL

El sistema se apoya sobre un motor de base de datos del tipo RDBMS (Relational Database Management System o Sistema de Gestión de Bases de Datos Relacionales). Se ha definido una estructura de tablas con distintas funciones y propósitos que será expuesta en un capítulo especial.

Todos los módulos del sistema, excepto la interfaz cliente web, se comunican con distintas secciones de la base de datos para obtener y guardar datos.

Módulo SCADA o Núcleo

Representa el núcleo del sistema es decir su parte principal, en él se realiza la adquisición de datos. Su función fundamental es la de realizar la interrogación periódica y secuencial de los dispositivos definidos para volcar luego esta información en la base de datos. Estos dispositivos pueden estar conectados a este módulo por diversos medios de comunicaciones.

Esto puede ser en forma directa, con un simple cable, a través de un vínculo punto a punto de radio o par de cobre o incluso a través de Internet o redes móviles.

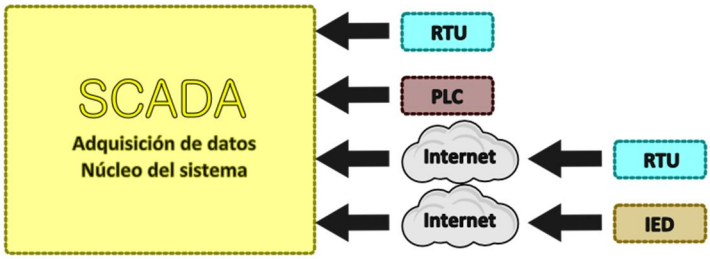


Figura 51 – Diagrama básico SCADA de adquisición de datos

Módulo de Configuración y Desarrollo

Permite al desarrollador del sistema definir todos los parámetros de configuración. Contiene todas las herramientas necesarias para permitir al usuario final poder realizar todas las modificaciones que crea necesarias sin necesidad de recurrir al proveedor del sistema. El objetivo es lograr una aplicación simple de usar incluso para personal sin demasiados conocimientos de este tipo de sistemas. En ese sentido es importante incorporarle una interfaz gráfica para facilitar y hacer más amena la creación de configuraciones.

Módulo HMI

El módulo HMI se maneja mediante un modelo cliente-servidor pudiendo funcionar o no en la misma computadora o servidor. El acceso debe ser multiusuario y concurrente, es decir que varios usuarios podrían estar conectados operando el sistema de forma simultánea.

Servidor Web

El servidor recibe peticiones desde el cliente y las canaliza hacia la base de datos, las informaciones recuperadas desde ésta son acondicionadas de acuerdo al requerimiento y enviadas en un formato apropiado hacia el Cliente Web.

Cliente Web

Los usuarios que deseen conectarse al sistema para su operación deberán solicitar primero la creación de un usuario al administrador. El acceso al sistema se realiza mediante un navegador Web pudiendo utilizar cualquiera de los más populares: Mozilla Firefox, Google Chrome, Opera, Safari, etc. Es suficiente con ingresar a una cierta dirección, como si se tratara de cualquier otra página web. A continuación la aplicación es cargada en el navegador, en cuestión de segundos el operador puede estar controlando el proceso en cuestión utilizando una computadora de la red local donde funcione el sistema o a través de Internet mediante cualquier computadora que tenga acceso a la WWW.

Sistema gráfico

El aspecto gráfico es un elemento fundamental en los sistemas SCADA debido a que mediante su uso se realizará la representación de los procesos bajo supervisión. Trabajará fuertemente ligado al módulo HMI, debido a que es justamente en la interfaz de operación del sistema

donde se presentarán las pantallas gráficas. Dado que el acceso a la misma se realiza mediante la web es necesario seleccionar una herramienta fuertemente arraigada a este tipo de tecnología.

5.3. Herramientas de programación a aplicar

Para la programación del sistema se ha seleccionado para su aplicación una serie de herramientas, plataformas y bibliotecas de código abierto. Las mismas son libres y están disponibles para su descarga desde Internet. Se ha procurado elegir plataformas de probada eficacia teniendo en cuenta los siguientes preceptos:

- ✓ Que tengan varios años de historia.
- ✓ Que sean utilizadas en otros desarrollos importantes.
- ✓ Con una gran comunidad mundial de usuarios que las respalde.
- ✓ Con un desarrollo activo que incorpore nuevas funcionalidades constantemente.
- ✓ Con corrección periódica de bugs y problemas.

Se han seleccionado, en función de esto, las siguientes plataformas.

- ✓ **Motor de base de datos**
  Entre las opciones de código abierto disponibles se pensó en primer lugar en utilizar MySQL de la empresa Oracle, un producto conocido mundialmente y utilizado en una enorme cantidad de aplicaciones. Sin embargo su esquema de licenciamiento GPL permite su utilización libre y gratuita sólo en proyectos también de código abierto, en proyectos comerciales donde no se desee revelar el código se debe comprar una licencia.
  Teniendo en cuenta las posibilidades de comercialización del sistema a futuro se consideró entonces como alternativa PostgreSQL, otro motor de base de datos que se define a sí mismo como "la base de datos de código abierto más avanzada del mundo". Esta afirmación podría sonar pretensiosa, sin embargo teniendo en cuenta la gran cantidad de empresas, organizaciones e instituciones que la utilizan en sus aplicaciones (http://www.postgresql.org/about/users/), la enorme comunidad de usuarios que la respaldan y el desarrollo activo que posee esta pareció la opción indicada. Por su parte su esquema de licenciamiento particular (http://opensource.org/licenses/postgresql) permite su aplicación incluso en proyectos comerciales de código cerrado.
- ✓ **Módulo SCADA y Módulo de Desarrollo y Configuración**
  En un principio, teniendo en cuenta que la aplicación debía ser multiplataforma y analizando opciones libres y de código abierto, se pensó en utilizar Java. Sin embargo como se sabe esta plataforma consigue funcionar en diferentes plataformas mediante la utilización de una máquina virtual (JVM, Java Virtual Machine o Máquina Virtual Java) creando una capa de abstracción entre la aplicación y el sistema operativo.
  Dado que se ponderó el acceso directo de la aplicación a los recursos de hardware de la computadora o servidor donde el sistema corriera, principalmente debido a que es fundamental el acceso con control absoluto, de bajo nivel y sin intermediarios a los

puertos de comunicaciones de la computadora, se decidió finalmente obviar esta posibilidad.

Se seleccionó entonces otra biblioteca o framework libre llamada Qt actualmente propiedad de la empresa Digia, la misma utiliza el lenguaje C++, es multiplataforma pero a diferencia de Java no hace uso de una máquina virtual sino que utiliza en cada plataforma un compilador diferente creando así programas objeto nativos para cada sistema operativo en el que la plataforma se encuentra disponible. Tiene un esquema de licenciamiento dual, con una versión bajo licencia LGPL que permite la utilización de la biblioteca en proyectos comerciales de código cerrado.

- ✓ **Servidor Web**

  Como soporte de esta función se eligió el sistema Nginx, un servidor web de código abierto usado por varios sitios de Internet importantes como Netflix, Wordpress, SourceForge, Facebook, Dropbox, Source Forge, Github, entre otros. Su licencia permite su utilización en productos de código cerrado.

  Se había considerado en un principio el servidor Apache, sin embargo si bien Nginx no tiene la misma popularidad que su contrincante es considerablemente más liviano, consume menos recursos de hardware y tiene un rendimiento excelente en comparación . Hay diversas pruebas de rendimiento disponibles en Internet que avalan este concepto (para citar una de ellas el lector puede acceder por ejemplo a http://wiki.dreamhost.com/Web_Server_Performance_Comparison, último acceso Mayo de 2015). Por su parte tiene una licencia aún menos restrictiva, lo que lo vuelve atractivo para este tipo de desarrollos.

  Para el desarrollo de la programación del lado servidor Web se seleccionó el lenguaje de scripting PHP. El mismo fue usado históricamente para este tipo de usos, es decir la creación de sitios web con contenido dinámico. Es considerado por la comunidad de desarrolladores web como uno de los lenguajes más potentes, flexibles y de alto rendimiento. Es utilizado en sitios como Facebook y en aplicaciones donde se requiera una gran demanda de tráfico.

- ✓ **Cliente Web**

  Para la programación del cliente Web se eligió Javascript. El mismo es un lenguaje de programación dinámico que mediante el uso de scripts permite comunicarse asincrónicamente con el servidor, interactuar con el usuario y controlar el contenido de la interfaz. Está disponible para ser usado en la gran mayoría de los navegadores web, esto significa que cuando uno descarga desde Internet alguno de los navegadores más populares está descargando al mismo tiempo una copia de Javascript lista para ser usada. Dado que se le dio un especial énfasis a que la interfaz de usuario cliente del sistema no requiriera la instalación de software adicional, excepto un navegador de Internet, esta característica fue importante para su selección.

  Se seleccionó además la biblioteca jQuery mediante la cual se simplifica notablemente la programación.

- ✓ **Sistema gráfico**

  Dada la naturaleza web del módulo HMI se seleccionó para la representación gráfica la especificación SVG. La misma es una implementación de gráficos vectoriales escalables soportada por todos los navegadores más populares de la actualidad. Los

mismos pueden ser redimensionados sin perder calidad. Por su parte los archivos tienen un reducido tamaño en comparación con los mapas de bits por lo que no tienen grandes requerimientos en cuanto a ancho de banda para ser enviados a través de redes de datos. Además estos documentos son fácilmente manipulables utilizando JavaScript. Estas características hacen de esta especificación una herramienta ideal para la representación de pantallas gráficas de supervisión en sistemas de este tipo.

Todas las herramientas elegidas cumplen con los preceptos planteados anteriormente, es decir tienen varios años de historia, fueron usadas en el desarrollo de otras aplicaciones importantes, existe una gran comunidad de usuarios y programadores que las respalda y hay un trabajo constante que busca incorporar nuevas funcionalidades y reducir los errores de código en las mismas.

5.4. Descripción de las herramientas

5.4.1. PostgreSQL

PostgreSQL es un sistema de gestión de base de datos relacional conocido como RDBMS (Relational Database Management System). Su primera implementación se lanzó en 1986, siendo en la actualidad la base de datos de código abierto más avanzada del mundo. Tiene más de 15 años de desarrollo activo y una probada arquitectura que ha ganado una fuerte reputación de confiabilidad, integridad de información y corrección. A diferencia de otros motores de base de datos propietarios, PostgreSQL es muy conocida por ser usada en implementaciones con gran tráfico de datos y con años de servicio ininterrumpido.

Puede funcionar en la mayoría de los sistemas operativos más conocidos, incluyendo Linux, Unix (AIX, BSD, HP-UX, SGI, IRIX, Mac OS X, Solaris, Tru64) y Windows. Tiene soporte completo de foreign keys, joins, views, triggers y stored procedures. Cumple con la mayoría de los tipos de datos especificados en el lenguaje SQL:2008 como INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL y TIMESTAMP. Soporta también el almacenaje de grandes objetos, como fotos, audios o videos. Tiene interfaces de programación para los lenguajes más conocidos como C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros.

La mayoría del código de PostgreSQL está disponible bajo una licencia de código abierto liberal: La licencia PostgreSQL (http://opensource.org/licenses/postgresql), que da libertad de usar, modificar y distribuir el sistema de cualquier forma que uno desee, en código abierto o cerrado. Por lo tanto PostgreSQL no es solo un poderoso sistema de base de datos sino una plataforma de desarrollo sobre la cual desarrollar productos de software, comerciales o no, que requieran una RDBMS a la altura.

5.4.1.1. Breve historia

El predecesor de PostgreSQL era Ingres, una de base de datos desarrollada en la Universidad de California, en Berkeley EEUU entre 1977 y 1985 bajo la tutela de Michael Stonebraker. Más tarde sería transformada en un producto por la firma Relational Technology quien se enfocó en el desarrollo de un servidor de base de datos orientado a objetos llamado Postgres (post es

la palabra latina para después, y dado que vino luego de Ingres se lo nombró con la contracción de ambas palabras). Usaba un lenguaje de consulta llamado _Quel_ como también lo hacía Ingres.

La empresa más tarde cambió su nombre a Ingres, coincidiendo en el nombre de la base de datos. Más tarde sería comprada por Computer Associates lo que causó un masivo éxodo de ingenieros.

En 1992, Michael Stonebraker, Gary Morganthaler, Michael Ubell y Paula Hawthorn se unieron para crear la empresa Miró. La firma tomó el código fuente de la Universidad de California y lo adaptó inmediatamente para usar SQL, esto ocurrió en 1995. Miró se transformó en Montage y luego en Illustra. El producto de la empresa se basaba en el de la universidad pero tenía una infraestructura muy mejorada, se mejoraba el código que escribían los estudiantes de la universidad que participaban en el proyecto y se agregaron herramientas de producción e interfaces de usuario.

En 1994 se completó oficialmente el proyecto Postgres en la universidad con la versión 4.2, debido a que mantener la gran cantidad de código tomaba mucho tiempo, el cual debía ser dedicado a la investigación de bases de datos.

Los graduados de Berkeley Jolly Chen y Andrew Yu mejoraron el soporte SQL de Postgres en 1994 y 1995. El proyecto se llamó Postgres95. Chen y Yu terminaron su carrera. Chen continuó con el desarrollo y el mantenimiento de Postgres95. Fue convertida completamente a ANSI C y tenía algunas mejoras importantes con respecto a Postgres 4.2. Más tarde Yu comenzó a trabajar en Illustra y Chen en WebLogic por lo que dejaron de tener tiempo para dedicarle al proyecto.

Entonces Jolly Chen expresó: “Este código necesita poca gente con mucho tiempo, no mucha gente con poco tiempo”. Ni Chen, ni Yu se veían como propietarios del desarrollo, en palabras de Chen: “Ni Andrew ni yo nos sentíamos los dueños del proyecto, después de todo Postgres fue el resultado del trabajo de muchos estudiantes de graduación durante años, no sólo el nuestro”. Para ese momento el código fuente tenía aproximadamente 250.000 líneas de C, al parecer tenía razón.

Se forma entonces un equipo para continuar el desarrollo llamado PostgreSQL Global Development Team. El desarrollo del núcleo quedó a cargo de un grupo formado por cuatro personas: Marc Fournier de Ontario (Canadá), Thomas Lockhart de Pasadena (EEUU), Vadim Mikheev de Krasnoyarsk (Russia) y Bruce Momjiam de Philadelphia (EEUU).

En 1996 era obvio que Postrgres95 no era el nombre correcto para la base de datos en el futuro, y se eligió PostgreSQL. El nombre refleja la relación entre el viejo sistema Postgres y la nueva funcionalidad SQL. El principal objetivo del equipo era mejorar el producto agregando características como sistemas de control de concurrencia multiversión, mejora de la velocidad, inclusión de ANSI SQL 92, subselects, triggers, defaults y constraints.

Al mismo tiempo Informix adquiere Illustra. Se trabajaba para convertir el producto de la tecnología existente basada en multi-threading a una orientada a objetos. Luego de un año de arduo trabajo finalmente se lanza el producto.

La ausencia de ventas significativas hace que algunos miembros del equipo se acerquen a Stonebraker con la idea de convertir la base de datos de Illustra en código abierto. Entonces se inician conversaciones informales con el PostgreSQL Global Development Team para mezclar los desarrollos, la parte corporativa de Informix nunca terminaría de aceptar esto. En 2001 Informix se divide y la porción de base de datos es vendida a IBM.

En 2000 se funda PgSQL INC con Marc Fournier como presidente, e integrada por los desarrolladores del core del producto, para ofrecer soporte comercial de PostgreSQL. La relación entre PgSQL INC y PostgreSQL siempre ha sido simbólica.

Desde entonces centenares de desarrolladores en todo el mundo trabajaron y trabajan en el desarrollo PostgreSQL comunicados a través de Internet. El número de usuarios aumenta constantemente.

Una enorme comunidad de profesionales y entusiastas respalda el producto, brindando un soporte muchas veces superior al de los productos propietarios.

5.4.2. Qt

Qt es una biblioteca de desarrollo de aplicaciones multiplataforma con interfaz de usuario. Las aplicaciones creadas en Qt pueden funcionar en varias plataformas de software y hardware con cambios mínimos en el código, conservando el poder, la velocidad y el aspecto de las aplicaciones nativas de cada plataforma.

Qt usa C++ estándar con extensiones, incluyendo señales y slots (signals and slots) un lenguaje para comunicación entre objetos que simplifica el manejo de eventos, lo que ayuda en el desarrollo de aplicaciones GUI y de servidor. Soporta muchos compiladores, incluyendo GCC C++ y la suite Visual Studio. Corre en la mayoría de las plataformas de escritorio y en algunas plataformas móviles. Permite el acceso a bases de datos SQL, manejo de datos XML y JSON, gestión de threads y soporte de red.

Principalmente se destacan la gran cantidad de bibliotecas disponibles para esta plataforma destinadas al desarrollo de aplicaciones, cubriendo ampliamente todos los requerimientos para la creación de un módulo de estas características. Otro factor importante es su despliegue multiplataforma, permitiendo que las aplicaciones desarrolladas puedan funcionar en una diversa gama de hardware y software realizando cambios mínimos en el código.

Se destaca también su carácter comunitario, el cual está respaldado por una extensa comunidad de desarrolladores de todo el mundo que se comunican a través de foros de intercambio de información, mantienen la extensa documentación constantemente actualizada y colaboran en la mejora continua de la plataforma.

Por último, a pesar de contar con una versión gratuita para todas las plataformas de hardware y software en las que opera, maneja un esquema de licenciamiento (LGPL) que permite la comercialización de aplicaciones que el programador desarrolle y sin la restricción de dar a conocer el código fuente de la obra.

Todas estas características hacen de ella una de las bibliotecas de desarrollo más usadas a nivel mundial.

5.4.2.1. Arquitectura de software

Cuando Qt fue lanzado por primera vez, se basó en los siguientes conceptos:

- ✓ Abstracción completa de la interfaz de usuario: Cuando fue lanzado Qt usaba su propio motor de dibujo y controles, emulando el aspecto de las diferentes plataformas sobre las que corría. Las versiones más recientes usan las APIs nativas de las diferentes plataformas para esta tarea, cuando están disponibles.
- ✓ Señales y slots: Mecanismo introducido por Qt para la comunicación entre objetos que facilita la implementación del patrón de diseño conocido como "observador" evitando la repetición de código. El concepto es que los widgets gráficos pueden enviar señales que contienen información de eventos, los cuales pueden ser recibidos por otros controles que usan funciones especiales conocidas como slots.
- ✓ Compilador metaobjeto: Es una herramienta que se corre sobre el código de un programa Qt. Interpreta ciertas macros como anotaciones, y las usa para generar código adicional que es agregado. Se utiliza para brindar al programador características no disponibles de forma nativa en C++: Señales y slots, llamado asíncrono a funciones, etc.

5.4.2.2. Plataformas soportadas

Qt corre en varias plataformas distintas, lo que lo hace atractivo para aquellos que desean que un código fuente funcione en todas ellas, por lo que resulta muy apropiado para su utilización en la programación del sistema HMI/SCADA en cuestión. Las siguientes plataformas son actualmente soportadas:

- ✓ Android.
- ✓ Linux/Embedded Linux.
- ✓ Integrity.
- ✓ iOS.
- ✓ OS X.
- ✓ QNX / Blackberry 10.
- ✓ VxWorks.
- ✓ Wayland.
- ✓ Windows XP / Windows Vista / Windows 7 / Windows CE / Windows RT.
- ✓ FreeBSD.

5.4.2.3. Resumen de la historia de Qt

El framework Qt estuvo disponible públicamente por primera vez en Mayo de 1995. Fue desarrollado por Haavard Nord y Eirik Chambe-Eng. Haavard y Eirik se conocieron en el Instituto Noruego de Tecnología en Trondheim, donde ambos se graduaron en una maestría en ciencias de la computación.

El interés de Haavar's en el desarrollo de interfaces gráficas de usuario (GUI, graphic user interface) C++ comenzó en 1988 cuando fue contratado por una empresa sueca para desarrollar un framework C++ para el desarrollo de aplicaciones con interfaces gráficas. Un par de años más tarde, en el verano de 1990, Haavard y Eirik trabajaban juntos en una aplicación de base de datos C++ para imágenes por ultrasonido. El sistema necesitaba poder correr con la misma GUI en Windows, Macintosh y Unix. Un día de ese verano, Haavard y Eirik se encontraban en el banco de una plaza cuando Haavard dijo: "Necesitamos un sistema de interfaz orientado a objetos". La discusión resultante llevó a la fundación intelectual del framework GUI orientado a objetos y multiplataforma que pronto crearían.

En 1991, Haavard comenzó a escribir las clases que eventualmente se volvieron Qt, con la colaboración de Eirik en el diseño. El año siguiente, Eirik tuvo la idea de "señales y slots", un paradigma de programación GUI simple pero poderoso que sería luego adoptado por otros entornos de programación. Haavard tomó la idea y la codificó. Para 1993, Haavard y Eirik habían desarrollado el primer kernel gráfico de Qt y estuvieron listos para implementar sus propias herramientas para entornos gráficos. Para el final del año, Haavard sugirió que deberían comenzar juntos un negocio para construir "el mejor framework GUI C++ del mundo".

El año 1994 comenzó de forma poco auspiciosa con dos jóvenes programadores tratando de ingresar a un mercado bien establecido, sin consumidores, un producto sin terminar y sin dinero. Afortunadamente sus esposas tenían empleo y podían apoyar a sus esposos durante los dos años que Eirik y Haavard estimaban que necesitarían para desarrollar el producto y empezar a producir ingresos.

La letra "Q" fue elegida como el prefijo de las clases porque lucía bella en el editor de texto Emacs de Haavard. La "t" fue agregada por "toolkit" o caja de herramientas. La empresa se creó el 4 de Marzo de 1994, se llamó originalmente Quasar Technologies, luego Trolltech.

En Abril de 1995, gracias al contacto realizado por uno de los profesores de universidad de Haavard, la empresa noruega Metis los contrató para desarrollar software basado en Qt. Por ese entonces, Trolltech contrató a Arnt Gulbrandsen, quien durante seis años ideó e implementó un ingenioso sistema de documentación además de contribuir al código de Qt.

El 20 de Mayo de 1995, Qt 0.90 fue cargado a sunsite.unc.edu. Seis días después, el lanzamiento fue anunciado en comp.os.linux.announce. Este fue el primer lanzamiento oficial de Qt. Qt podía ser usado tanto en Windows como en Unix, ofreciendo la misma API en ambas plataformas. Y estaba disponible en dos tipos de licencia desde el primer día: Una comercial requerida para el desarrollo de aplicaciones comerciales, y una libre disponible para desarrollo de código abierto. El contrato de Metis mantuvo a Trolltech a flote, dado que por diez largos meses nadie compró una licencia comercial.

En Marzo de 1996, la Agencia Espacial Europea se convirtió en el segundo cliente de Qt, con la compra de diez licencias comerciales. Fue entonces que Eirik y Haavard contrataron a otro desarrollador. Qt 0.97 fue lanzado para fines de Mayo, y el 24 de Septiembre de 1996 se lanzó Qt 1.0. Para el final del año se liberó la versión 1.1; ocho clientes, de distintos países, habían

comprado 18 licencias en total. En este año también se produjo la fundación del proyecto KDE, llevado adelante por Matthias Ettrich.

Qt 1.2 fue lanzado en Abril de 1997. Matthias Ettrich decidió utilizar Qt para el desarrollo de KDE, lo que ayudó a Qt a volverse el estándar de facto para el desarrollo de aplicaciones GUI C++ en Linux. En Septiembre de 1997 se lanza Qt 1.3.

Matthias se unió a Trolltech en 1998, y Qt 1.4 fue lanzado en Septiembre de ese año. En Junio de 1999 Qt 2.0 vio la luz, con una nueva licencia de código abierto, la Q Public License (QPL), que cumplía con la Open Source Definition. En Agosto de 1999 Qt ganó el premio LinuxWorld para la mejor biblioteca/herramienta. En ese entonces se establecía Trolltech Pty Ltd en Australia.

En el año 2000 Qt lanza Qtopia Core (llamado luego Qt/Embedded). Se diseñó para correr en dispositivos Linux embebidos y proveía su propio sistema de ventanas como reemplazo al existente X11. Tanto Qt/X11 como Qtopia Core serían ofrecidos bajo licencia GPL (General Public License) así como también bajo licencias comerciales. Para el fin del año 2000, Trolltech se estableció en Estados Unidos y había lanzado la primera versión de Qtopia, una plataforma de aplicaciones para teléfonos móviles y PDAs. Qtopia Core ganó el premio "Mejor Solución Linux Embebida" de LinuxWorld en 2001 y 2002, el teléfono Qtopia tuvo el mismo reconocimiento en 2004.

Qt 3.0 se lanzó en 2001, ahora disponible para Windows, Mac OS X, Unix y Linux. Tenía 42 nuevas clases y su código superaba las 500.000 líneas. Qt3 era un gran paso adelante desde Qt 2, incluyendo soporte considerablemente mejorado de Unicode y configuración regional, un widget para edición y visualización de texto, una clase de expresiones regulares tipo Perl. Qt 3 ganó el premio "Jolt Productivity Award" de Software Development Times en 2002.

A mediados de 2005 se anunció el lanzamiento de Qt 4.0. Con alrededor de 500 clases y 9000 funciones era más grande y completo que las versiones precedentes. Se dividió en varias bibliotecas para que los desarrolladores solo tuvieran que compilar las partes que necesitaran. Representó un enorme avance con grandes mejoras. Qt 4 es la primera edición disponible para desarrollo comercial y de código abierto en todas las plataformas que soporta.

También en 2005, Trolltech abre una representación en Beijing para brindar a los clientes de China servicios de venta, capacitación y soporte técnico de Qtopia.

En 2007 comienza el desarrollo de Qt Creator, un entorno integrado de desarrollo de Qt que incluye un debugger, un diseñador de ventanas y herramientas como autocompletado y resaltado de sintaxis. No sería lanzado oficialmente hasta Marzo de 2009.

A mediados de 2007 Nokia, la empresa Finlandesa de telefonía móvil, se manifestó interesada en adquirir Trolltech. Finalmente en Enero de 2008 y luego de varias negociaciones, se anunció un acuerdo por el cual Nokia compraría la empresa por una suma cercana a los 100 millones de Euros. La operación fue aprobada en Junio de 2008 y todas las acciones de Trolltech pasaron a ser propiedad de Nokia. A fines de Septiembre Nokia cambió el nombre de la recientemente adquirida empresa a Qt Software. Por su parte Qtopia pasó a llamarse Qt Extended.

Más tarde en Marzo de 2011 Nokia vendería los negocios relacionados con las licencias comerciales de Qt a Digia, una compañía de software y servicios también de origen finlandés, para Septiembre de 2012 todos los negocios restantes serían también vendidos a esta empresa. Digia se concentra entonces en potenciar el despliegue multiplataforma con especial énfasis en las plataformas móviles.

Ya bajo la tutela de Digia, en Diciembre de 2012 se lanza la primera versión de Qt 5, lo que presume un gran cambio en la plataforma, con gráficos acelerados por hardware, QML y Javascript tomando un rol importante. Los widgets basados solo en C++ son todavía soportados pero no se benefician de las mejoras en performance disponibles en la nueva arquitectura. Se facilita además el desarrollo de interfaces de usuario.

La popularidad de Qt ha crecido sin cesar y continúa haciéndolo hasta el día de hoy. Su éxito es el reflejo de su calidad y de cuan interesante es su uso. En la última década, Qt ha pasado de ser un producto usado por unos pocos a uno que es usado diariamente por miles de desarrolladores de código abierto en todo el mundo.

5.4.3. Javascript

JavaScript es el lenguaje de programación de la Web. La gran mayoría de los sitios web modernos usan JavaScript, y todos los navegadores de Internet modernos, en computadoras de escritorio, tabletas y teléfonos móviles inteligentes, incluyen intérpretes de JavaScript, haciendo de éste el lenguaje más omnipresente de la historia. JavaScript forma parte de la triada de tecnologías que todo desarrollador Web debe aprender: HTML, para especificar el contenido de las páginas web, CSS para definir la presentación, y JavaScript para manejar el comportamiento.

Su implementación permite a scripts en el lado cliente interactuar con el usuario, controlar el navegador, comunicarse de forma asincrónica y alterar el contenido del documento que se muestra en pantalla. También puede ser usado en el lado servidor con bibliotecas como Node.js. Fue estandarizado en la especificación del lenguaje ECMAScript.

Es un lenguaje de programación de alto nivel que se adapta a los estilos de programación orientados a objetos y funciones. El nombre JavaScript deriva de Java, aunque es diferente del lenguaje de programación Java. Su sintaxis es similar al C, y adopta nombres y convenciones de Java, sin embargo sus usos son diferentes.

Ha madurado desde sus raíces de lenguaje de scripting para transformarse en un robusto y eficiente lenguaje de propósitos generales. Sus últimas versiones definen nuevas características útiles para el desarrollo de proyectos de software de gran escala.

Técnicamente "JavaScript" es una marca registrada licenciada por Sun Microsystems (ahora Oracle) para describir la implementación de Netscape (luego Mozilla) del lenguaje. Netscape envió el lenguaje para su estandarización a ECMA (European Computer Manufacturer´s Association) y debido a cuestiones legales la estandarización del lenguaje se conoce como ECMA Script. Por razones similares la versión de Microsoft del lenguaje se conoce formalmente como JScript. En la práctica todos lo llaman JavaScript.

Durante la década pasada todos los navegadores web más populares implementaron la versión 3 de ECMAScript y no ha habido necesidad de pensar en los números de versión: el lenguaje era estable y las implementaciones de los navegadores fueron en su mayoría interoperables. Recientemente, una nueva versión del lenguaje ha sido definida como ECMAScript versión 5 y los navegadores lo han implementado.

5.4.3.1. Breve historia

JavaScript fue desarrollado originalmente por Brendan Eich mientras trabajaba para Netscape Communications Corporation, una empresa de servicios de software de EEUU, conocida principalmente por el navegador Netscape, producto en donde se pensó incluirlo agregándole capacidades de scripting. El lenguaje se llamó en un principio Mocha, luego LiveScript cuando fue lanzado en la versión beta de Netscape 2.0 y finalmente renombrado a JavaScript en 1995 con el lanzamiento del navegador Netscape 2.0B3.

En ese entonces los dos principales navegadores del mercado eran Netscape e Internet Explorer. Este último todavía no había sido integrado al sistema operativo ni era gratuito.

Es evidente que JavaScript fue bien recibido porque luego de un año fue sometido a ingeniería inversa por Microsoft y revendido como JScript para incluirlo en Internet Explorer junto con el lenguaje que la empresa esperaba fuera el competidor de JavaScript: VBScript.

Mientras VBScript tuvo éxito en otros rubros, como el lenguaje elegido para ASP y entre los productos Microsoft Office, nunca fue adoptado por la comunidad de desarrolladores web como lenguaje de scripting del lado cliente. Esto se debió especialmente a que sólo Internet Explorer lo soportaba, mientras JavaScript era soportado por todos los navegadores (aunque fuera llamado con otros nombres).

En este contexto, y quizás en un intento de diluir la marca, que Netscape envió JavaScript al comité de ECMA para ser estandarizado como ECMAScript.

JavaScript continuó creciendo en popularidad y adopción por parte de la comunidad en sintonía con la expansión y el crecimiento de Internet y el refinamiento de la experiencia de navegación del usuario. ECMA continuó lanzando nuevas actualizaciones del estándar, y los navegadores también agregaron sus propias mejoras.

La mejora más notable fue el objeto XMLHttpRequest (XHR), que llevó a la innovación de Ajax (Asynchronous JavaScript and XML). Curiosamente XHR surgió primero como un control ActiveX en Internet Explorer 5 antes de ser adoptado como un objeto JavaScript.

Con la introducción de XHR, los desarrolladores web pudieron finalmente, desde el lado cliente, enviar o recibir datos desde un servidor web de forma sincrónica o asincrónica. Esta innovación expandió el campo de acción de lo que podía hacerse con aplicaciones web, pero también incrementó la complejidad del desarrollo web en general.

Para mantenerse al ritmo de los requerimientos de alto nivel que esta nueva complejidad suponía para sus productos, los desarrolladores de navegadores de Internet se esforzaron notablemente en refinar sus propios intérpretes de JavaScript. Con la complejidad creciente,

tanto en posibilidades de la tecnología así como también en la eficiencia de los navegadores, los desarrolladores web tuvieron que actualizar sus talentos. Ahora era posible introducir contenido nuevo en las páginas web en tiempo de ejecución, pero esto también suponía nuevos problemas.

Mientras los desarrolladores web se volvían más y más adeptos al uso de JavaScript, desarrollaban bibliotecas para hacer más rápido el desarrollo encargándose de las funcionalidades de bajo nivel. Surgieron bibliotecascomo jQuery, Prototype y otras. Las mismas facilitaron el trabajo de los desarrolladores y les permitieron hacer más en menos tiempo.

5.4.4. jQuery

jQuery es una biblioteca JavaScript multiplataforma destinada a simplificar la programación de scripts del lado cliente. Actualmente es usada por más del 60% de los 10.000 sitios más visitados y es la biblioteca JavaScript más usada de la actualidad. Es libre, de código abierto y liberada bajo la licencia MIT.

Su sintaxis está diseñada para facilitar la navegación del documento web, seleccionar elementos, crear animaciones, manejar eventos, y desarrollar aplicaciones Ajax. También brinda posibilidades de crear plug-ins que corren sobre ella. Los plug-ins son extensiones de la bibliotecaque no son parte del núcleo de la misma y añaden características. Esto permite implementar una capa de abstracción de las funcionalidades de bajo nivel.

5.4.4.1. Breve historia

La biblioteca fue ideada por John Resig, un ingeniero de software y emprendedor de EEUU. Él fue el encargado de crear y liderar el desarrollo de la primera versión de jQuery.

Su creación fue anunciada por primera vez en un evento abierto y participativo conocido como BarCamp en donde se presentan nuevas aplicaciones web y tecnologías de código abierto. El lugar fue Nueva York y la edición fue la de 2006. Resig notó que las bibliotecas disponibles en este entonces podían ser ampliamente mejoradas reduciendo la “pelusa sintáctica” y agregando nuevos controles para las acciones comunes. Fue entonces que se dedicó a la creación de una nueva bibliotecaque supliera estas deficiencias.

jQuery provocó un gran impacto en la comunidad de desarrolladores y rápidamente ganó importancia. Surgieron muchos desarrolladores que se ofrecieron para mejorar la biblioteca, lo que significó el primer lanzamiento estable de jQuery, la versión 1.0, el 26 de Agosto de 2006.

Desde entonces, jQuery ha evolucionado constantemente y se ha visto la aparición de una enorme cantidad de plug-ins de la comunidad de desarrolladores.

5.4.5. PHP

PHP es un lenguaje de scripting que opera del lado servidor, es decir que consiste en el procesamiento de peticiones provenientes de un cliente permitiendo generar páginas HTML dinámicas como respuesta.

PHP funciona en la mayoría de los sistemas operativos, Unix, Linux, FreeBSD, Solaris, Windows, Mac OS X, etc. Es usado en conjunto con los servidores webs más conocidos como Apache o Microsoft IIS. Es un lenguaje de propósito general y muy flexible, no solo se limita a generar documentos HTML sino que muchos otros documentos son soportados, por ejemplo archivos PDF.

Incluye extensiones para conectarse a la mayoría de las bases de datos más conocidas como MySQL, PostgreSQL, Oracle, Sybase, etc. Crear páginas web con contenido dinámico obtenido desde bases de datos utilizando las mismas es simple.

Es considerado uno de los lenguajes más poderosos, flexibles y de alta performance por lo que ha sido elegido por sitios con gran demanda de tráfico como Facebook.

5.4.5.1. Breve historia

Antes de que PHP fuera inventado el código del lado servidor era usualmente escrito en C o Perl, lenguajes de programación generales que fueron adaptados para su uso en Internet.

El primer lanzamiento de PHP fue creado por Rasmus Lerdorf en Junio de 1995, para hacer varias tareas comunes de programación web más fáciles y menos repetitivas. El nombre originalmente significó “Personal Home Page” o “Página Web Personal”, pero desde entonces se ha convertido en un acrónimo recursivo cuyo significado es “PHP: Hypertext Preprocessor”. El objetivo de ese desarrollo era minimizar la cantidad de código requerido para desarrollar una aplicación. Estaba escrito en C, permitía reconocer ciertas etiquetas dentro del código HTML de una página web y llamar distintas funciones escritas en C, alojadas en el servidor, en función de ellas. No se podía todavía hablar de lenguaje debido a la simplicidad de este sistema.

El segundo lanzamiento de PHP ocurre en Abril de 1996 y es conocido como PHP/FI 2.0. Fue el primero en alcanzar amplia popularidad, con él es la primera vez que se habla de “scripting lenguage” o “lenguaje de scripting”, se reemplazó el código de reconocimiento de etiquetas de la versión 1 por un analizador de código que podía reconocer estructuras más complejas de etiquetas y que era más sofisticado en comparación con la versión anterior. A pesar de algunas inconsistencias en el análisis del código pudo atraer a varios adeptos.

El lanzamiento de PHP 3 en Junio de 1998 fue fundamentalmente liderado por Zeev Suraski y Andi Gutmans, de Tel Aviv, Israel quienes se ofrecieron para reescribir completamente el motor analizador de código desde el principio corrigiendo los problemas que presentaba la versión anterior. Más personas se comprometieron a trabajar en otras partes del código. Fue así que el proyecto pasó de ser el esfuerzo de una persona con unos pocos contribuyentes a un verdadero proyecto de código abierto con muchos desarrolladores alrededor del mundo. PHP 3 también facilitó la extensión del lenguaje dando a los desarrolladores la posibilidad de escribir sus propios módulos, agregando funcionalidades.

A partir de la versión 3 PHP se convirtió en un lenguaje parcialmente orientado a objetos, lo que permitió que su popularidad siguiera creciendo. Para el momento en que PHP 3 fue reemplazado, a mediados de 2000, estaba instalado en alrededor de 2,5 millones de sitios web comparado con los 250.000 en los que era usado apenas 18 meses antes. Su sucesor, PHP 4,

lanzado en Mayo de 2000 contenía numerosos cambios importantes, incluyendo el cambio a "Zend Engine" o "Motor Zend".

Zend es una compañía fundada por Zeev Suraski y Andi Gutmans para promover PHP en el ambiente corporativo. El motor que crearon trajo numerosas ventajas, al reemplazar el núcleo de PHP, el motor Zend introdujo un conteo de referencias, técnica para contabilizar la cantidad de veces que un recurso está siendo referido, para evitar derrames de memoria. Además incorporaron una capa de abstracción entre el servidor y el lenguaje, ahora PHP podía correr en el servidor web Apache, Internet Information Service de Microsoft, Zeus entre otros. También se cambió la manera en que el código PHP era ejecutado, ahora era leído una vez, convertido a un formato interno y luego ejecutado. Este nuevo paradigma de ejecución permitió el uso de caches de código externo, también conocidos como "PHP accelerators" o "aceleradores PHP", que mejoraron la performance.

El cambio de PHP 4 a PHP 5, si bien no fue tan grande en cuanto a mejoras que el de PHP 3 a PHP 4, significó también algo importante. Se mejoró mucho la orientación a objetos, se agregó el manejo de errores y excepciones con el mecanismo "try/catch", se agregó la extensión SimpleXML que permitió interactuar fácilmente con documentos XML, y se agregó también la extensión para dar soporte a SQLite, una base de datos simple usada para facilitar el despliegue de aplicaciones que necesiten acceso a datos.

La versión más reciente de PHP al momento de escribir este trabajo es la 5.6, PHP ha recorrido un largo camino hasta transformarse en la herramienta elegida por cientos de miles de sitios web del mundo para el desarrollo de código del lado servidor por su confiabilidad y performance.

5.4.6. Especificación SVG

SVG (Scalable Vector Graphics o gráficos vectoriales escalables) es un lenguaje abierto estándar usado que define la sintaxis y las características para la creación de gráficos vectoriales bidimensionales estáticos y animados con soporte para animación interactiva y dinámica.

La primera especificación de SVG vio la luz a fines de 2001. Fue escrita por el World Wide Web Consortium (W3C), un consorcio internacional que ha escrito decenas de recomendaciones para la WWW, como por ejemplo HTML, HTTP, PNG, XML, entre muchas otras.

Se admiten tres tipos de elementos gráficos: Formas vectoriales (conformadas por líneas rectas y curvas), imágenes y texto. Los elementos gráficos pueden ser agrupados, estilizados, transformados y compuestos.

Los gráficos SVG pueden ser interactivos y dinámicos. Es posible definir animaciones que pueden ser disparadas ya sea declarativamente dentro del mismo archivo o mediante scripts.

El uso de lenguajes de scripting da la posibilidad acceder el documento SVG lo que permite realizar aplicaciones sofisticadas. Se pueden controlar los elementos, atributos y propiedades. Se provee además un conjunto de manejadores de eventos, como "onmouseover" cuando el

mouse pasa sobre el elemento o “onclick” cuando se hace click en el elemento. Estos pueden ser asignados a cualquier objeto gráfico SVG.

Está basado en el formato XML (Extensible Markup Language) el cual es un lenguaje de marcas utilizado para almacenamiento de datos en forma legible desarrollado por el W3C (World Wide Web Consortium).

A diferencia de los formatos de mapa de bits la imagen vectorial puede ser escalada de forma arbitraria sin perder calidad de imagen. Por su parte siendo su contenido XML, el mismo se compone de varios fragmentos de texto repetidos, por lo que es apropiado para algoritmos de compresión sin pérdida de información. Esto lo hace ideal para transmisiones vía internet.

Hoy en día todos los navegadores de Internet más populares pueden interpretar este lenguaje cuando cargan una página web que contenga código SVG sin necesidad de instalar complementos ni software adicional.

Existen muchos programas de diseño gráfico para la edición de este tipo de archivos. El más popular de todos ellos es Inkscape, un programa de código abierto, con una activa comunidad de desarrolladores y de amplia difusión mundial. El mismo será usado para la edición de pantallas del sistema.

6. Capítulo VI: Conceptos del sistema

El presente capítulo pretende definir una serie de conceptos que han sido forjados durante el desarrollo del sistema y que serán mencionados y utilizados durante el resto del presente trabajo. Los mismos están relacionados con diferentes áreas, funciones y características y ayudan a comprender globalmente el comportamiento del sistema.

6.1. Elementos de adquisición de datos

Existen cuatro conceptos básicos definidos en el sistema desde el punto de vista de la adquisición de datos.

- ✓ **Estación**
  Station

- ✓ **Dispositivo**
  Device

- ✓ **Canal de comunicación**
  Communication Channel

- ✓ **Grupo de escaneo**
  Scan Group

6.1.1. Estación

Símbolo del elemento: 

La estación es un concepto que no tiene una función técnica sino organizativa. Es un elemento que sirve al desarrollador del sistema para clasificar o agrupar a los dispositivos de acuerdo a factores tales como:

- ✓ **Su ubicación geográfica**
  Posición física de los dispositivos dentro de una planta industrial. Por ejemplo “Línea de producción 1”, “Sala de comunicaciones”, “Sala de bombas”, etc.
- ✓ **Su naturaleza o tipo**
  La característica, el modelo, la marca de los dispositivos. Por ejemplo “PLC Schneider”, “RTU Siemens”, “Analizador de red Elster”, etc.

- ✓ **Su función**
  Tarea llevada a cabo por los dispositivos. Por ejemplo “Sensores de presión”, “Medidores de temperatura”, “Controladores de velocidad”, etc.

Queda a elección del usuario el criterio de agrupación que aplicará a los dispositivos del sistema.

6.1.2. Dispositivo

Símbolo del elemento: 

Concepto que describe a un elemento de adquisición de datos, es decir un elemento capaz de tomar muestras de información, provenientes de un ambiente donde se produce un fenómeno físico bajo estudio, y del que se desea obtener un conjunto de señales analógicas para acondicionarlas, digitalizarlas y transmitirlas para su posterior procesamiento a un ente de jerarquía superior, en nuestro caso el sistema SCADA.

Son equipos de esta naturaleza los PLC, las RTU, los medidores o sensores inteligentes de magnitudes físicas y cualquier otro elemento o dispositivo de campo que tome información de un proceso físico y que posea capacidades de comunicación. La conexión con el sistema se producirá utilizando uno o más **Canales de comunicación** y será interrogado periódicamente en base a ciertas pautas definidas en los **Grupos de Escaneo.**

6.1.3. Canal de comunicación

Símbolo del elemento: 

Es el elemento que define el medio de transmisión entre el sistema SCADA y los dispositivos definidos. El medio puede ser cables de cobre, ondas de radio, fibra óptica, etc.

Los canales se conectan a la computadora donde funciona el sistema ya sea mediante puertos seriales (tipo EIA/TIA RS232C o EIA RS485) o puertos Ethernet.

La conexión a través de los Canales de Comunicación sigue el modelo cliente/servidor, en donde el sistema SCADA envía peticiones o interrogaciones a un cierto **Dispositivo** (especificando una dirección de hardware a la que debe ir dirigido el paquete) respondiendo éste luego con la información requerida. Cada intercambio de información se inicia siempre desde el lado del sistema SCADA y no desde el lado de los Dispositivos.

Cada canal es asignado a uno o más **Dispositivos**. Cuando un canal es compartido se suele utilizar en configuraciones del tipo RS485 o Multidrop en donde los **Dispositivos**, al compartir el medio de transmisión, deberán tener direcciones de hardware distintas.

6.1.4. Grupos de Escaneo

Símbolo del elemento: 

El Grupo de Escaneo define un bloque de información que será leído desde un **Dispositivo** a intervalos regulares. Un mismo Grupo de Escaneo puede ser compartido por varios Dispositivos si es que se desea adquirir la misma información de ellos, en este caso si bien se trata del mismo Grupo de Escaneo cada Dispositivo manejará la información leída de forma independiente sin ocurrir interferencia entre ellos.

Hay cuatro factores fundamentales que definen a un Grupo de Escaneo:

- ✓ **Tipo de dato a interrogar**
  La naturaleza de la información que se desea obtener del **Dispositivo**. De forma genérica se trata de registros entrada o salida tanto analógicos como digitales. Los registros podrán ser por ejemplo de 1 o 2 bits de longitud para los digitales y de 16 o 32 bits de longitud para los analógicos.
- ✓ **Rango**
  Los **Dispositivos** internamente almacenarán una determinada cantidad de registros de uno o más tipos. Cada uno de estos registros posee un número de orden que lo diferencia del resto. Cuando el sistema envía una interrogación especifica un número de orden inicial y uno final. El **Dispositivo** deberá responder con todos los registros cuyo número de orden se encuentre dentro del intervalo especificado. Se conoce a este intervalo como rango de interrogación.
- ✓ **Periodicidad**
  Determina el tiempo que el sistema esperará antes de enviar una nueva petición o interrogación de este tipo.
- ✓ **Protocolo**
  El protocolo de telecontrol con el cual el sistema enviará la interrogación. Esto no modifica el contenido o significado del mensaje sino el "lenguaje" con el que se transmite al **Dispositivo**.

6.1.5. Relación entre elementos

Los elementos definidos se vinculan entre sí mediante asignaciones, así una **Estación** tiene asignados uno o más **Dispositivos**, y a su vez los **Dispositivos** tienen asignados uno o más **Canales de comunicación** y **Grupos de Escaneo**.

Algunas consideraciones:

- ✓ No hay límite en cuanto al número de **Dispositivos** que pueden asignarse a una **Estación**.
- ✓ No hay límite en cuanto número de **Canales de Comunicación** que pueden asignarse a un **Dispositivo**. De esta forma se tiene multiplicidad de caminos, de forma que de producirse errores en un canal pueden existir alternativas para vincularse al mismo.
- ✓ No hay límite en cuanto número de **Grupos de Escaneo** que pueden asignarse a un **Dispositivo**.
- ✓ Las **Estaciones** están al tope de la jerarquía y por lo tanto no se asignan a ningún otro elemento.
- ✓ Un **Grupo de Escaneo** puede ser compartido por varios **Dispositivos**.
- ✓ Un **Canal de Comunicación** puede ser compartido por varios **Dispositivos**.

- ✓ Un **Dispositivo** no puede ser compartido por varias **Estaciones**.

El diagrama con la relación entre los elementos se conoce como "**Árbol de Elementos**", a continuación un ejemplo de éste:

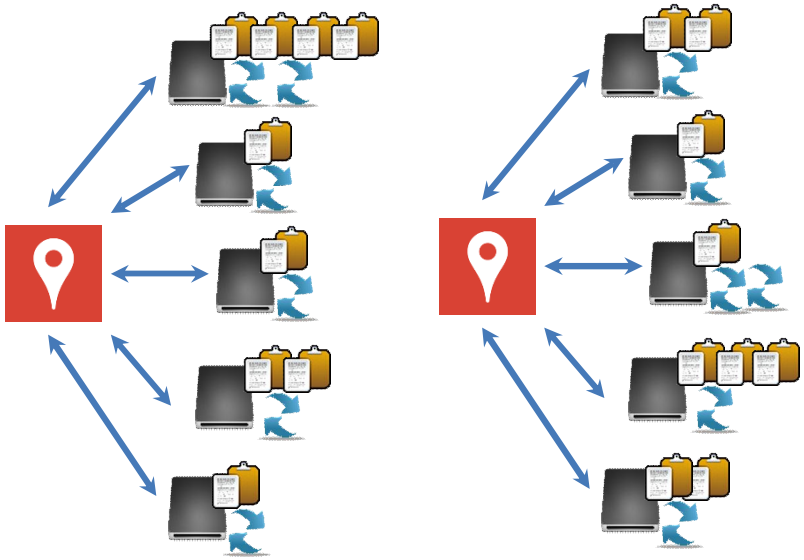


Figura 52 – Representación gráfica del Árbol de Elementos

Puede verse en el esquema dos **Estaciones**, las cuales tienen asignados cinco **Dispositivos** cada una. Por su parte cada dispositivo tiene asignado **Grupos de Escaneo** y **Canales de comunicación**.

En el módulo de configuración y desarrollo del sistema existe una pestaña llamada "**Árbol de elementos**" desde donde pueden definirse los distintos elementos, establecer sus parámetros de funcionamiento y realizar asignaciones entre los mismos.

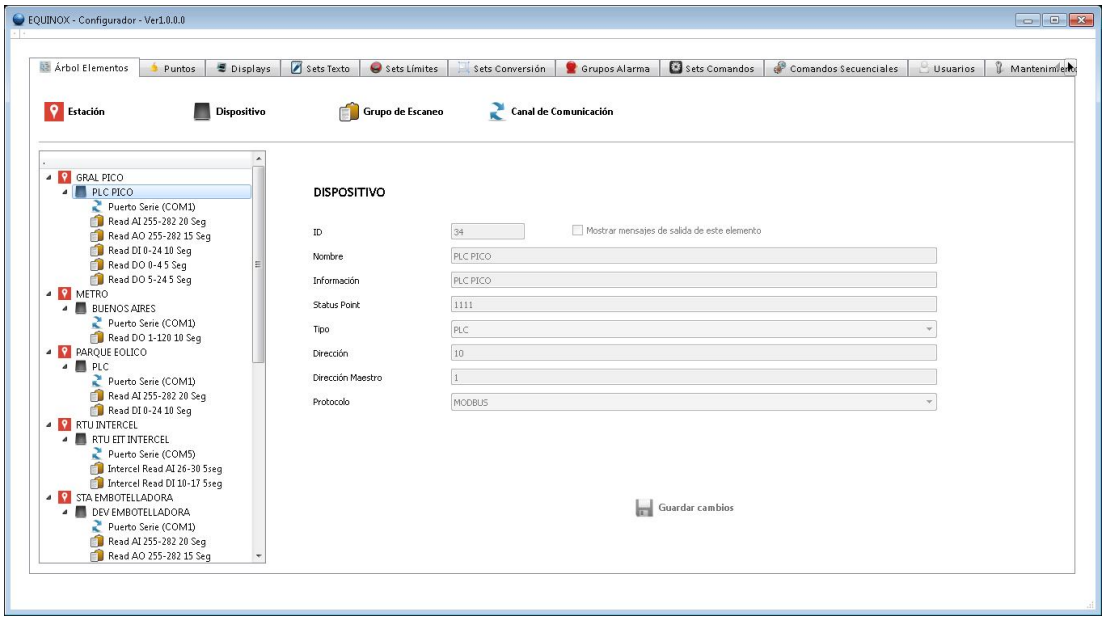


Figura 53 – Captura de pantalla de la sección "Árbol de Elementos" del Módulo de Configuración y Desarrollo del sistema

6.2. Mapa de datos del Dispositivo

El módulo principal del sistema (Core) luego de cargar la configuración del sistema en el arranque reserva un área de memoria por cada Dispositivo. Esta área de memoria se denomina **Mapa de Datos**. Su función es almacenar una copia local del estado de los registros, entradas y salidas digitales almacenadas en la memoria del Dispositivo.

Se definen seis tipos de datos diferentes, cada uno de ellos ocupará una porción del mapa de datos. Éstos son los siguientes:

6.2.1. Registro de Entrada 16 Bits / Registro de Salida 16 Bits

Símbolos: 

Son registros de entrada o salida con una longitud de 16 bits, es decir de dos bytes. Pueden asociarse a símbolos del tipo "**Status**", "**Measure**", "**Bar**" o "**Gauge**".

- ✓  8 Bits LSB
  Se aplica una máscara para tomar la parte baja (byte menos significativo) del registro de 16 bits.
- ✓  8 Bits MSB
  Se aplica una máscara para tomar la parte alta (byte más significativo) del registro de 16 bits.
- ✓  16 Bits
  Se toma el registro entero de 16 bits sin aplicar máscaras.
- ✓  16 + 16 Bits

Se toman dos registros consecutivos de 16 bits, por lo que se considerará un registro doble de 4 bytes de longitud.

- ✓  Bit 0 – Bit 15
  Se selecciona uno de los 16 bits de los dos bytes del registro.

El último ítem podrá asignarse a símbolos "status". El resto podrán asignarse a símbolos "measure", "bar", o "gauge".

6.2.2. Registro de Entrada 32 Bits / Registro de Salida 32 Bits

Símbolos: 

Son registros de entrada o salida con una longitud de 32 bits, es decir de cuatro bytes. Pueden asociarse a símbolos del tipo "**Status**", "**Measure**", "**Bar**" o "**Gauge**".

- ✓  8 Bits B0
  Se aplica una máscara para tomar el primer byte (menos significativo) del registro de 32 bits.
- ✓  8 Bits B1
  Se aplica una máscara para tomar el segundo byte del registro de 32 bits.
- ✓  8 Bits B2
  Se aplica una máscara para tomar el tercer byte del registro de 32 bits.
- ✓  8 Bits B3
  Se aplica una máscara para tomar el cuarto byte (más significativo) del registro de 32 bits.
- ✓  16 Bits Low
  Se aplica una máscara para tomar el primer y segundo byte del registro de 32 bits, formando un subregistro de 16 bits.
- ✓  16 Bits High
  Se aplica una máscara para tomar el tercer y cuarto byte del registro de 32 bits, formando un subregistro de 16 bits.
- ✓  32 Bits
  Se toma el registro entero de 32 bits sin aplicar máscaras.
- ✓  Bit 0 – Bit 31
  Se selecciona uno de los 32 bits de los cuatro bytes del registro.

El último ítem podrá asignarse a símbolos "status". El resto podrán asignarse a símbolos "measure", "bar", o "gauge".

6.2.3. Entrada Digital/ Salida Digital

Símbolos: 

Son puntos de naturaleza discreta, es decir que toman un número finito de estados posibles. Pueden asociarse sólo a símbolos del tipo "**Status**".

- ✓ 0 Bit único
  Tienen un bit de longitud, por lo tanto pueden tomar sólo dos estados: "0" o "1", correspondiendo cada uno de ellos a un "abierto" / "cerrado" de una válvula, a un "encendido" / "apagado" de un interruptor, etc.
- ✓ 0 1 Doble bit
  Tienen dos bits de longitud, por lo tanto pueden tomar cuatro estados: "0", "1", "2", o "3". Un ejemplo de uso puede ser un motor con cuatro velocidades posibles, o un seccionador que puede estar tanto "abierto", "cerrado" como "en tránsito".

6.3. Sets de Límites

Los puntos del sistema relacionados con registros de naturaleza analógica, tanto de entrada como de salida, pueden ser asignados a un Set de Límites. Cada Set de Límites define un conjunto de seis valores numéricos (tres límites superiores y tres inferiores) que serán utilizados para informar al usuario mediante la generación de alarmas de situaciones en las que el valor del registro exceda o deje de exceder a cada uno de ellos. Las alarmas se generarán si el punto correspondiente está habilitado para ello.

Así las temperaturas, los niveles, las presiones, las velocidades y cualquier otra magnitud de naturaleza analógica pueden ser analizadas con esta herramienta para que el operador del sistema pueda tomar conocimiento de situaciones críticas que supongan la violación de valores límites de las mismas.

6.4. Sets de Texto

Los Sets de Texto son conjuntos de frases o palabras a definir por el desarrollador del sistema que permiten identificar claramente y de una forma amena para el usuario los distintos valores que pueden tomar los puntos de estado o discretos.

Estos puntos, que tienen uno o dos bits de longitud, pueden tomar un número finito de valores numéricos (dos o cuatro según el caso). Para cada de estos valores se asigna una denominación que ayudará al operador del sistema a determinar con cual situación se corresponde el valor numérico actual.

Para ejemplificarlo supongamos que deseamos determinar el estado actual de un motor eléctrico, el cual puede estar encendido o apagado. El motor posee un contacto seco de salida el cual se cierra cuando el motor está encendido y permanece abierto cuando está apagado. Este contacto seco se encuentra conectado a una entrada digital de un dispositivo de adquisición de datos. El motor apagado representará un "0" lógico en la entrada digital mientras que el motor encendido representará un "1".

Esta información será transmitida posteriormente al sistema SCADA a través de un vínculo de comunicación y de protocolo de telecontrol como MODBUS.

Se asignará entonces al punto del sistema representante del motor un Set de Texto acorde que contenga las palabras "ENCENDIDO" y "APAGADO" las cuales serán presentadas al operador del sistema pudiendo determinar más claramente el estado del motor que si se mostrara "0" y "1".

Los Sets de Texto pueden también utilizarse para la definición de las palabras que se mostrarán ante una violación de límite de acuerdo al Set de Límite asignado al punto. Dado que existen seis límites por set (tres superiores y tres inferiores) y que el valor del punto puede exceder o dejar de exceder cada uno de ellos necesitaremos en total 2 x 6 = 12 palabras para representarlos, a saber:

- ✓ Exceso Límite Inferior 1, Exceso Límite Inferior 2, Exceso Límite Inferior 3.
- ✓ Exceso Límite Superior 1, Exceso Límite Superior 2, Exceso Límite Superior 3.
- ✓ Retorno Exceso Límite Inferior 1, Retorno Exceso Límite Inferior 2, Retorno Exceso Límite Inferior 3.
- ✓ Retorno Exceso Límite Superior 1, Retorno Exceso Límite Superior 2, Retorno Exceso Límite Superior 3.

6.5. Sets de Conversión

Permite convertir los valores de los puntos de naturaleza analógica a unidades de ingeniería.

Los puntos son transmitidos al SCADA desde los dispositivos de adquisición de datos como cuentas o valores numéricos sin una unidad o magnitud definida. Así por ejemplo un registro de dos bytes podrá tener un valor decimal que variará desde 0 hasta 65.535. Sin embargo este valor por sí solo no nos dice nada, es necesario convertirlo a una unidad de ingeniería que nos hable de 20 metros, 17 grados centígrados, 1926 RPM, 73 Km/hora, etc.

Para ello se emplea un Set de Conversión que no es otra cosa que una recta en donde se define una pendiente y una ordenada al origen, luego el valor crudo sin unidad procedente del registro se ingresará en la fórmula de la recta como la abscisa "x" y se calculará la ordenada "y" que corresponderá al valor del punto en su magnitud correspondiente. Por ejemplo:

$$\mathbf{y}\,[\text{metros}] = m \,.\, \mathbf{x} + b$$

Siendo "m" la pendiente y "b" la ordenada al origen, luego si el valor decimal del registro es "123", la pendiente "2" y la ordenada al origen "5", podemos calcular que el valor actual del punto deberá ser "251".

6.6. Grupos de Alarma

Los cambios de estado en los puntos del sistema pueden generar o no alarmas. Esto se define en la configuración de cada uno de ellos. Cuando se define que un punto genere una alarma es porque se desea alertar sobre situaciones críticas que requieren la atención y posiblemente la intervención del operador del sistema.

Con el objeto de clasificar las alarmas de acuerdo a la criticidad, el área o dispositivo al cual pertenecen o el tipo de señal que representan se definen en el sistema Grupos de Alarma. Cada uno de ellos asignado a uno o más puntos.

Por su parte se establece un color que será el utilizado para colorear las líneas de texto correspondientes a las alarmas de este grupo dentro del Listado de Alarmas del sistema.

Además así como un Grupo de Alarma está relacionado con un conjunto de puntos también puede relacionarse con un conjunto de usuarios a los cuales se les avisará por correo electrónico ante la generación de una alarma perteneciente al grupo.

6.7. Sets de Comandos

Un **Set de Comandos** es un conjunto de telemandos que se enviarán a un cierto dispositivo a través de un canal de comunicación cuando un operador del sistema lo desee. Pueden asignarse a uno o varios puntos dándole la posibilidad al usuario activarlos cuando haga clic sobre el símbolo gráfico correspondiente.

Los telemandos o comandos son órdenes enviadas a los dispositivos conectados al sistema con el objetivo de realizar una acción sobre los mismos. Las acciones pueden ser, por ejemplo, alterar el valor de una dirección de memoria o cambiar el estado de salidas digitales o analógicas.

Existen dos tipos de telemandos que pueden definirse dentro de un Set de Comandos: Los Comandos Simples permiten realizar cambios de salidas digitales del dispositivo, por su parte los Comandos Complejos Secuenciales son ejecuciones secuenciales de uno o más telemandos complejos.

Los Comandos Complejos se utilizan para alterar el estado de bits en registros de salida o para establecer su valor en función de otros puntos del sistema o de valores fijos, o para cambiar de estado varias salidas digitales.

Cada Set de Comandos contiene una combinación de Comandos Simples y Comandos complejos secuenciales aunque podría ser un conjunto de sólo uno de ellos.

Los Sets de Comandos utilizan un Set de Texto pero solo para nominar cada uno de los telemandos que lo integran. Así por ejemplo se le presentarán al operador las opciones de “ARRANQUE” y “PARADA” relativos a una bomba y ambos estarán relacionados ya sea a un Comando Simple o a un Comando complejo secuencial.

6.8. Puntos

En los sistemas SCADA los puntos son unidades de información leídas desde los dispositivos de control definidos en el sistema. Hay una relación entre cada punto creado y un registro de entrada o salida de algún dispositivo conectado. La indicación del punto será actualizada periódicamente para ser coherente con el valor del registro.

A su vez los puntos del sistema son los encargados de conectar esta información con su contraparte gráfica, es decir el símbolo gráfico que lo represente. La relación entre punto y símbolo se establece mediante dos cadenas de texto llamadas “Nombre” y “Sub-nombre”, de esta cuando punto y el símbolo coincidan en su denominación el cambio en la indicación del punto producirá una alteración de la representación gráfica del símbolo.

Por su parte los puntos son los usuarios de los Sets de Conversión, Sets de Límites, Grupos de Alarma, Sets de Texto y Sets de Comando:

- ✓ El Set de Conversión se utilizará para convertir el valor del registro desde el que se toma la información de punto a su indicación.
- ✓ El Set de Límites se utilizará para producir las alarmas correspondientes cuando la indicación del punto viole los límites establecidos en el set.
- ✓ El Grupo de Alarma será usado para categorizar las alarmas producidas por el punto, para establecer el color del texto en la lista de alarmas y para determinar que usuarios deben ser alertados ante cambios que generen alarmas.
- ✓ El Set de Texto será utilizado para determinar que texto deberá ser presentado en pantalla junto con las alarmas para cada indicación posible en los puntos digitales.
- ✓ El Set de Comandos, cuando esté habilitado su uso, determinará la acción que deberá llevarse a cabo cuando el operador del sistema intente, a través del símbolo gráfico, ejecutar un comando.

7. Capítulo VII: Arquitectura del sistema

7.1. Diagrama funcional

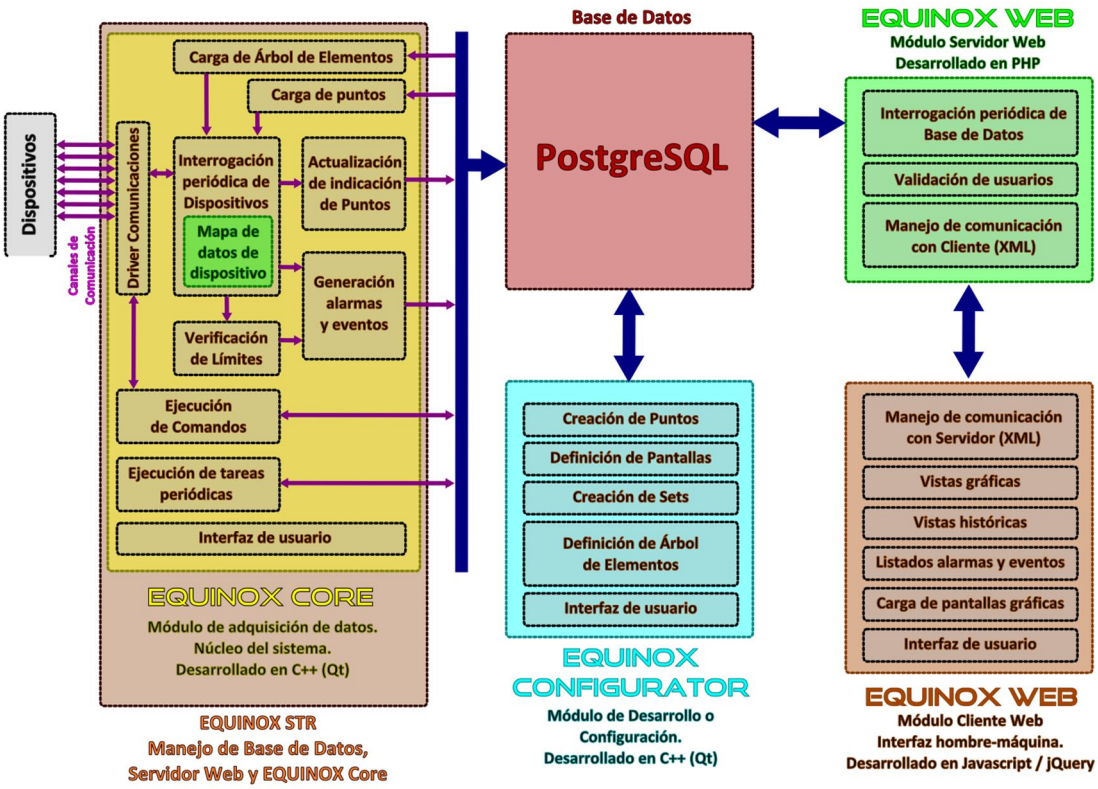


Figura 54 – Diagrama funcional del sistema

El sistema se conforma de tres módulos funcionales fundamentales:

- ✓ Módulo SCADA o Núcleo (**EQUINOX Core**).
- ✓ Módulo de Configuración y Desarrollo (**EQUINOX Configurator**).
- ✓ Módulo HMI (**EQUINOX Web**).

Los módulos funcionan de forma independiente e intercambian información entre ellos mediante la base de datos del sistema y mediante sockets TCP/IP. El desarrollo de todos los módulos del sistema fue llevada a cabo utilizando diversos lenguajes de programación pero todos ellos fueron elegidos teniendo en cuenta que fueran orientados a objetos.

La arquitectura planteada brinda la flexibilidad de distribuir los módulos funcionales en nodos físicos de acuerdo al requerimiento de la aplicación. Para aplicaciones de menor envergadura

todos los módulos podrán instalarse en un solo servidor, ahorrando costos de hardware. Sin embargo, de ser necesaria una mayor potencia computacional global debido a las características de la aplicación, entonces los módulos podrán distribuirse en varios servidores sin mayores cambios en la configuración.

Así el motor de base de datos podrá residir en un servidor, el módulo HMI en otro y por último el módulo SCADA en un tercero. De esta forma estamos separando tres funciones claramente diferentes, dedicando para cada una de ellas un hardware específico.

7.2. Descripción de módulos

7.2.1. Módulo SCADA o Núcleo

El Módulo SCADA o Núcleo del sistema, también conocido como "**Core**" es, como su nombre lo indica, el principal y de mayor importancia. Si lo comparamos en cuanto a sus funciones con otros sistemas SCADA su equivalente sería el módulo conocido como "runtime".

Su función fundamental es la de interrogar secuencialmente y periódicamente a todos los dispositivos de control configurados en el sistema, de acuerdo a pautas definidas que especifican el tipo y cantidad de información que se desea obtener de cada uno de ellos.

Para el desarrollo de este módulo se eligió como lenguaje de programación C++, asimismo dentro de los entornos de desarrollos disponibles en este lenguaje se eligió a la biblioteca Qt, que ya fue expuesta, debido a su potencialidad y confiabilidad.

7.2.1.1. Principio de funcionamiento

El módulo Core basa su funcionamiento en la creación de múltiples hilos de ejecución o threads. La tecnología multi-hilo o multithreading es una característica común a la mayoría de los sistemas operativos modernos que permite a un único proceso manejar pedidos múltiples simultáneamente mediante la creación de varios hilos de ejecución.

El sistema operativo maneja estos hilos de forma independiente mediante su planificador de tareas y un mecanismo de división por multiplexado de tiempo, también conocido como multitasking. La conmutación ocurre a una velocidad tan elevada que el usuario percibe que la ejecución de los mismos se efectúa de forma simultánea.

Esto evita la necesidad de tener varias copias del mismo proceso corriendo al mismo tiempo. Es una función soportada por la gran mayoría de las plataformas de hardware recientes donde corren los sistemas operativos más habituales.

El módulo núcleo del sistema se aprovecha de esta funcionalidad creando un hilo de ejecución por cada dispositivo definido en la base de datos. De esta forma se logra que cada uno de ellos se maneje de forma independiente de los demás, además de estructurar de una forma más organizada el código fuente de la aplicación.

Cada dispositivo, asociado unívocamente a un hilo de ejecución, maneja su propio mapa de datos, realiza sus propias interrogaciones, verifica límites de los valores leídos, genera sus alarmas y eventos.

7.2.1.2. Funciones

Carga de configuración

El módulo en su arranque lee la configuración del sistema desde la base de datos. Esto incluye la carga del Árbol de Elementos incluyendo las Estaciones, los Dispositivos, los Canales de Comunicación, los Grupos de Escaneo, luego se cargan los Sets de Límites y de Conversión, y

por último el listado de puntos asignado a cada uno de los Dispositivos, información a partir de la cual se construye el Mapa de Datos.

Interrogación de Dispositivos

A partir de la carga de Grupos de Escaneo se establece una secuencia de interrogación que se repetirá indefinidamente y de forma periódica con el objeto de obtener información actualizada del estado actual de los registros (entradas y salidas digitales y analógicas) de cada Dispositivo. La interrogación se realiza respetando estrictamente los parámetros definidos desde la utilidad de configuración, esto comprende el período en segundos, el tipo de dato a interrogar, el rango (índice o registro de comienzo y de finalización) y el protocolo de telecontrol con el cual se realizará la pregunta o petición.

Las respuestas de las interrogaciones son analizadas y se utilizan para refrescar el Mapa de Datos de cada Dispositivo almacenado en la memoria de la aplicación. Previamente se compara el estado de cada punto con la interrogación anterior en busca de cambios, se analizan violaciones de límites y si fuera necesario se generan las alarmas y eventos necesarios que reflejen eventuales cambios en los puntos afectados. Posteriormente toda esta información es impactada en la base de datos del sistema.

Ejecución de comandos

De forma periódica el módulo verificará en la base de datos del sistema si existen comandos pendientes de ejecución, en caso encontrarse telemandos para realizar el módulo los analizará y enviará la acción correspondiente al Dispositivo especificado en los mismos, indicando en el paquete de datos enviado el tipo de dato que se desea alterar y el índice o registro sobre el que se quiere actuar.

Driver de comunicaciones

El driver de comunicaciones se encarga de convertir los mensajes de interrogación y de comandos al protocolo de telecontrol con el cual se dialoga con el Dispositivo. Funciona como una capa de abstracción entre los mensajes internos del sistema y su adaptación una trama de bytes apropiada de acuerdo al protocolo seleccionado.

Registro de históricos

Este módulo se encarga además de almacenar cada un intervalo definido todas las indicaciones de los puntos. Este intervalo se encuentra especificado en un archivo de configuración externo, de manera predeterminada es 10 minutos. De esta forma cada 10 minutos el sistema toma una fotografía de la indicación actual todos los puntos definidos y almacena la información recabada en una tabla especial dentro de la base de datos. Esta información puede ser posteriormente consultada desde la interfaz de operación.

Ejecución de tareas periódicas

Existen en el sistema una serie de tareas periódicas destinadas a realizar distintas tareas de mantenimiento sobre el sistema, lo que incluye limpieza de datos antiguos en la base de datos,

el respaldo preventivo de la configuración del sistema, el borrado de archivos viejos del sistema de archivos, etc. Estas tareas son ejecutadas cada un tiempo determinado por este módulo siguiendo los lineamientos definidos en la base de datos del sistema.

Para definir los instantes en que cada tarea debe ser ejecutada se utilizó un formato similar a la tabla del comando "cron" de Unix, en donde se especifica el mes, el día del mes, el día de la semana, la hora y el minuto en que cada tarea debe ser ejecutada.

Interfaz de usuario

El módulo es una aplicación de consola por lo que su salida es sólo texto, no hay una interacción con el usuario sino que simplemente se imprimen mensajes de salida. El texto se muestra dentro de otra aplicación que se describe a continuación.

Manejo de Base de Datos, Servidor Web y arranque de módulo núcleo

Existe una aplicación separada, denominada "STR" que se encarga del arranque y parada de los siguientes programas:

- ✓ Motor de Base de Datos (PostgreSQL).
- ✓ Servidor Web (Nginx).
- ✓ Servidor PHP.
- ✓ Módulo Núcleo.

Existe una secuencia en que los programas deben ser arrancados y parados. Por ejemplo, antes de lanzar el Módulo Núcleo, es necesario que el Motor de Base de Datos encuentre listo para operar. Asimismo para que los operadores puedan conectarse desde el Módulo Web es necesario que tanto la Base de Datos como los Servidores Web y PHP se encuentren en operación. También existe un orden establecido para la parada del sistema, en donde la Base de Datos siempre es la última en dejar de funcionar.

La comunicación se realiza a través de la ejecución de ciertos archivos ejecutables pertenecientes a los programas involucrados y se analiza la salida de texto de cada uno de ellos para determinar en qué condición se encuentran en cada momento. En el caso del módulo núcleo, una vez arrancada la aplicación, la comunicación se realiza mediante un socket TCP/IP en donde se envían órdenes específicas que van desde el arranque o la parada hasta colectar el estado de operación actual de los Dispositivos y Canales, pausar las interrogaciones momentáneamente, etc.

Esta aplicación se desarrolló con una interfaz gráfica de usuario desde donde se pueden comandar las acciones principales del sistema, como reiniciar, pausar o verificar el estado de los elementos, también se puede lanzar el Módulo de Configuración y Desarrollo, etc.

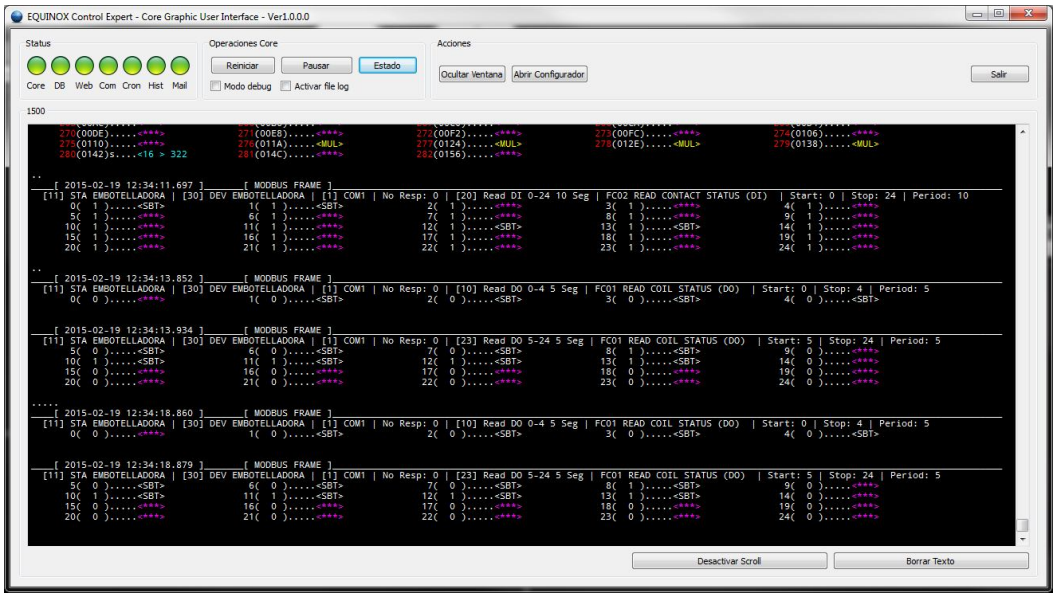


Figura 55 – Captura de pantalla del Módulo Núcleo y su manejador "STR"

En la Figura 55 puede verse una imagen de la ventana gráfica del Módulo Núcleo y su manejador "STR". Existe un recuadro denominado "Status" donde se especifica mediante luces verdes/amarillas/rojas el estado del Módulo Núcleo ("Core"), del Motor de Base de Datos ("DB"), del Servidor Web ("Web"), de la comunicación entre la aplicación STR y el Módulo Núcleo ("Com") y el estado del Historiador ("Hist") y del envío de mensajes de alerta ("Mail").

También existen operaciones de reinicio, pausa, verificación de estado de elementos, activación del modo de depuración (debug) y del registro de mensajes.

Por su parte el cuadro de texto central corresponde a la salida de mensajes de texto de la aplicación de consola del Módulo Núcleo, este texto puede recorrerse con las barras laterales, se puede desactivar su desplazamiento o vaciarlo.

7.2.2. Módulo de Configuración y Desarrollo

El Módulo de Configuración y Desarrollo, también conocido como "**Configurator**" permite al desarrollador del sistema definir todos los parámetros de configuración necesarios. Fue creado pensando en darle una completa libertad al usuario para realizar todo tipo de cambios sin necesidad de recurrir al proveedor. Además se puso especial énfasis en lograr una herramienta intuitiva, fácil de entender y que pueda ser utilizada por personas con conocimientos reducidos sobre sistemas HMI/SCADA.

Al igual que para el Módulo Núcleo se eligió como lenguaje de programación C++ y la biblioteca Qt.

7.2.2.1. Funciones

Definición de Árbol de Elementos

Esta sección permite la creación del Árbol de Elementos del sistema, es decir la definición de los elementos de adquisición de datos: Estaciones, Dispositivos, Canales de Comunicación y Grupos de Escaneo y relacionarlos entre sí para reflejar la arquitectura de la red de elementos conectados al sistema SCADA.

Para cada elemento existen configuraciones particulares:

- ✓ **Estaciones**: Al ser un elemento puramente organizativo sólo se establece una denominación y un campo informativo sólo para referencia del usuario.
- ✓ **Dispositivos**: Se define un nombre, un campo informativo, un número de punto que se utilizará para representar cambios de estado del dispositivo (en línea, fuera de línea, no disponible, etc.), una dirección de hardware, la dirección de hardware del maestro (el propio sistema en este caso) y el protocolo de telecontrol a utilizar.
- ✓ **Canales de Comunicación**: Al igual que en los Dispositivos se define un nombre, un campo informativo, un punto usado para reflejar los cambios de estado (en línea, fuera de línea, no disponible, etc.), el tipo de puerto que puede ser serial o TCP/IP, la dirección del puerto (COM1, dirección IP, etc.), la velocidad aplicable sólo a puertos seriales, parámetros adicionales del puerto, el timeout del canal y el timeout entre bytes.
- ✓ **Grupos de Escaneo**: Las definiciones incluyen un nombre, el protocolo de telecontrol con el cual serán utilizados (Modbus, DNP, etc.), el tipo y el subtipo de dato a interrogar, el registro o índice de comienzo y finalización y el período de interrogación.

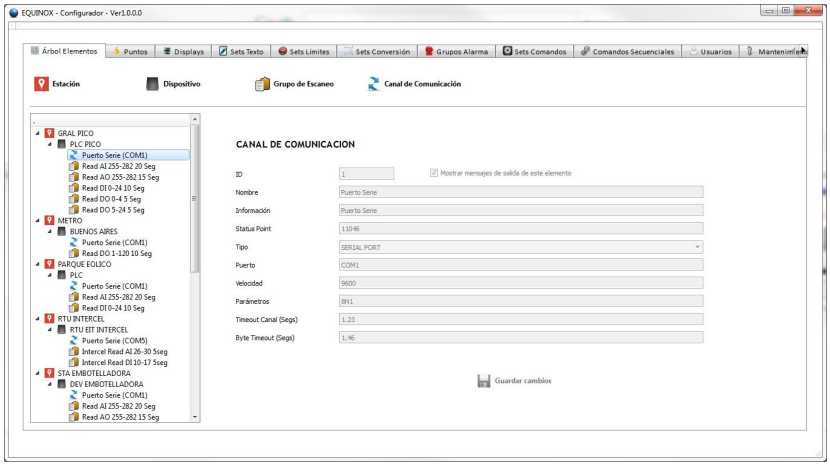


Figura 56 – Captura de pantalla de la sección Árbol de Elementos

Definición de Pantallas del Sistema

Este módulo define además las pantallas del sistema. En principio se les asigna un nombre, una denominación para la pestaña con la que aparecerán en el Módulo Web y una descripción con propósito puramente informativo para el usuario. Además es posible cambiar el color de

fondo, activar la posibilidad de hacer zoom sobre la misma y activarla o desactivarla en función de si se desea que la pantalla se muestre o no a los operadores del sistema. Por último se debe seleccionar un archivo gráfico vectorial (formato SVG), creado con anterioridad con el programa de edición Inkscape, que será mostrado en pantalla al seleccionar la pantalla correspondiente.

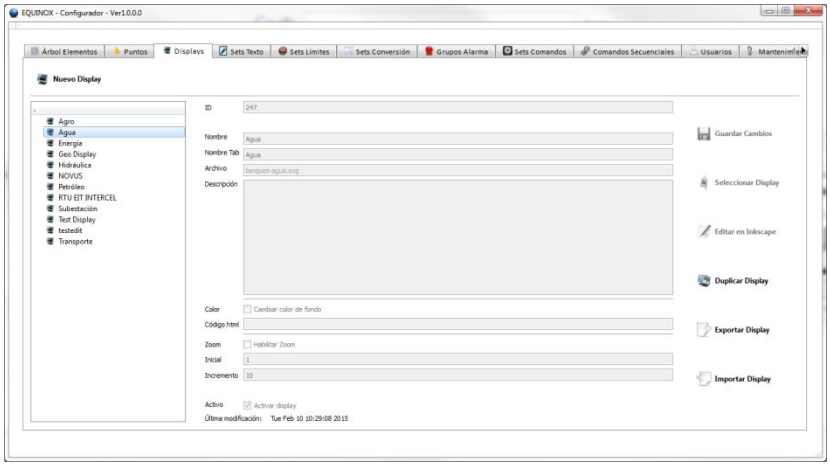


Figura 57 – Captura de pantalla de la sección de Displays

Definición de Sets de Texto

La creación de Sets de Texto permite la asignación de frases o palabras para cada estado posible de los puntos de estado o discretos. Ej: "0" equivale a "CERRADO", "1" equivale a "ABIERTO". Luego cada Set puede ser asignado a ciertos puntos del sistema. En su creación se debe definir un nombre, que permite su identificación posterior, y un listado de pares valor-texto.

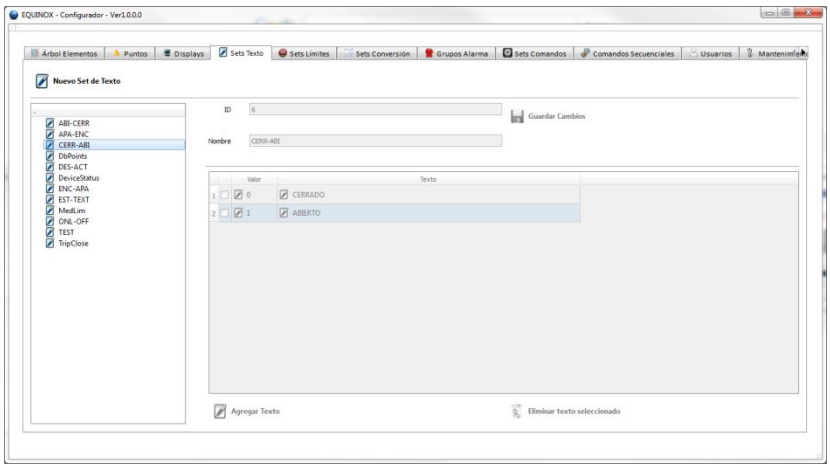


Figura 58 – Captura de pantalla de la sección Sets de Texto

Definición de Sets de Límites

La creación de Sets de Límites permite la definición de tres valores límites superiores y tres inferiores. Estos Sets son asignados a ciertos puntos del sistema. Posteriormente la comparación del valor actual del punto con estos límites en cada interrogación determina la generación de alarmas ante violaciones de dichos límites. En su creación se debe definir un nombre, que permite su identificación posterior y seis valores numéricos uno para cada límite.

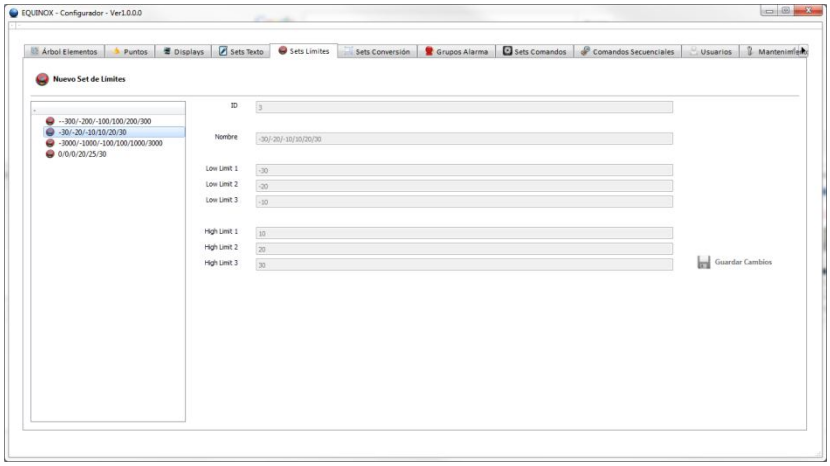


Figura 59 – Captura de pantalla de la sección Sets de Límites

Definición de Sets de Conversión

Estos Sets representan la fórmula de una recta que se utiliza para convertir cuentas en unidades de ingeniería. Cada Sets puede ser asignado más tarde a una serie de puntos. En la creación de estos Sets se define un nombre para su identificación posterior. En cuanto a la recta se define su pendiente la ordenada al origen y la pendiente.

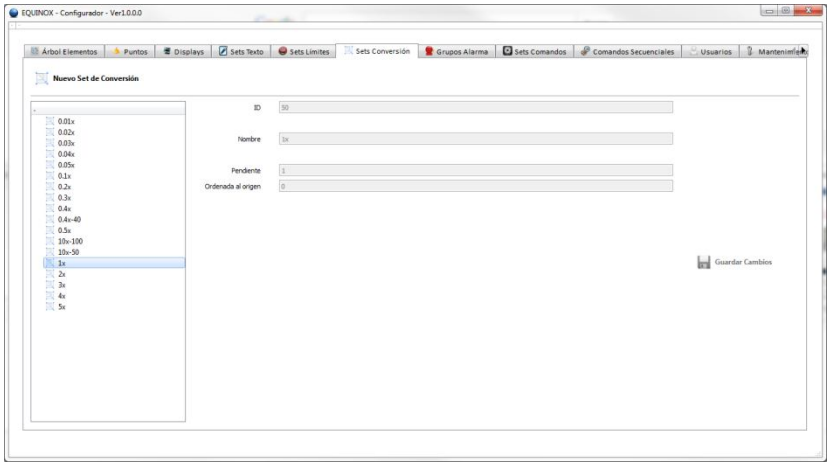


Figura 60 – Captura de pantalla de la sección Sets de Conversión

Definición de Grupos de Alarma

Los grupos de alarma se utilizan para categorizar las alarmas que genere cada punto del sistema de acuerdo a su naturaleza, prioridad, función, área, etc. Su definición comprende su denominación, para su identificación posterior, un color que se utilizará para el texto de las alarmas de este grupo en los listados y la selección de los usuarios que deban ser alertados por correo electrónico ante la ocurrencia de alarmas de este grupo.

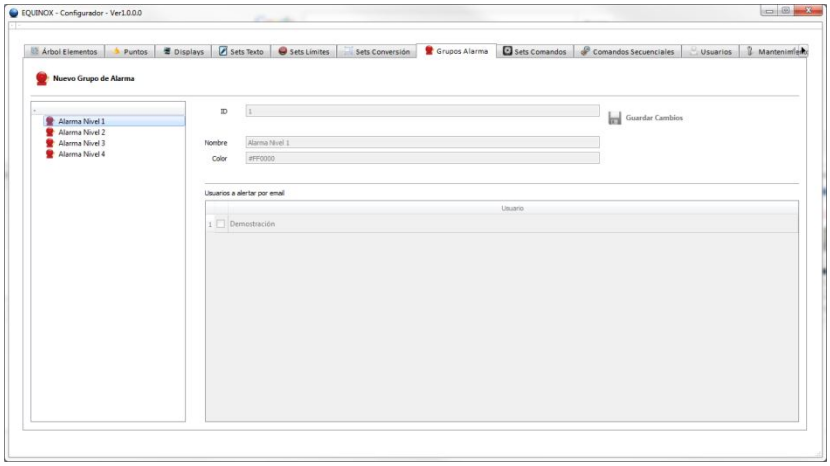


Figura 61 – Captura de pantalla de la sección Grupos de Alarma

Definición de Sets de Comando

En esta sección se crean los Sets de Comando, los cuales son conjuntos de telemandos o comandos mediante los cuales se envían órdenes a Dispositivos del sistema. Estos Sets pueden contener tanto Comandos Simples (cambios en salidas digitales) o Complejos (secuencia de telemandos que ejecutan acciones complejas). Se asigna una denominación, para su identificación posterior, se selecciona un Set de Texto para denominar la acción correspondiente a cada uno de los comandos. Luego se agrega un comando por cada posición del Set de Texto definiendo en cada uno de ellos si es Simple o Complejo y sus parámetros correspondientes.

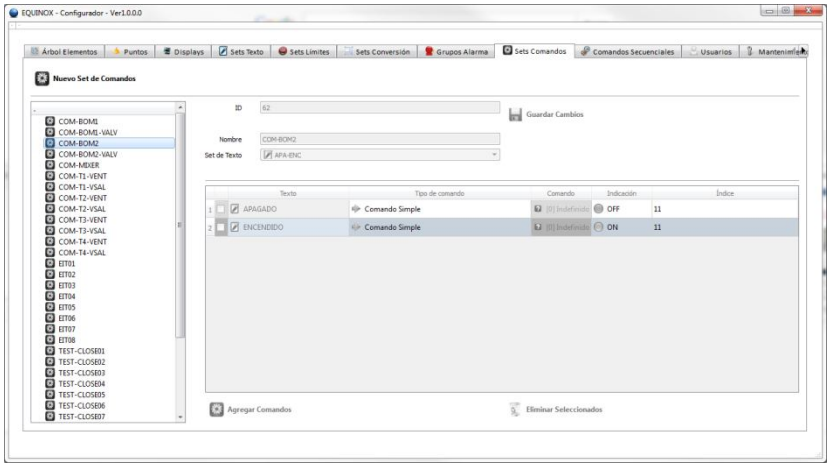


Figura 62 – Captura de pantalla de la sección Sets de Comando

Definición de Comandos Complejos Secuenciales

Esta sección permite la definición de una secuencia de Comandos Complejos. Para cada uno de ellos se selecciona el tipo de orden o acción que se desea que lleve a cabo, las opciones son alterar el estado de bits en registros de salida, establecer su valor en función de otros puntos del sistema o de valores fijos, y cambiar de estado varias salidas digitales en una sola operación. Además se define la dirección o índice del primer registro donde se ejecutará la acción, continuando luego con los registros siguientes y una lista de números de bits, número de punto o valores dependiendo de la acción que se haya seleccionado.

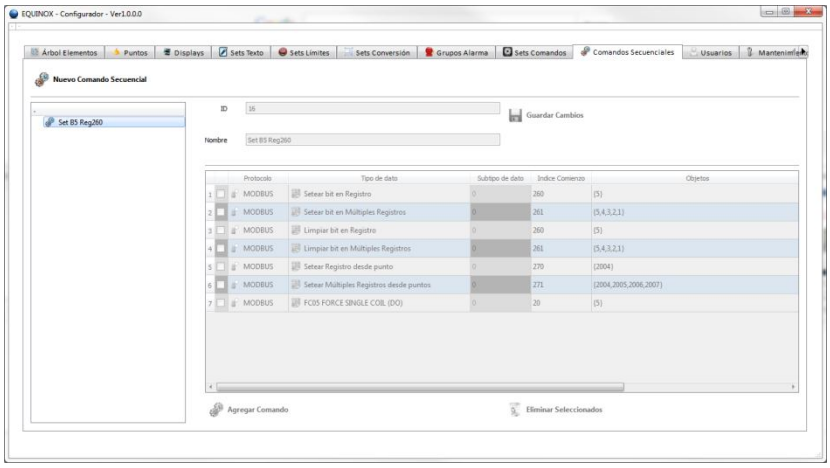


Figura 63 – Captura de pantalla de la sección Comandos Secuenciales

Definición de Puntos

Los puntos pueden definirse de dos formas, una de ellas es a través del Módulo Web activando el "modo edición" en donde es posible seleccionar símbolos vírgenes y asignarles un Nombre y un Sub-Nombre, esta asignación representará la creación de un nuevo punto del sistema. La segunda es hacerlo a través del Módulo de Configuración y Desarrollo. En cualquier caso este módulo es necesario para definir el resto de los parámetros del punto, es decir el Tipo y el Sub-tipo (entradas o salidas digitales o analógicas de 16 o 32 bits), el Set de Conversión para el pasaje de cuentas a unidades de ingeniería, el Set de Límites que permita calcular violaciones de valores límites, el Grupo de Alarma para categorizar el punto, el Set de Texto para establecer un texto para cada estado posible del punto y si deben generarse o no alarmas ante cambios de estado.

Por su parte se selecciona también la Estación, el Dispositivo y el número de índice o registro desde el cual se tomará el valor de este punto, esto teniendo en cuenta el Tipo y el Sub-Tipo elegido. Por último se determina si el punto realizará telemandos o si funcionará como vínculo a otra pantalla del sistema, si así fuera se seleccionará a continuación el telemando o la pantalla correspondiente.

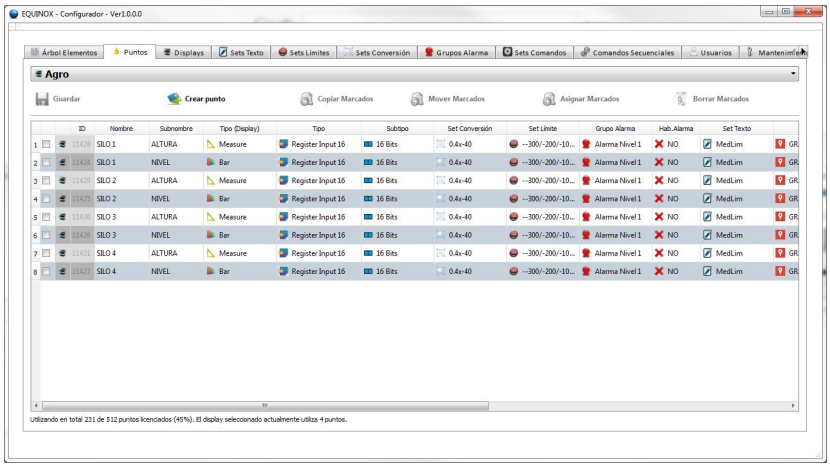


Figura 64 – Captura de pantalla de la sección Puntos

Definición de Usuarios

El Módulo Configurador y de Desarrollo permite la creación de usuarios con permiso de acceder al sistema. En la definición de cada usuario se ingresa su nombre real, el área a la que pertenece, un nombre de acceso y una contraseña. Además se ingresa su correo electrónico el cual es usado para el envío de mensajes de alarma. Por su parte se seleccionan las pantallas del sistema a las cuales se desea que el usuario tenga acceso.

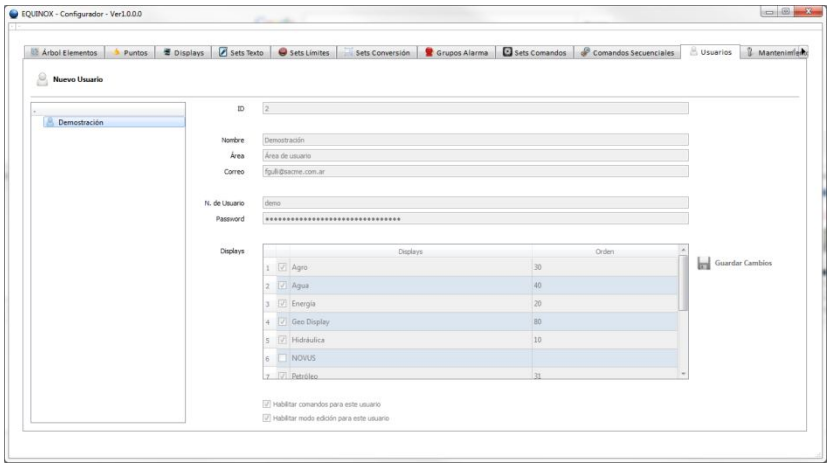


Figura 65 – Captura de pantalla de la sección Usuarios

Operaciones de Mantenimiento

Las operaciones de mantenimiento que pueden llevarse a cabo a través de este módulo son la exportación e importación de la configuración, lo cual implica un volcado completo del contenido de la base de datos del sistema, los archivos gráficos vectoriales de las pantallas y demás configuraciones. También es posible borrar la configuración actual y vaciar las listas de alarmas, eventos y valores históricos.

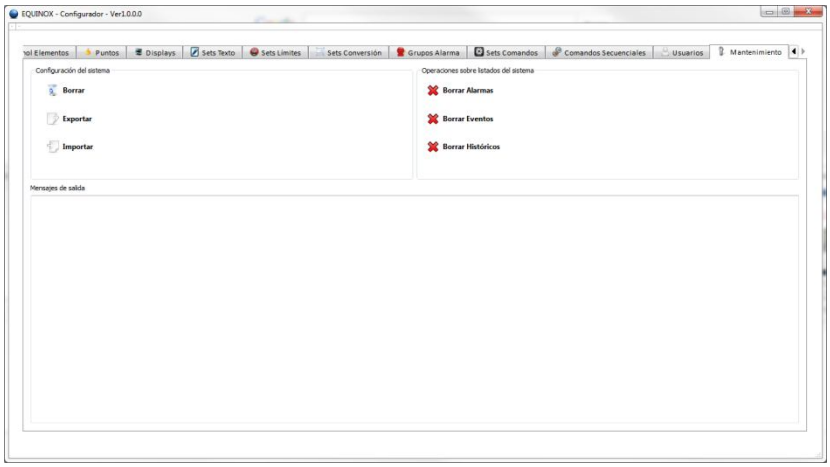


Figura 66 – Captura de pantalla de la sección Mantenimiento

7.2.3. Módulo HMI

El módulo HMI es la interfaz de operación del Sistema, también conocido como módulo "**Web**" es la herramienta mediante la cual los operadores pueden conectarse para interactuar y obtener información de los procesos bajo supervisión del sistema. Podemos decir que la computadora donde funcione esta aplicación es una consola de operación.

El enfoque tradicional para este tipo de función fue a través de la utilización de programas o aplicaciones especiales que debían ser instaladas en las consolas de operación. Éstas se comunicaban con el sistema mediante protocolos propietarios u obsoletos. Algunas plataformas incorporaron luego la posibilidad de acceso vía web, pero ésta en general no era un rediseño completo del acceso original sino que era más bien una adaptación para que pudiera funcionar a través de una LAN o de Internet. Un ejemplo de esto es el uso de algunos sistemas de tecnologías de escritorio remoto, como VNC, para extender la funcionalidad de la aplicación original sin grandes cambios. Es el caso del producto Simatic WinCC de la empresa Siemens.

Intentando superar el enfoque tradicional se decidió para la interfaz de operación del sistema diseñar una nueva aplicación que fuera completamente basada en tecnologías de web, esto es utilizando lenguajes de programación, herramientas y bibliotecas utilizadas en el mundo de Internet. Esto determinaba la utilización de una arquitectura cliente-servidor, en donde el cliente sería una aplicación para navegadores de internet y el servidor alguna plataforma que pudiera procesar las peticiones provenientes del navegador.

Para el cliente se eligió JavaScript junto con jQuery para simplificar la programación y del lado servidor se escogió PHP. Las razones de estas elecciones ya fueron expuestas.

7.2.3.1. Funciones

Comunicación cliente-servidor

Al comienzo del desarrollo de este módulo se debió definir el método de intercambio de información entre el cliente y el servidor. Fue necesario escoger un formato apropiado para esta tarea teniendo en cuenta la naturaleza de los datos enviados.

En un primer momento se seleccionó XML (Extensible Markup Lenguaje o Lenguaje de Marcas Extensible) creado por el consorcio internacional W3C. El mismo consiste en un conjunto de reglas para codificar documentos que pueden ser interpretados por máquinas pero también por personas.

Se compone de marcas y contenido, las marcas son cadenas de texto que ya sea comienzan con el carácter "<" y terminan con el ">"o bien comienzan con el carácter "&" y terminan con el ";". El contenido es todo lo que no sean marcas.

El documento se compone de elementos que contienen tags o etiquetas de inicio ("<ejemplo>") y de fin ("</ejemplo>"), el contenido es todo aquello que va entre ambas. También existen elementos vacíos que se especifican con etiquetas del tipo "<ejemplo/>".

El siguiente es un fragmento de código XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
<point_id>3</point_id>
<point_name>TANQUE 1</point_name>
<point_sub_name>LEVEL</point_sub_name>
<analog_indication>2104</analog_indication>
<l_u_epoch>1425291086.385</l_u_epoch>
<tags>1024</tags>
</root>
```

Como puede verse su estructura es la misma que la de los documentos HTML. En este caso se pretende representar un punto cuyo nombre es "TANQUE 1" su sub-nombre "LEVEL", su indicación "2104", su última actualización, y su status.

Considerando otras alternativas finalmente se decidió utilizar JSON o JavaScript Object Notation (notación de objetos JavaScript). El mismo es un formato abierto de intercambio de datos, liviano y también fácil de entender y escribir por personas. Aunque es un derivado del lenguaje JavaScript es un formato independiente de lenguajes. Un documento JSON se

compone de dos estructuras de datos. Por un lado una colección de pares “nombre/valor” desordenados, conocido como objetos, y listas de valores ordenados o arrays.

Los objetos comienzan con el carácter “{” y terminan con el “}”. Cada nombre es seguido de “:” y los pares nombre/valor son separados por comas. Los arrays comienzan con el carácter “[” y terminan con el “]”, los valores del array se separan por comas.

Representemos ahora la misma información antes expresada con XML pero esta vez JSON.

```
{
  "point_id":"3",
  "point_name":"TANQUE 1",
  "point_sub_name":"LEVEL",
  "tags":"1024",
  "l_u_epoch":"1425291086.385",
  "analog_indication":"2104"
}
```

Si comparamos ambas opciones veremos que el fragmento JSON más corto (menor tamaño) que el XML pero representa la misma información. En este ejemplo el ahorro en tamaño no es importante pero de querer transmitir conjuntos de decenas o cientos de puntos se hace notable. Esto lo hace el formato ideal para el intercambio de datos entre cliente y servidor y por esa razón fue elegido para el sistema.

La comunicación cliente-servidor intercambia diversos tipos de informaciones. Se muestran a continuación ejemplos de comunicación para los intercambios más comunes destacando los parámetros con los que el cliente consulta al servidor (ver pestaña “Parámetros”) y la respuesta desde el mismo en formato crudo (pestaña “Respuesta”) e interpretado como JSON (pestaña “JSON”).

Las capturas de pantalla corresponden al add-on “Firebug” para el navegador Mozilla Firefox, una herramienta de depuración de aplicaciones web que fue muy utilizada durante el desarrollo del sistema.

Lectura periódica de puntos de estado y analógicos cada un período determinado.

El cliente especifica como parámetros la pantalla de la cual se desean obtener puntos (parámetro “displayid”),el tipo de punto (en este caso estados, un valor numérico especial para el parámetro “type”) y un instante de tiempo para indicar que se desean los puntos actualizados con posterioridad a él. Este último parámetro evita recibir con cada interrogación la misma información, teniendo en cuenta que estos paquetes pueden llegar a ser largos esto ayuda a reducir el ancho de banda de la comunicación. El servidor responde con la información solicitada.

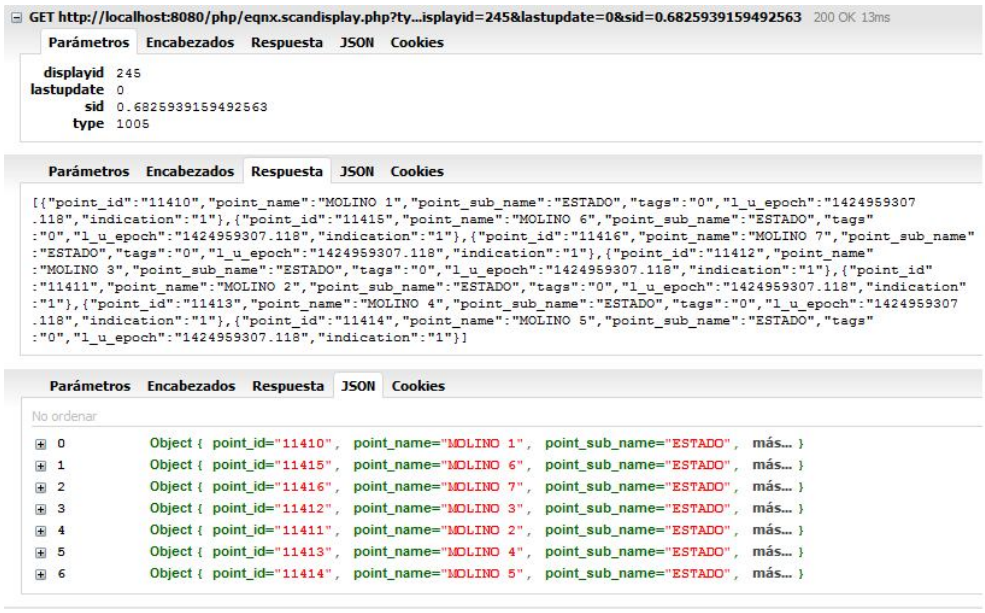


Figura 67 – Ejemplo de lectura de puntos de estado

Lectura periódica de alarmas, lectura de alarmas y eventos a petición del operador y en base a filtros establecidos.

El cliente especifica la cantidad de registros deseados (parámetro “Duration”), si se desean alarmas o eventos (parámetro “list”), el offset a partir del cual deben comenzar las alarmas (parámetro “Start”) y el usuario u operador que requiere esta información (parámetro “User_Id”). El servidor responde con una lista de registros indicando para cada uno la etiqueta de tiempo, el nombre de la estación y el dispositivo, el nombre y sub-nombre del punto, el texto correspondiente al estado actual y el color con que deben mostrarse el texto, entre otras informaciones.

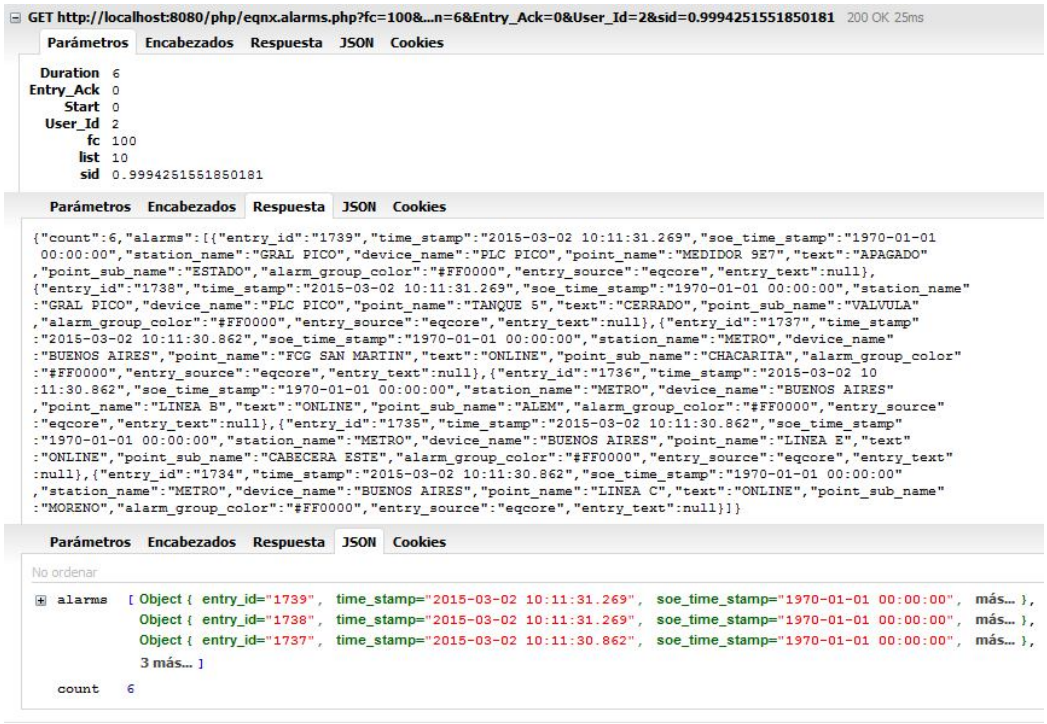


Figura 68 – Ejemplo de lectura de alarmas

Lectura de indicaciones de puntos para representación gráfica.

Los gráficos se construyen a partir de un conjunto de puntos. El cliente especifica la identificación de los puntos deseados (parámetro "Point_Ids" con formato array). Como respuesta el servidor envía la información solicitada detallando la identificación del punto, su valor y su status ("tags", que indica por ejemplo en línea, fuera de línea, etc.).

Figura 69 – Ejemplo de lectura de valores para representación gráfica

Lectura de indicaciones de puntos para representación histórica.

En este caso el cliente indica un intervalo de tiempo que especifica el rango de tiempo de la información que se desea obtener (parámetros "fromDate" y "toDate"). Asimismo se indica una vista (parámetro "Id"), la misma está definida en otra tabla de la base de datos y está conformada por un conjunto de puntos. Como respuesta el servidor envía una lista de etiquetas de tiempo, indicaciones y status ("tags"). La respuesta en la **Figura 70** aparece incompleta debido a que ésta contiene una gran cantidad de muestras, sin embargo para la interpretación del lector es suficiente con considerar sólo una de las ellas ya que su estructura es siempre la misma, alterándose solo sus valores.

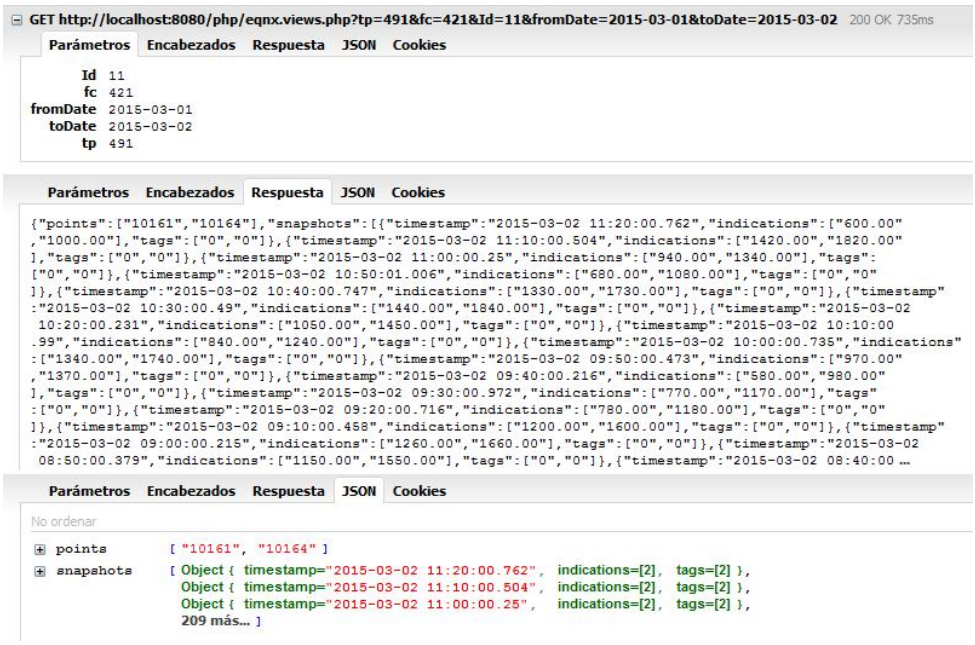


Figura 70 – Ejemplo de lectura de valores para representación histórica

Servidor Web

El servidor es la parte del Módulo HMI que atiende las peticiones del cliente, las procesa y hace las consultas correspondientes a la base de datos. Como ya se ha mencionado esta función es implementada mediante el software Nginx en cuanto al servicio. Por su parte se ha elegido como lenguaje de scripting PHP. Utilizando este lenguaje se han programado un conjunto de archivos que realizan varias funciones interactuando con el código JavaScript que corre en el navegador. Las respuestas JSON analizadas en el punto anterior son generadas por esta sección.

Interfaz de usuario

El cliente incorpora una interfaz gráfica para la interacción con los operadores del sistema, como ya se mencionó anteriormente se eligió para el desarrollo el lenguaje JavaScript. Por su parte también se utilizaron las siguientes bibliotecas:

- ✓ jQuery 2.0.3
  http://jquery.com/
  Biblioteca de uso general destinada a simplificar la programación de aplicaciones de lado cliente. Su uso fue constante en todos los aspectos de la interfaz de usuario.

- ✓ jquery.jqplot
  https://bitbucket.org/cleonello/jqplot/downloads/
  Biblioteca de generación de gráficas. Se utilizó para la representación de la evolución de la indicación de puntos del sistema.

- ✓ jquery.panzoom
  https://github.com/timmywil/jquery.panzoom

Biblioteca con funciones para realizar zoom y panning sobre archivos de gráficos vectoriales SVG. Se utilizó para poder realizar estas opciones sobre las pantallas gráficas del sistema.

- ✓ jquery.multiselect
  http://www.erichynds.com/blog/jquery-ui-multiselect-widget
  Permite implementar listas desplegables con elección de múltiples opciones. Se utilizó para opciones de filtrado de alarmas y eventos donde era importante poder seleccionar varias pantallas, estaciones, dispositivos y grupos de alarma.

- ✓ jquery.qtip
  http://qtip2.com/
  Herramienta que permite mostrar "tooltips" los cuales son pequeños letreros que se muestran cuando el mouse pasa por una determinada área de la pantalla. En particular se utilizó para mostrar el nombre y el sub-nombre de los puntos cuando el operador pasara el mouse sobre ellos.

- ✓ jquery-ui-1.11
  http://jqueryui.com/
  Biblioteca con varios componentes visuales ideales para la creación de aplicaciones basadas en web. Incluye botones, diálogos, pestañas, calendarios, barras de progreso, entre otras.

- ✓ json2.min
  http://json.org/
  Pequeña biblioteca para manipular documentos JSON.

Ingreso

El acceso a la interfaz es mediante la dirección IP o el dominio del servidor o computador donde se ha instalado el sistema. Se ha elegido como puerto de escucha el "8080", por lo cual para el ingreso se deberá abrir un navegador de internet y escribir en la barra de dirección: [IP_DEL_SERVIDOR]:8080. Al confirmar se cargará la pantalla de inicio de sesión.

Figura 71 – Pantalla de ingreso al sistema

Los usuarios se definen en el Módulo de Configuración y Desarrollo. Al ingresar los datos requeridos el operador ya está listo para trabajar con los procesos bajo supervisión. Como puede verse esta conexión puede llevarse a cabo en pocos segundos y sin necesidad de instalar software adicional.

Aspecto

El aspecto de la interfaz incluye una serie de pestañas, cada una de ellas corresponde a una pantalla de supervisión asignada al operador conectado. Además hay una pestaña de Listados, Historiador y Gráficas que se detallarán luego.

Se incluye un recuadro, que se ubica en la parte inferior de la pantalla, donde se muestran las últimas alarmas ocurridas sobre puntos del sistema. Este recuadro está constantemente en pantalla independientemente de la pestaña que se haya seleccionado para que el operador siempre tenga a la vista esta información.

Si la pestaña seleccionada corresponde a una pantalla de supervisión entonces el sistema cargará el archivo vectorial gráfico SVG asignado a la misma y lo mostrará en pantalla. Luego cargará desde la base de datos todos los puntos pertenecientes a la pantalla y los buscará dentro del documento SVG símbolos gráficos que posean el mismo nombre y del sub-nombre que los puntos.

Este reconocimiento de símbolos permite que luego el sistema pueda modificarlos gráficamente en función del estado actual del punto al que están relacionados. Así por ejemplo tomará el símbolo de una válvula y mostrará el gráfico cerrado o abierto según el punto tenga

indicación “1” o “0”. Lo mismo ocurre con los valores analógicos en donde reemplazará el texto del símbolo por la indicación actual del punto. También existen barras verticales y medidores que cambiarán su altura o el ángulo de sus manecillas también en función del punto. En el siguiente capítulo se explicará con mayor detalle el comportamiento de las pantallas gráficas y los símbolos.

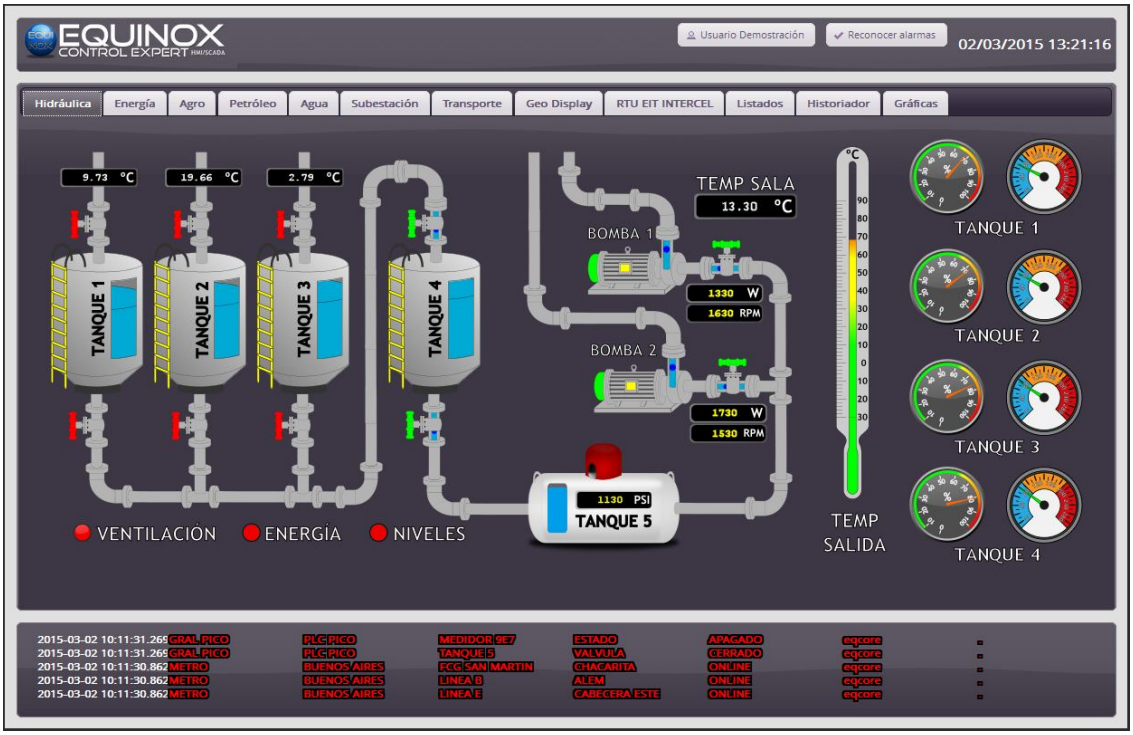


Figura 72 – Vista de Interfaz de usuario y ejemplo de pantalla gráfica del sistema

Lista de alarmas y eventos

En esta sección de la interfaz de usuario el operador puede consultar el listado de alarmas y eventos registrados por el sistema. Entre las opciones disponibles es posible filtrar la información presentada en pantalla en función de la pantalla, la estación, el dispositivo o el grupo de alarma al cual pertenecen los puntos sobre los cuales se produjeron estos registros. Se puede seleccionar entre eventos, alarmas no reconocidas y alarmas reconocidas. Además se incluyen botones para paginación y para el reconocimiento de la página de alarmas. Los datos del listado se pueden descargar en formato CSV (valores separados por coma) para su posterior análisis, por ejemplo en Microsoft Excel.

Figura 73 – Listado de alarmas y eventos

Vistas Gráficas

Existe una pestaña dentro de la interfaz para poder visualizar de forma gráfica la indicación de ciertos puntos del sistema en tiempo real. Un cierto conjunto de puntos se define como vista. El operador tiene la posibilidad de definir estas vistas y luego consultarlas de forma gráfica, a través de esta herramienta, o en forma histórica como se verá a continuación.

Entre las opciones del gráfico encontramos:

- ✓ **Tasa de refresco:** indica el intervalo de tiempo de toma de muestras de los puntos involucrados en el gráfico.
- ✓ **Ventana de tiempo:** indica el intervalo temporal mostrado en pantalla, el instante actual es siempre el extremo derecho del gráfico y a partir de este se muestra 1, 5, 10, 15, 30, 60 o 300 minutos hacia atrás.
- ✓ **Ventana de muestras:** Indica cuantas muestras están siendo presentadas en pantalla.
- ✓ **Botones de navegación temporal:** Se incluye la posibilidad de moverse de forma temporal hacia atrás y hacia adelante una determinada cantidad de ventanas de tiempo. También se puede hacer una pausa en la toma de muestras.

Por su parte se pueden consultar las muestras tomadas de cada punto incluido en la vista o quitarlos de la misma.

Cuando se produce la conexión del usuario comienza inmediatamente la toma de muestras de todas las vistas definidas, aun cuando no se encuentre seleccionada la pestaña de gráficas. Sin embargo si se seleccionara una ventana de tiempo que abarcara un tiempo mayor al de conexión del operador entonces el espacio faltante es completado tomando información de los archivos históricos del sistema.

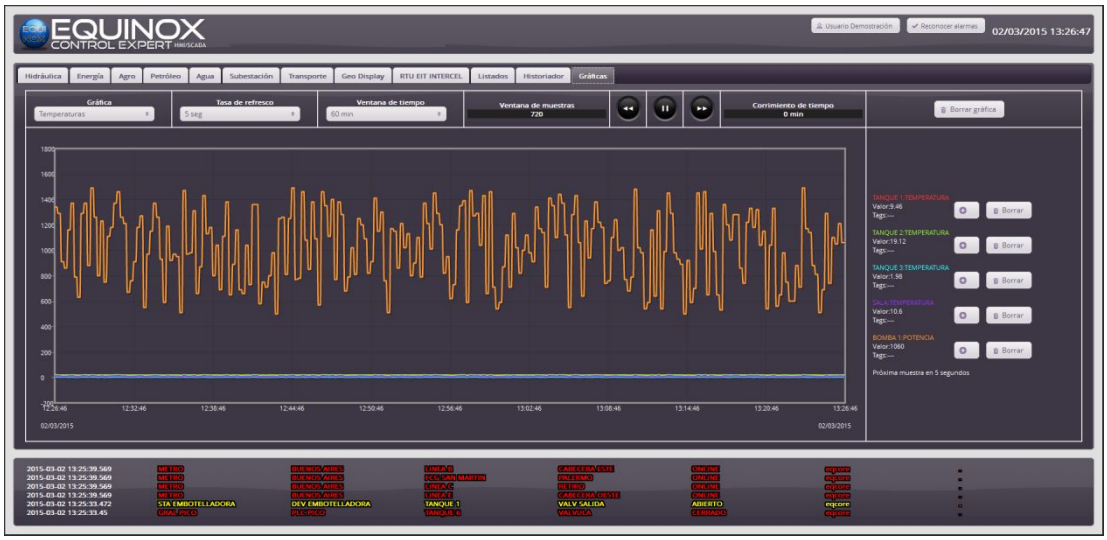


Figura 74 – Gráfica de evolución en tiempo real de la indicación de ciertos puntos del sistema

Vistas Históricas

Mediante esta herramienta se pueden consultar los datos almacenados periódicamente de todas las indicaciones de los puntos definidos. De forma predeterminada esta "fotografía" se toma cada 10 minutos.

El usuario puede armar vistas (conjuntos de puntos) para luego presentar en pantalla la información de las mismas en formato de tabla. La primera columna de esta tabla es la etiqueta de tiempo y las siguientes son las indicaciones históricas de cada punto integrante de la vista.

Se define un intervalo temporal mediante una fecha de comienzo y una de finalización y luego se activa la consulta. La información recuperada está paginada y se han ubicado botones para poder avanzar o retroceder las mismas. Por su parte se pueden descargar los datos en formato CSV (valores separados por coma) para su posterior análisis en otros programas como Microsoft Excel.

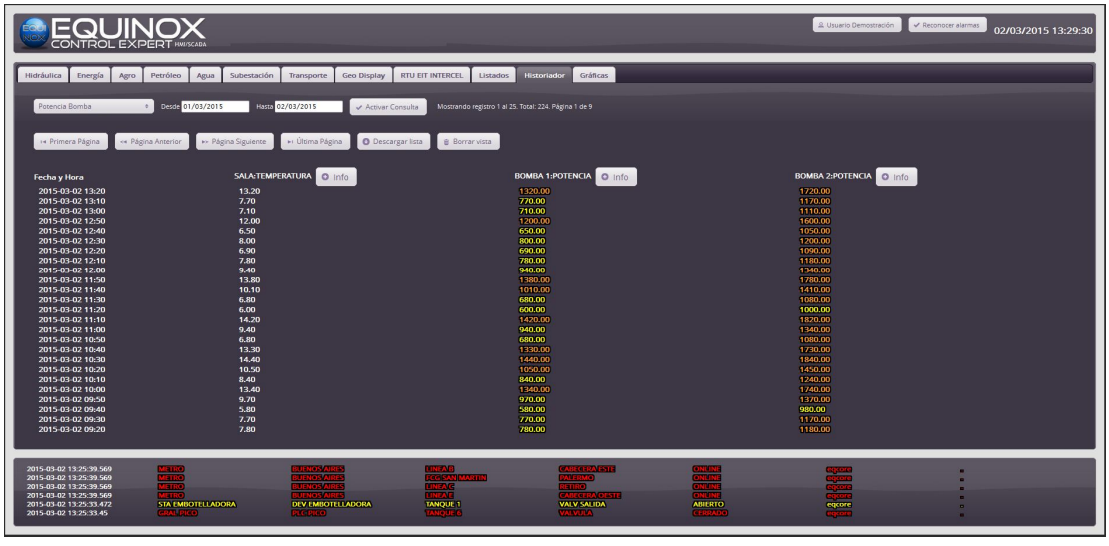


Figura 75 – Ejemplo de vista histórica de tres puntos del sistema

8. Capítulo VIII: Símbolos y pantallas SCADA

En los sistemas SCADA un símbolo es una agrupación de elementos gráficos destinados a representar de forma pictórica un determinado elemento, variable, parámetro o magnitud del mundo real.

Un símbolo puede representar una válvula, un interruptor, el nivel de un tanque, el valor de un termómetro, el estado de operación de un equipo o maquinaria, la presión de un recinto, entre muchas otras posibilidades. Los mismos son capaces de ser alterados o modificados dinámicamente de acuerdo a la condición actual del elemento que representan.

Para ejemplificarlo: el símbolo de una válvula cambiará alternativamente entre un gráfico de una válvula abierta y el de una cerrada de acuerdo a si la válvula real a la cual representa se encuentre abierta o cerrada. Esto puede verse con mayor claridad en la Figura 76.

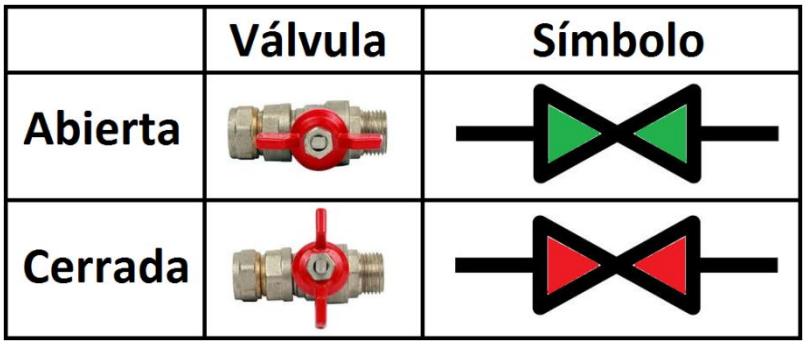


Figura 76 – Válvula abierta o cerrada y símbolo correspondiente

Son diseñados pensando en que el usuario pueda intuitivamente entender su significado rápidamente de un golpe de vista y sin necesidad información adicional. Los símbolos se agrupan dentro de pantallas SCADA. Una pantalla SCADA se compone de un conjunto de símbolos que representan un determinado proceso o situación de interés para el usuario del sistema.

Para la representación de símbolos y pantallas SCADA el sistema utiliza gráficos vectoriales en formato SVG. Se han elegido los gráficos vectoriales en general y el formato SVG en particular por las ventajas que ya fueron expuestas y que hacen ideal su uso para esta aplicación.

Se valoró especialmente la posibilidad de escalar sin pérdida de resolución o calidad, su reducido tamaño respecto a los mapas de bits para su envío a través de Internet y la facilidad con que puede ser manipulado en todos sus aspectos desde un navegador, produciendo las alteraciones gráficas necesarias sobre el mismo (cambios, movimientos, animaciones, etc.) y en tiempo de ejecución respondiendo a los cambios del proceso o sistema que representan.

Podemos categorizar los símbolos de acuerdo a su naturaleza. El sistema en estudio define cinco tipos:

- ✓ **Tipo Bar** Símbolos que representan barras.
- ✓ **Tipo Gauge** Símbolos que representan medidores.
- ✓ **Tipo Static** Símbolos estáticos.
- ✓ **Tipo Status** Símbolos de estado.
- ✓ **Tipo Measure** Símbolos para registros analógicos.

8.1. Características comunes

Los símbolos del sistema son grupos (elemento <g>) que contienen elementos gráficos simples como textos, líneas, rectángulos, círculos, paths, y diversas formas.

Los grupos correspondientes a los símbolos tienen un atributo especial, a ser interpretado sólo por el sistema, llamado "eq-symbol". Este atributo contiene como valor una cadena JSON con un solo par clave-valor correspondiente al tipo de símbolo, siendo la clave "type" y el valor "bar", "gauge", "status", "static" o "measure". Se eligió el formato de intercambio de datos JSON debido al posible requerimiento futuro de agregar pares clave-valor adicionales conteniendo otro tipo de información relativa al símbolo.

Dentro de este grupo principal que representa al símbolo se incluyen uno o más grupos con distintas funciones, excepto en el caso de los símbolos del tipo "measure" que contienen un elemento del tipo <text>. Estos elementos (<g> o <text>) tienen un atributo especial, a ser interpretado sólo por el sistema, llamado "eq-data". Este atributo contiene como valor una cadena JSON con uno o más pares clave-valor representando distintas informaciones relativas a este elemento. A continuación, en el detalle de cada tipo de símbolo se explicará con mayor profundidad el contenido de este atributo.

8.2. Descripción de tipos de símbolo

A continuación se expone con mayor detalle cada uno de ellos.

8.2.1. Tipo Bar

Los símbolos del tipo "bar" conceptualmente son barras o rectángulos cuyas dimensiones se ven alteradas en función del valor que tenga el registro de entrada o salida al cual estén asociados. Es un símbolo para representar valores del tipo analógico.

Estos símbolos son útiles para la representación de niveles de diversa índole. Por ejemplo el nivel de líquido dentro de un tanque, el nivel de mercurio dentro de un termómetro, el porcentaje actual de un determinado parámetro, etc.

8.2.1.1. Estructura

```
<svg>

    <g eq-symbol="{'type':'bar'}">
```

```
<!—Contenido estático (opcional) -->
<g eq-data=”{‘type’:’bar’,
’maxHeight’:’300’,
‘minHeight’:’10’,
’maxValue’:’90’,
’minValue’:’-30’}”>
<!—Otros elementos (opcional) -->
<rect></rect>
</g>
</g>
</svg>
```

El grupo principal del símbolo contiene un atributo “eq-symbol” que indica el tipo de símbolo, en este caso “bar”. Este atributo es reconocido sólo por el sistema para identificar, cuando se carga la pantalla SCADA, todos los símbolos de este tipo.

Dentro de este grupo encontramos lo siguiente:

- ✓ Grupo con atributo “eq-data”: Este grupo contiene los elementos gráficos que cambiarán de aspecto con las variaciones en el registro asociado al símbolo. Dentro del mismo se puede incluir:
  - o Elemento <rect>: Los elementos del tipo “rect” cambiarán su altura, es decir el atributo “height”.
  - o Otros elementos: Se utilizan para mejorar el aspecto del símbolo, por ejemplo para que luzca en tres dimensiones. Cambiarán su posición vertical para acompañar las variaciones de altura del <rect> anterior.
- ✓ Contenido estático: Otros elementos gráficos que no cambian de aspecto en relación a los registros asociados al símbolo. Por ejemplo: En el caso de un símbolo que represente el nivel de un tanque, aquí se colocarán los elementos gráficos que representen al tanque en sí mismo, su estructura, sus soportes, una escalera lateral, etc. Es decir todos los elementos excepto el nivel propiamente dicho.

8.2.1.2. Parámetros del atributo “eq-data”

- ✓ **type**: Indica el tipo de símbolo, coincide con lo indicado en el grupo padre dentro del atributo “eq-symbol” con el parámetro “type”.

- ✓ **maxHeight:** Este parámetro determina la altura máxima en píxeles que podrá tomar el rectángulo. Si se omite este parámetro se tomará como altura máxima la altura estática del <rect>, es decir la altura del elemento antes de comenzar a ser manipulado por el sistema.
- ✓ **minHeight:** Este parámetro determina la altura mínima en píxeles que podrá tomar el <rect>. Si se omite esta parámetro se tomará como cero.
- ✓ **maxValue:** Este parámetro define el valor máximo en unidades de ingeniería que el registro asociado al símbolo debería tener. Por ejemplo si se trata de un tanque en donde el nivel podrá tomar como máximo una altura de 10 metros este valor deberá ser 10. Si se omite este parámetro se considera un porcentaje, tomando como valor predeterminado cien.
- ✓ **minValue:** De forma similar este parámetro define el valor mínimo en unidades de ingeniería que el registro asociado al símbolo debería tener. Si se omite este parámetro se considera un porcentaje, tomando como valor predeterminado cero.

Cuando el sistema lee estos parámetros del símbolo realiza los siguientes cálculos:

slope= (maxHeight - minHeight) / (maxValue - minValue)

yIntercept = (2*(minHeight * minValue)-(maxHeight * minValue)-(minHeight * maxValue)) / (maxValue - minValue)

newHeight = slope * indication + yIntercept

Básicamente se construye una recta en base a los parámetros recogidos calculando una pendiente y una ordenada al origen. Con ellos se calcula la nueva altura en píxeles que el rectángulo debería tomar. Más tarde se aplica el valor "newHeight" al <rect>. Por otro lado los otros elementos pertenecientes al grupo con atributo "eq-data" son desplazados verticalmente para acompañar el cambio de altura del <rect>.

8.2.1.3. Ejemplo

El siguiente es un símbolo del tipo "bar" desglosado para evaluar por separado los elementos que lo componen.

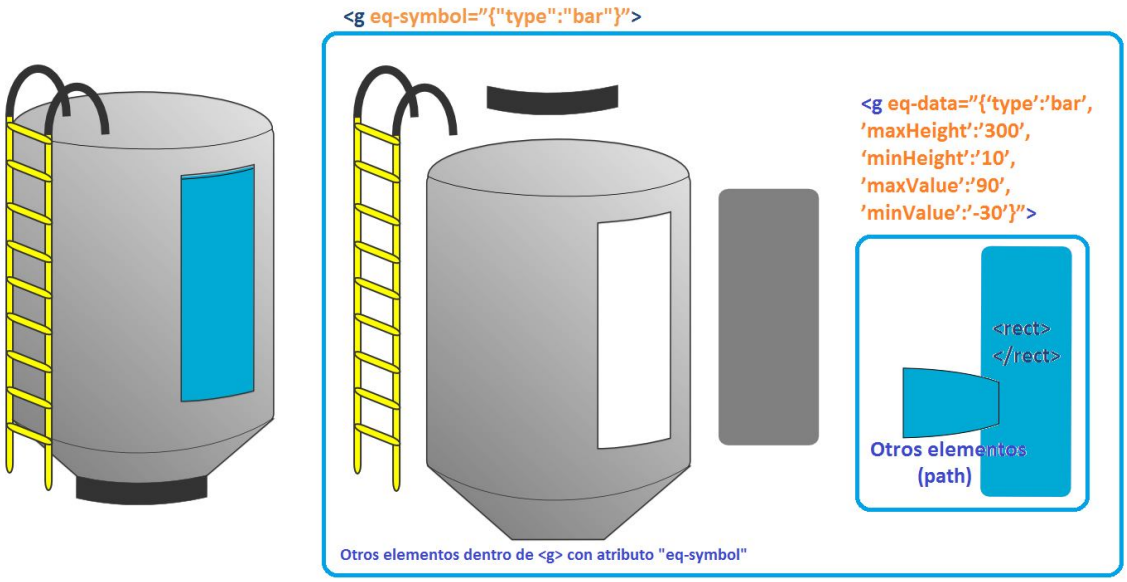


Figura 77 – Desglose de símbolo del tipo "bar"

Se trata de un tanque en tres dimensiones, como vemos todos sus elementos forman parte de un grupo con atributo "eq-symbol". Algunos de ellos son estáticos, como los que pueden verse a la izquierda del diagrama, y otros dinámicos, como los que forman parte del grupo con atributo "eq-data". En este último se definen los parámetros "maxValue", "minValue", "maxHeight" y "minHeight". El <rect> dentro del grupo con atributo "eq-data" cambiará de altura de acuerdo a lo expuesto con anterioridad y los otros elementos que lo acompañan, en este caso un <path> cambiarán de posición para permanecer atados a la nueva altura del <rect>.

8.2.2. Tipo Gauge

Los símbolos del tipo "gauge" son conceptualmente medidores cuya manecilla toma un ángulo en función del valor que tenga el registro de entrada o salida al cual estén asociados. Es un símbolo para representar valores del tipo analógico.

Estos símbolos son útiles para la representación de valores de diversa índole. Permiten al operador determinar de un golpe de vista si una serie de parámetros se encuentran dentro de los valores esperados. Por ejemplo la temperatura de un horno, la tensión o corriente de un circuito, la presión de una tubería, etc.

8.2.2.1. Estructura

```
<svg>
    <g eq-symbol="{'type':'gauge'}">
        <!—Contenido estático (opcional) -->
        <g eq-data="{'type':'gauge',
```

```
                    'maxAngle':'260',
                    'minAngle':'30',
                    'maxValue':'90',
                    'minValue':'-90'
                    'center':'100 100'}">
                        <path></path>
                    </g>
                </g>
            </svg>
```

El grupo principal del símbolo contiene un atributo "eq-symbol" que indica el tipo de símbolo, en este caso "gauge". Este atributo es reconocido sólo por el sistema para identificar, cuando se carga la pantalla SCADA, todos los símbolos de este tipo.

Dentro de este grupo encontramos lo siguiente:

- ✓ Grupo con atributo "eq-data": Este grupo contiene el elemento gráfico que rotará en relación a las variaciones en el registro asociado al símbolo. Dentro del mismo se debe incluir un elemento del tipo "path" que será una representación gráfica de la manecilla del medidor.
- ✓ Contenido estático: Otros elementos gráficos que no cambian de aspecto en relación a los registros asociados al símbolo. Por ejemplo: marco del medidor, color de fondo, escala, etc.

8.2.2.2. Parámetros del atributo "eq-data"

- ✓ **type**: Indica el tipo de símbolo, coincide con lo indicado en el grupo padre dentro del atributo "eq-symbol" con el parámetro "type".
- ✓ **maxAngle:** Este parámetro determina el ángulo máximo que podrá tomar la manecilla del medidor. Si se omite este parámetro se tomará como ángulo máximo el valor 360 grados.
- ✓ **minAngle:** Este parámetro determina el ángulo mínimo que podrá tomar la manecilla del medidor. Si se omite este parámetro se tomará como ángulo máximo el valor 0 grados.
- ✓ **maxValue:** Este parámetro define el valor máximo en unidades de ingeniería que el registro asociado al símbolo debería tener. Por ejemplo si se trata de una presión donde el fondo de escala del medidor debe corresponder a 300 psi, este valor deberá

ser 300. Si se omite este parámetro se considera un porcentaje, tomando como valor predeterminado cien.

- ✓ **minValue:** De forma similar este parámetro define el valor mínimo en unidades de ingeniería que el registro asociado al símbolo debería tener. Si se omite este parámetro se considera un porcentaje, tomando como valor predeterminado cero.
- ✓ **center:** Par de valores [x,y] en píxeles en torno a los cuales deberá girar la manecilla del medidor, es decir el elemento gráfico dentro del grupo con atributo "eq-data".

Cuando el sistema lee estos parámetros del símbolo realiza los siguientes cálculos:

slope = (maxAngle - minAngle) / (maxValue - minValue)

yIntercept = (2*(minAngle * minValue)-(maxAngle * minValue)-(minAngle * maxValue)) / (maxValue - minValue)

newAngle = slope * indication + yIntercept

Básicamente se construye una recta en base a los parámetros recogidos calculando una pendiente y una ordenada al origen. Con ellos se calcula el ángulo que la manecilla del medidor debería tomar. Más tarde se agrega un atributo del tipo "transform" al elemento gráfico <path> que representa a la manecilla del medidor, este atributo contiene el valor newAngle indicando el ángulo que debe rotar y en torno a que punto, indicado por el parámetro center leído desde el símbolo.

8.2.2.3. Ejemplo

El siguiente es un símbolo del tipo "gauge" desglosado para evaluar por separado los elementos que lo componen.

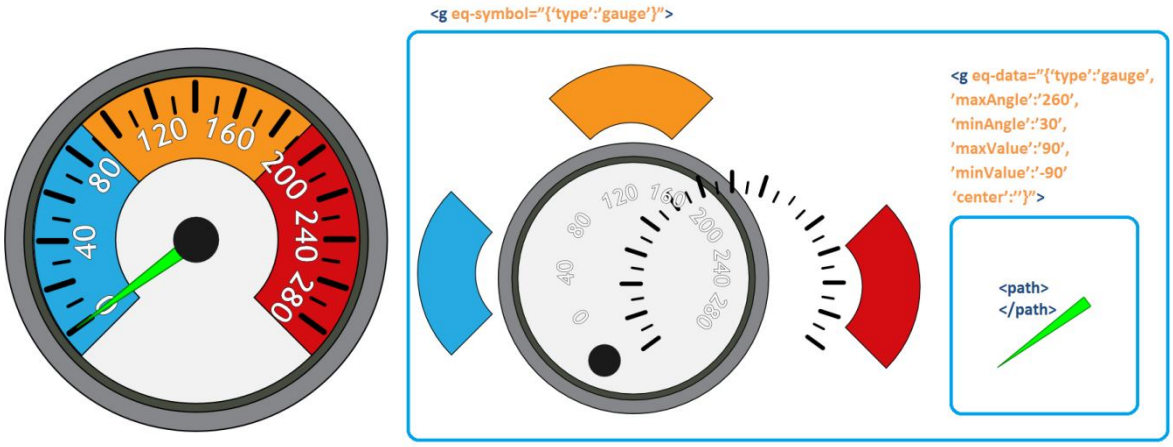


Figura 78 – Desglose de símbolo del tipo "gauge"

Se trata de un medidor con una escala que va desde el valor 0 hasta el 280. Como vemos todos sus elementos forman parte de un grupo con atributo "eq-symbol". Algunos de ellos son estáticos, como los que pueden verse a la izquierda del diagrama, y uno de ellos es dinámico, como el que forman parte del grupo con atributo "eq-data". En este último se definen los

parámetros “maxValue”, “minValue”, “maxAngle”, “minAngle” y “center”. El <path> dentro del grupo con atributo “eq-data” rotará en torno a “center” un cierto ángulo de acuerdo a lo expuesto con anterioridad.

8.2.3. Tipo Static

Los símbolos del tipo “static” son símbolos que, como su nombre lo indica, no cambian su aspecto durante su utilización aunque sí están relacionados a un determinado punto del sistema.

Estos símbolos son útiles para la representación de otro tipo de elementos, los cuales si bien son estáticos forman parte del proceso que representa la pantalla y deben ser presentados para ejecutar acciones sobre ellos. Ejemplos de uso son botones para vincular pantallas del sistema o para ejecutar telemandos sobre dispositivos conectados al sistema.

8.2.3.1. Estructura

```
<svg>
    <g eq-symbol=”{‘type’:’static’}”>
        <!—Contenido estático -->
    </g>
</svg>
```

El grupo principal del símbolo contiene un atributo “eq-symbol” que indica el tipo de símbolo, en este caso “static”. Este atributo es reconocido sólo por el sistema para identificar, cuando se carga la pantalla SCADA, todos los símbolos de este tipo.

Dentro de este grupo encontramos elementos gráficos que componen el contenido del símbolo. Estos no tienen ninguna función en particular sino que forman parte de la representación pictórica.

8.2.3.2. Parámetros del atributo “eq-data”

- **type**: Indica el tipo de símbolo, coincide con lo indicado en el grupo padre dentro del atributo “eq-symbol” con el parámetro “type”.

8.2.3.3. Ejemplo

El siguiente es un símbolo del tipo “static” desglosado para evaluar por separado los elementos que lo componen.

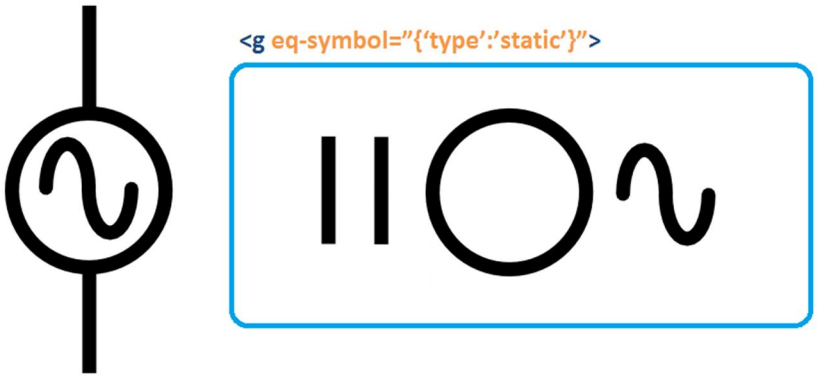


Figura 79 – Desglose de símbolo del tipo "static"

Se trata del símbolo de un generador, el cual podrá ser usado por ejemplo para ejecutar una acción sobre el mismo al hacer clic en el gráfico.

8.2.4. Tipo Status

Los símbolos del tipo "status" se utilizan para representar valores del tipo digital de uno o dos bits. Pueden tomar un número finito de estados: dos en el caso de los estados de un bit o cuatro en el de dos bits. El estado será seleccionado de acuerdo a la situación actual del punto de entrada o salida del sistema al cual esté el símbolo asignado.

Estos símbolos son útiles para la representación de la condición de dispositivos tales como bombas (encendida/apagada), válvulas (abierto/cerrado), interruptores (abierto/cerrado/en tránsito/error), seccionadores, estado de equipos (en línea/fuera de línea) o de cualquier elemento que pueda tomar un número finito de estados, siendo las posibilidades dos (estados digitales de un bit de longitud pudiendo tomar valores "0" o "1") o cuatro (estados digitales de dos bits de longitud pudiendo tomar valores "00", "01", "10" o "11").

8.2.4.1. Estructura

```
<svg>
    <g eq-symbol="{'type':'status'}">
        <!—Contenido estático (opcional) -->
        <geq-data="{'type':'status','valqual':'0:0'}">
            <!—Contenido  -->
        </g>
        <geq-data="{'type':'status','valqual':'0:1'}">
            <!—Contenido  -->
```

```
            </g>
            <geq-data="{'type':'status','valqual':'1:0'}">
                <!—Contenido -->
            </g>
            <geq-data="{'type':'status','valqual':'1:1'}">
                <!—Contenido -->
            </g>
        </g>
    </svg>
```

El grupo principal del símbolo contiene un atributo "eq-symbol" que indica el tipo de símbolo, en este caso "status". Este atributo es reconocido sólo por el sistema para identificar, cuando se carga la pantalla SCADA, todos los símbolos de este tipo.

Dentro de este grupo encontramos lo siguiente:

- ✓ Grupos con atributo "eq-data": Contienen la representación gráfica del par estado:calidad especificado dentro del atributo con el parámetro "valqual". Este grupo se adquirirá visibilidad, es decir se mostrará en pantalla, cuando la información especificada en este parámetro se corresponda con la realidad, al mismo tiempo se ocultarán el resto de los grupos con atributo "eq-data".
- ✓ Contenido estático: Otros elementos gráficos que no cambian de aspecto en relación a los registros asociados al símbolo. Por ejemplo: decoración adicional, marcos, colores de fondo, etc.

8.2.4.2. Parámetros del atributo "eq-data"

- ✓ **type**: Indica el tipo de símbolo, coincide con lo indicado en el grupo padre dentro del atributo "eq-symbol" con el parámetro "type".
- ✓ **valqual:** Este parámetro contiene dos indicaciones numéricas separadas por el signo ":". La primera de ellas corresponde al número de estado, pudiendo ser "0" o "1" para los estados digitales de un bit de longitud o "0", "1", "2" o "3" para los estados digitales de dos bits de longitud. El segundo valor numérico corresponde a la calidad de la información recibida, siendo "0" para el caso en donde la calidad es correcta y "1" cuando no lo es. Esto es útil para señalizar cuando, por ejemplo, se ha perdido la comunicación con el dispositivo desde donde se lee este valor.

8.2.4.3. Ejemplo

El siguiente es un símbolo del tipo “status” desglosado para evaluar por separado los elementos que lo componen.

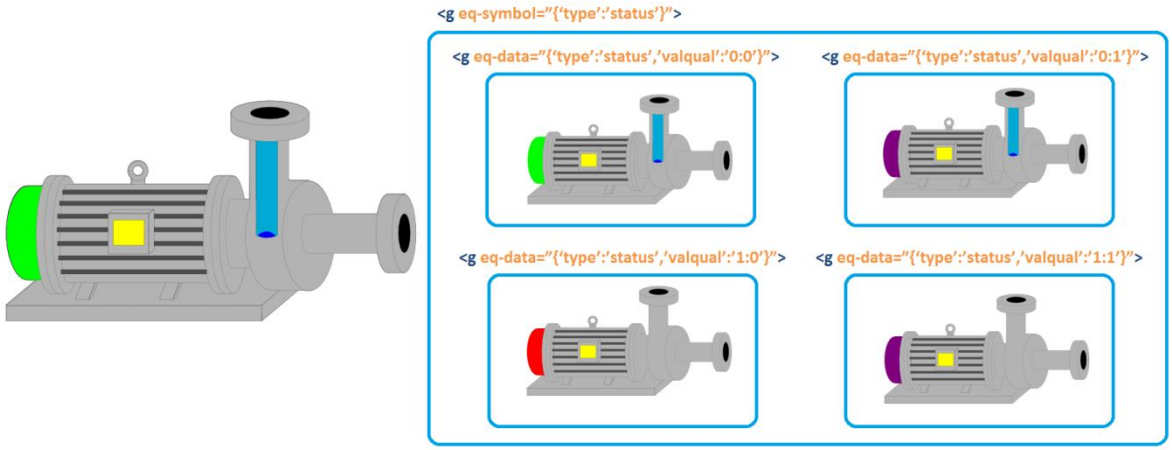


Figura 80 – Desglose de símbolo del tipo “status”

Se trata de un símbolo que representa una bomba. Como puede verse existen en realidad cuatro bombas dibujadas, una para cada combinación posible “[estado]:[calidad]”. Los cuatro dibujos se encuentran superpuestos, cada uno de ellos pertenece a un grupo con atributo “eq-data”. A su vez estos cuatro grupos pertenecen a un grupo padre con atributo “eq-symbol”. Cuando se utiliza el símbolo, alguna de las cuatro gráficas es presentada en pantalla. Las otras tres permanecen ocultas. El sistema define la gráfica que se visualizará en pantalla en función del par “[estado]:[calidad]” vigente en cada momento.

Por ejemplo, si la bomba se encuentra encendida (estado numérico 0) y la comunicación con el dispositivo desde donde se está leyendo la información de este punto es correcta (calidad 0) entonces se presentará en pantalla el dibujo que se encuentra arriba a la izquierda en la figura. En cambio si la comunicación con el dispositivo está interrumpida (calidad 1) y el sistema sabe que la bomba se encontraba apagada (estado numérico 1) entonces se presentará en pantalla el dibujo que se encuentra abajo a la derecha en la figura.

Puede incluirse, aunque no es el caso en este ejemplo, contenido estático dentro del grupo con atributo “eq-symbol”. Este contenido será siempre visible independientemente del estado del punto.

8.2.5. Tipo Measure

Los símbolos del tipo “measure” son representaciones numéricas de registros de entrada o salida del sistema. Es un símbolo para esquematizar valores del tipo analógico.

Estos símbolos son útiles para la representación de valores de diversa índole. Permiten al operador determinar de un golpe de vista si una serie de parámetros se encuentran dentro de los valores esperados. Por ejemplo la temperatura de un horno, la tensión o corriente de un circuito, la presión de una tubería, etc.

8.2.5.1. Estructura

```
<svg>
    <g eq-symbol="{'type':'measure'}">
        <!—Contenido estático (opcional) -->
        <text eq-data="{'type':' measure'"></text>
    </g>
</svg>
```

El grupo principal del símbolo contiene un atributo "eq-symbol" que indica el tipo de símbolo, en este caso "measure". Este atributo es reconocido sólo por el sistema para identificar, cuando se carga la pantalla SCADA, todos los símbolos de este tipo.

Dentro de este grupo encontramos lo siguiente:

- ✓ Texto con atributo "eq-data": Contiene la representación numérica del valor actual del registro de entrada o salida al cual se encuentra asignado el símbolo.
- ✓ Contenido estático: Otros elementos gráficos que no cambian de aspecto en relación a los registros asociados al símbolo. Por ejemplo: decoración adicional, marcos, colores de fondo, etc.

8.2.5.2. Parámetros del atributo "eq-data"

- ✓ **type**: Indica el tipo de símbolo, coincide con lo indicado en el grupo padre dentro del atributo "eq-symbol" con el parámetro "type".

8.2.5.3. Ejemplo

El siguiente es un símbolo del tipo "measure" desglosado para evaluar por separado los elementos que lo componen.

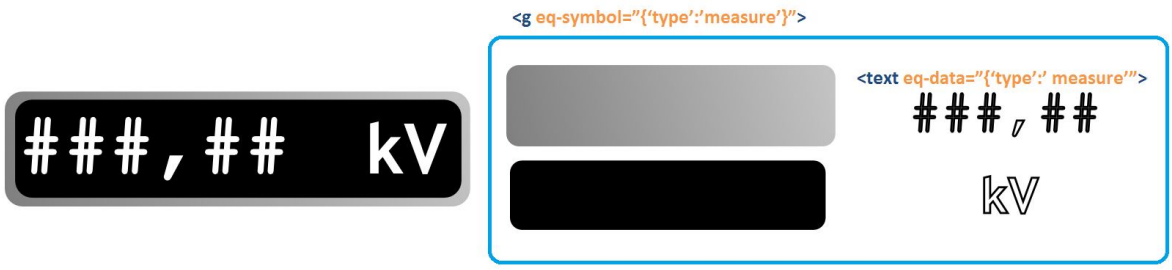


Figura 81 – Desglose de símbolo del tipo "measure"

Se trata de un símbolo que indica un valor numérico expresado en kilo volts. Como vemos todos sus elementos forman parte de un grupo con atributo "eq-symbol". Algunos de ellos son estáticos, como los que pueden verse a la izquierda del diagrama, por su parte el elemento <text> con atributo "eq-data" es dinámico. Este cambiará de contenido de acuerdo al valor actual del punto al cual el símbolo se encuentre asociado. Existe otro elemento del tipo <text> correspondiente al texto "kV", sin embargo este permanecerá estático por no tener definido el atributo "eq-data".

8.3. Edición pantallas y de símbolos

8.3.1. Inkscape

Para la edición de pantallas y símbolos se utiliza un programa de computación de código abierto y gratuito disponible para ser descargado de Internet.

Su nombre es Inkscape y su función es la edición de gráficos vectoriales en formato SVG. Es una alternativa gratuita a programas como Adobe Illustrator, a pesar de no tener costo tiene características que lo equiparan técnicamente a este último. Es multiplataforma por lo que pueden conseguirse versiones para los sistemas operativos del tipo OS X, Unix, Linux y Windows.

8.3.1.1. Breve historia

Su historia se remonta al proyecto Sodipodi, el predecesor de Inkscape, desarrollado hasta el año 2004. El mismo fue escrito por Ralph Levien y Lauris Kaplinsky, entre otros. Ralph es un miembro destacado de la comunidad de desarrolladores de software libre. Levien creó en 1999 una primera versión llamada Gill tomando de base un editor y visualizador muy simple para el nuevo formato SVG que en ese entonces se encontraba en discusión en el W3C. Más tarde abandonó el proyecto.

Más tarde Kaplinsky retomaría el desarrollo renombrando el proyecto a Sodipodi. Y estableció una meta mucho más ambiciosa: Desarrollar un editor de gráficos vectoriales poderoso con una interfaz de usuario parecida a CorelDRAW pero también influenciada por Gimp. Lauris aportó un gran caudal de trabajo a Sodipodi, una gran porción del código de Inkscape está

registrado a su nombre. Sin embargo la conducción autocrática del proyecto se volvió un obstáculo (Kirsanov, 2009, pág. 15).

En Octubre de 2003 un grupo de desarrolladores insatisfechos luego de no lograr un acuerdo son Kaplinsky comenzaron una bifurcación del proyecto. Tomaron el código fuente más reciente de Sodipodi, agregaron sus parches y le dieron un nuevo nombre: Inkscape. Por un tiempo Inkscape y Sodipodi se desarrollaron en paralelo, pero pronto este último congeló su desarrollo.

En contraste con su predecesor Inkscape es más abierto. Nadie tiene la palabra suprema de cómo se deben hacer las cosas. Para tener derechos plenos de desarrollador, incluyendo el derecho a aplicar cambios directamente al repositorio central, alcanza con presentar dos parches exitosos.

Como resultado el proyecto ha crecido rápidamente y sin cesar. Se ha transformado en una herramienta poderosa, simple de usar y fácil de aprender.

Sus características la hacen ideal para su aplicación en la edición de pantallas y gráficas y símbolos del sistema.

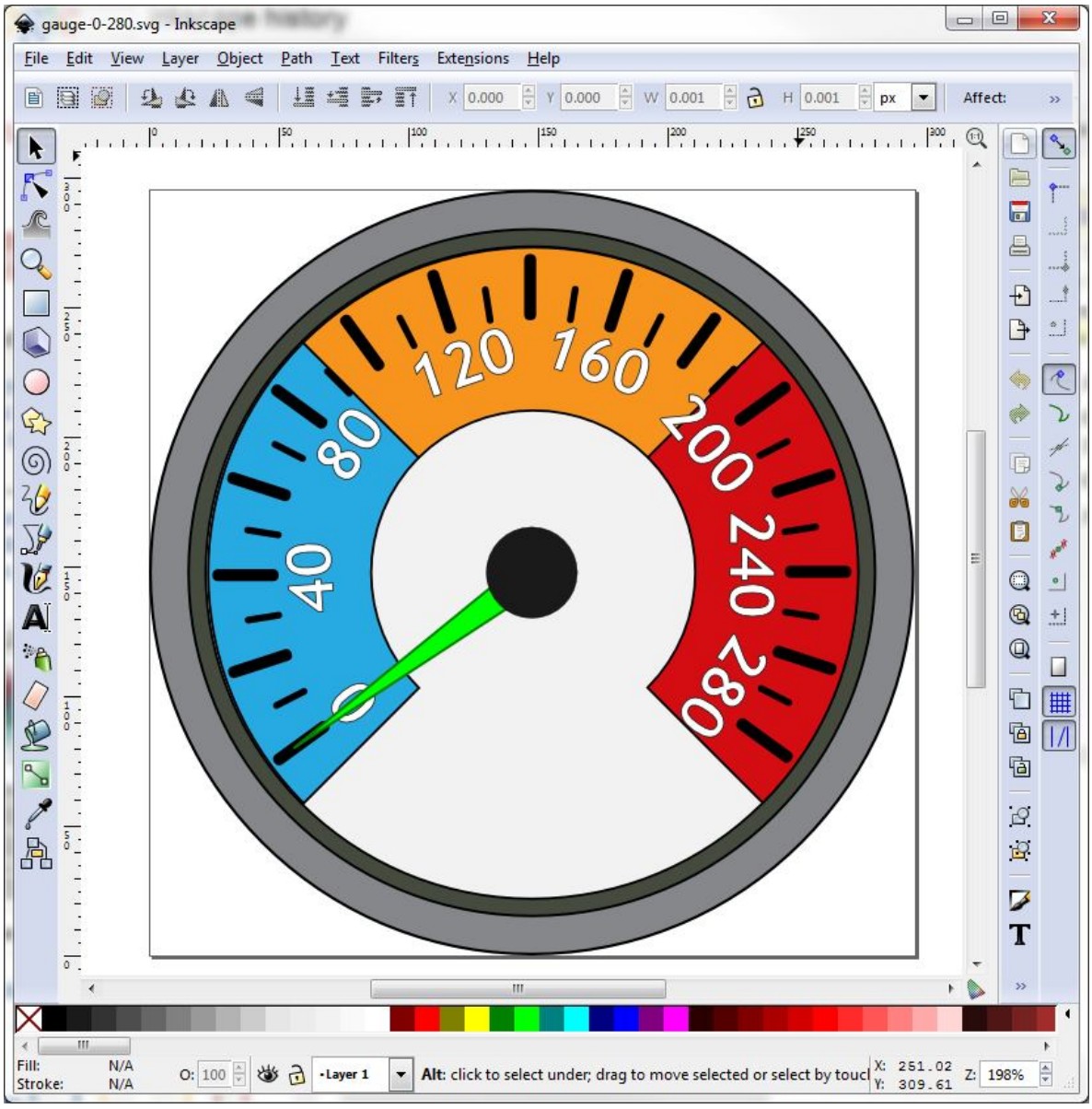


Figura 82 – Edición de símbolos en Inkscape

8.4. Biblioteca de símbolos

Se ha creado una pequeña biblioteca de símbolos para el sistema. Los mismos están categorizados de acuerdo al tipo: bar, gauge, static, status, measure. Los siguientes son algunos ejemplos de símbolos pertenecientes al sistema.

8.4.1. Tipo Bar

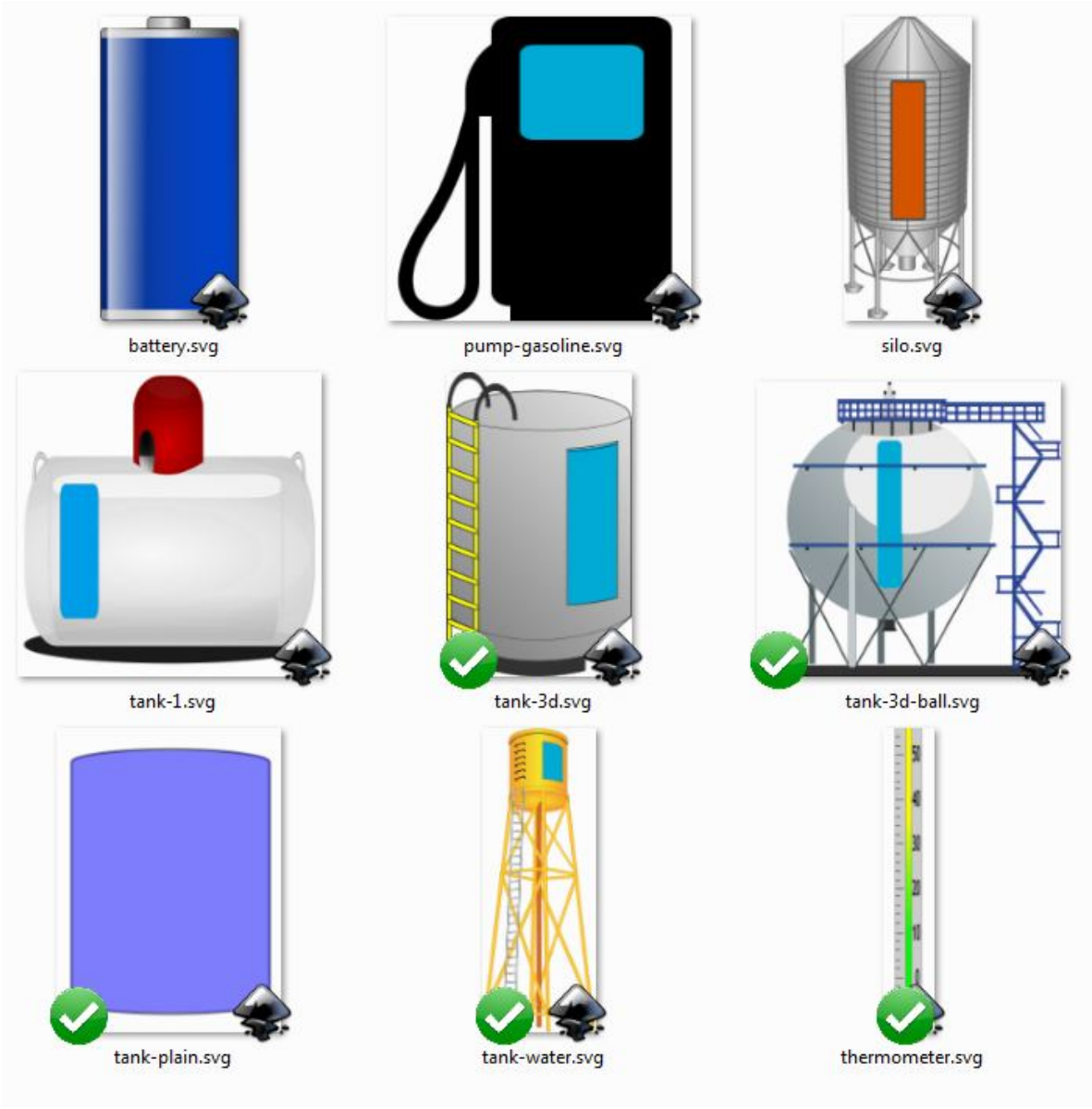


Figura 83 – Ejemplos de símbolos del tipo "bar"

8.4.2. Tipo Gauge

Figura 84 – Ejemplos de símbolos del tipo “gauge”

8.4.3. Tipo Measure

Figura 85 – Ejemplos de símbolos del tipo “measure”

8.4.4. Tipo Static

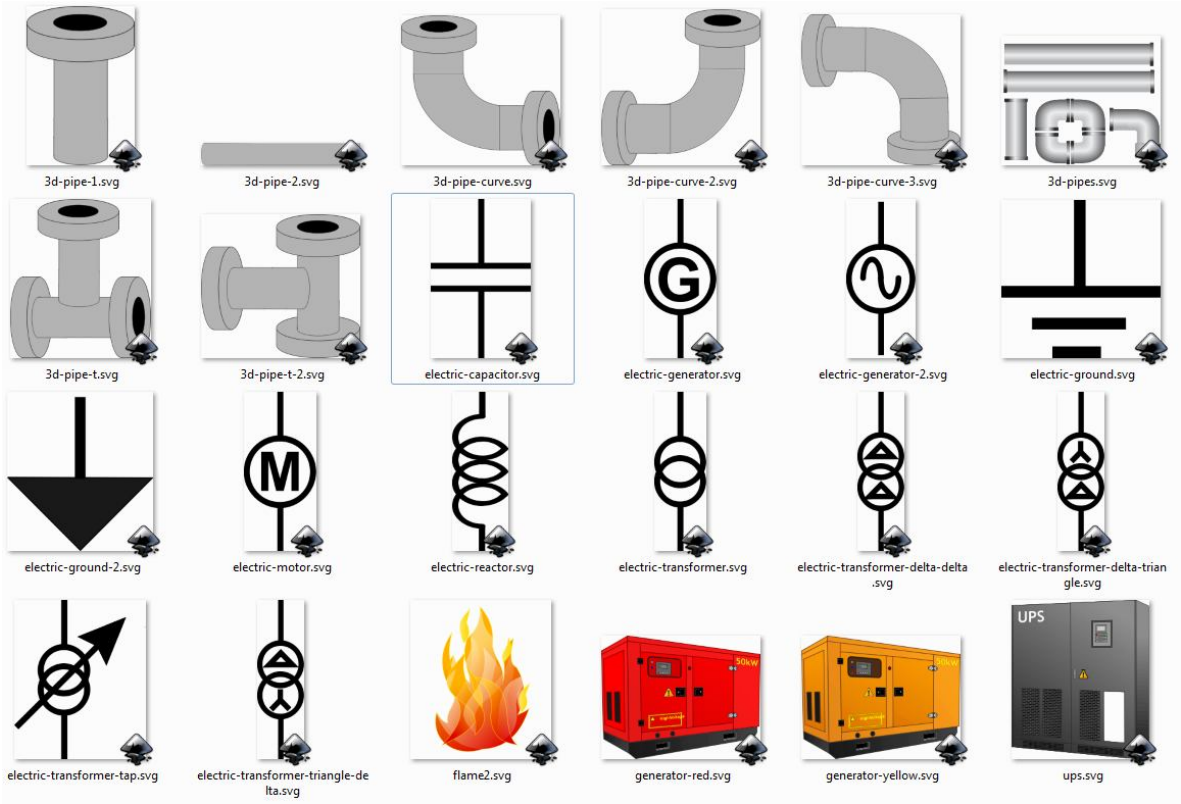


Figura 86 – Ejemplos de símbolos del tipo "static"

8.4.5. Tipo Status

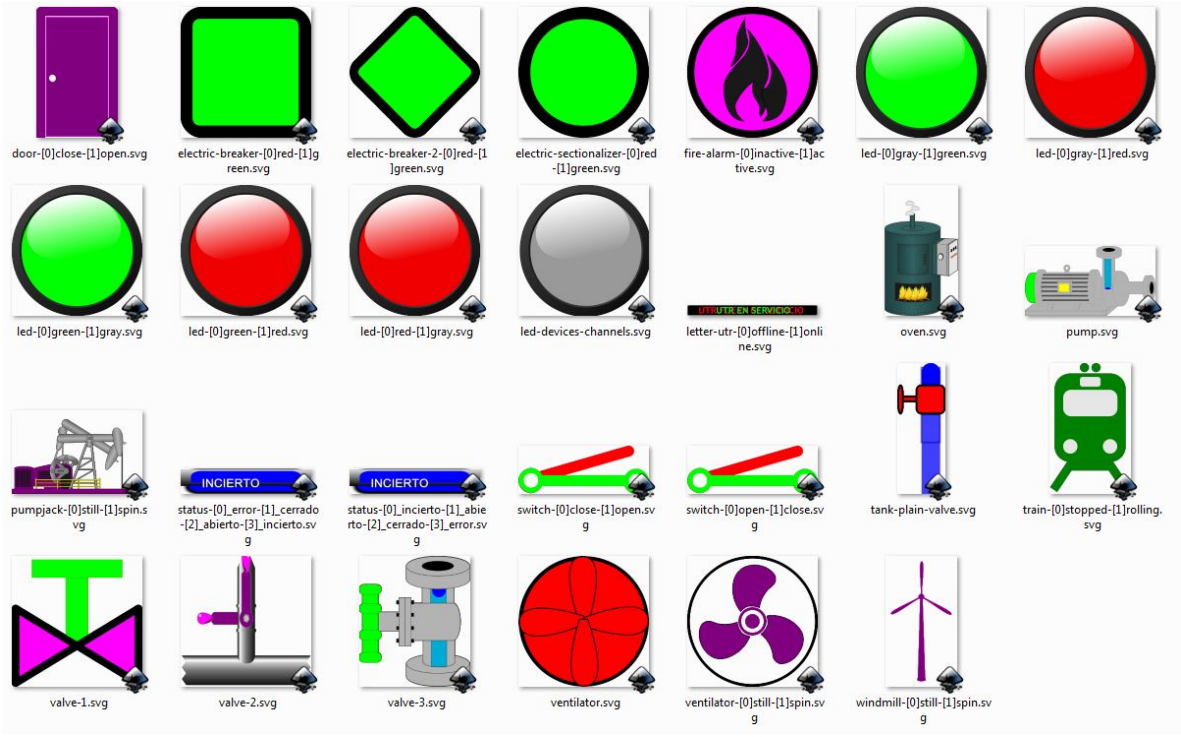


Figura 87 – Ejemplos del símbolos del tipo "status"

8.5. Edición de pantallas

Para la edición de pantallas el usuario dibuja el contenido estático (todo lo que no corresponda a símbolos, es decir el contenido decorativo de la pantalla) utilizando las herramientas de dibujo tradicionales de Inkscape. Para facilitar esta tarea existen muchas bibliotecas de archivos de imágenes vectoriales en formato SVG disponibles de forma gratuita. Sólo para citar algunos ejemplos encontramos:

- ✓ CLKER http://www.clker.com
- ✓ OPEN CLIPART https://openclipart.org
- ✓ 4 VECTOR http://4vector.com
- ✓ FREEPIK http://www.freepik.es

Más tarde se arrastran los archivos de los símbolos requeridos y se sueltan sobre la pantalla del programa, quedando estos incrustados automáticamente sobre el documento. El usuario luego los deberá ordenarlos colocando cada uno de ellos en la posición correcta.

Como resultado final se obtiene una pantalla como la siguiente, que combina símbolos del tipo bar, status, static, measure y gauge más algunos elementos adicionales estáticos como los textos indicadores:

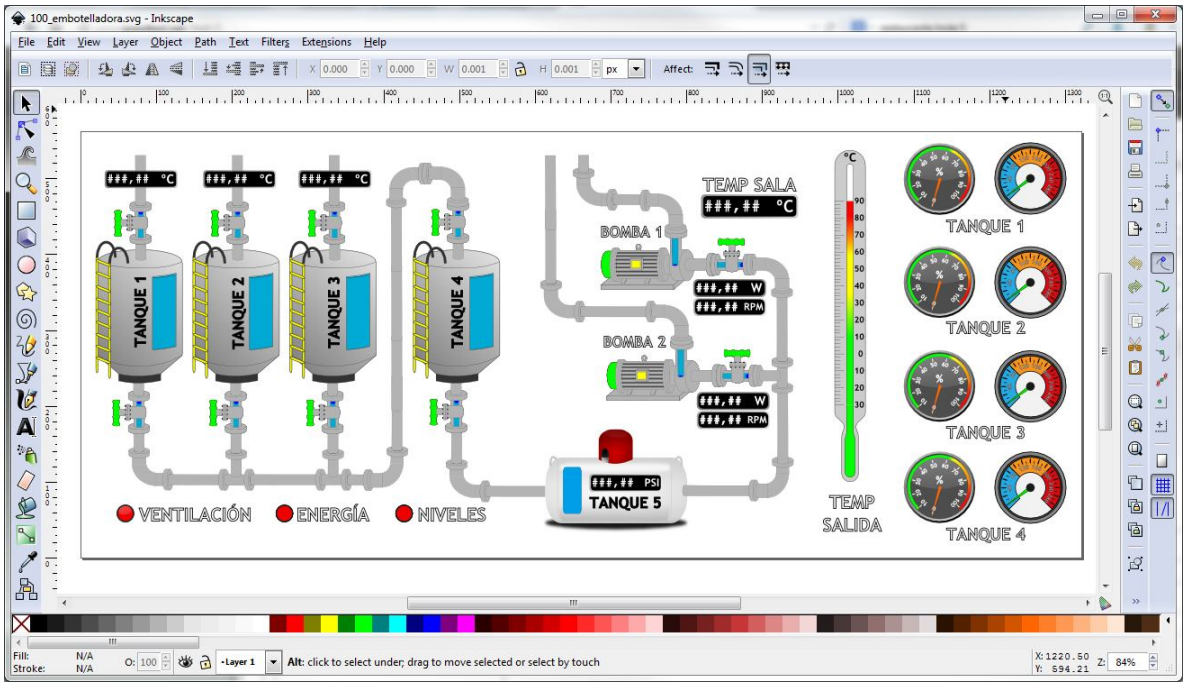


Figura 88 – Edición de pantallas del sistema en Inkscape

Esta pantalla es luego asignada al sistema desde la herramienta de Desarrollo y Configuración para luego desde la plataforma web especificar un nombre para cada uno de los símbolos. Estos nombres serán la identificación con la que el sistema conectará la parte gráfica con lo definido en la base de datos.

9. Capítulo IX: Consideraciones de seguridad

En los sistemas SCADA las consideraciones en cuanto a cuestiones de seguridad informática no han sido históricamente una preocupación.

Una de las razones es debido a que tradicionalmente se han utilizado vínculos propios para las comunicaciones. Estos vínculos eran físicamente de difícil acceso para intrusos y no se utilizaban redes públicas de comunicaciones las cuales estaban poco desarrolladas o directamente eran inexistentes.

Por su parte la poca difusión que estas tecnologías tenían al público en general hacían difícil que alguien que no fuera un especialista contara con los conocimientos necesarios para perpetrar un ataque informático.

En la actualidad con la popularización de Internet y las modernas redes de comunicaciones este precepto ya no es correcto. La flexibilidad que ofrecen las redes públicas y las facilidades que brindan han hecho que los sistemas SCADA en la mayoría de los casos opten por ellas para vincularse con sitios lejanos. Por su parte las tecnologías usadas en estos enlaces son en la mayoría de los casos de dominio público y de amplia difusión.

Asimismo es muy simple localizar en Internet información específica sobre sistemas de automatización, la mayoría de los manuales y especificaciones están disponibles en línea. En el pasado quienes tuvieran la motivación de realizar ataques destructivos contra redes SCADA no tenían los conocimientos de lo que estaban atacando. Esto ya no es cierto.

Estos aspectos combinados con la proliferación de organizaciones terroristas a nivel mundial y la aparición de comunidades de hacking hacen de las cuestiones de seguridad informática un tema ineludible.

9.1. El caso Stuxnet

Como ejemplo de este tipo de peligros puede citarse el caso del gusano informático Stuxnet.

Stuxnet es el primer malware diseñado específicamente diseñado para atacar sistemas de automatización y control industrial. Se estima que fue desarrollado en 2009 aunque no fue descubierto sino hasta 2010 por un fabricante de software antivirus llamado VirusBlokAda. Para ese entonces más de 100.000 computadoras y servidores habían sido infectados en todo el mundo.

El objetivo de este gusano eran los sistemas SCADA WinCC/PCS 7 de Siemens, específicamente el PLC Siemens Simatic S7, uno de los PLCs más utilizados a nivel mundial y empleado en una multitud de procesos industriales.

Su diseminación inicial es a través de memorias USB, pudiendo propagarse luego a través de redes locales. Se aprovecha de cuatro vulnerabilidades de día cero del sistema operativo

Windows para infectar el sistema. Una vez dentro utiliza las contraseñas por defecto de WinCC para tomar el control. A pesar de esto Siemens aconseja no cambiar las contraseñas predeterminadas de este producto para no afectar la estabilidad del sistema, esto no hace más que empeorar el panorama.

Una vez producida la infección su tarea consiste en la modificación de ciertos parámetros de funcionamiento del PLC, consiguiendo interferir en el proceso industrial sin que los operadores del sistema puedan notarlo. Es notable como sólo ataca a sistemas que tienen configurados unos parámetros muy determinados, puntualmente controladores de motores que giran entre 870Hz y 1210Hz. Esta frecuencia de giro tan alta sólo se utiliza en un número muy limitado de procesos industriales, por ejemplo en las centrifugadoras para enriquecer uranio utilizadas fundamentalmente en Irán, de hecho el 60% de las infecciones se localizaron en este país.

Esto hace pensar que fue un ataque intencional. Su complejidad y especificidad apuntan a que ha sido desarrollado por una organización grande y bien financiada. El diario The New York Times ha apuntado a los programas de defensa informática de Israel y Estados Unidos como los responsables de su creación con el objetivo de dificultar el desarrollo del programa de enriquecimiento de uranio de Irán.

Stuxnet ha sido el primer gusano de su clase, sin embargo no ha sido el único ni será el último. Variantes como Duqu o Flame han demostrado ser tan o más complejos. Las amenazas que antes eran terreno de películas de ciencia ficción ahora son una realidad.

9.2. Mitigación de riesgos

Sería necio pensar que el sistema que se ha detallado a lo largo de este trabajo no está expuesto a los riesgos de seguridad informática descriptos. Es por eso que se deben implementar contramedidas con el fin de mitigar las consecuencias de un posible ciberataque.

Por otro lado cabe destacar que estas medidas no son inherentes al sistema sino que están relacionadas con el entorno de trabajo en el que éste funciona. Es así que podrían aplicarse a cualquier sistema de control y automatización de similares características.

9.2.1. Segmentación de redes y uso de Firewalls

Un primer enfoque para proteger de ciberataques a un sistema SCADA que funcione dentro de una organización consiste en seccionar la conectividad de los equipos en tres segmentos de red LAN con distintos niveles de seguridad: Red Corporativa, Zona Desmilitarizada y Red de Tiempo Real. Se puede agregar a este esquema una cuarta Red de Control donde se ubicarían los dispositivos de control que operen dentro del edificio, aunque esta última no estaría aislada de la Red de Tiempo Real. El aislamiento se realiza a través de Firewalls o Cortafuegos. A continuación se describe con mayor detalle cada parte.

Red Corporativa

Esta red alberga todas las funciones empresarias y administrativas de la organización, contiene los puestos de trabajo de los empleados, servidores de correo electrónico, de dominio, de base

de datos, impresoras de red, servidores de almacenamiento, etc. Esta sería la típica red LAN que podríamos encontrar en cualquier empresa, incluso aunque esta no se dedique a actividades industriales. Puede contener además consolas remotas de operación.

Red de Tiempo Real

En esta red funciona el sistema SCADA y todos sus componentes, esto es servidores de históricos, servidores de bases de datos, servidores de aplicaciones, consolas de operación, servidores de comunicaciones, etc.

Zona Desmilitarizada

Esta red también es conocida como DMZ por la sigla en inglés Demilitarized Zone, su propósito es crear una zona neutral de separación entre la Red Corporativa y la Red de Tiempo Real.

Existen dos Firewalls que aíslan las tres redes hasta ahora expuestas, uno separa la Red Corporativa de la Zona Desmilitarizada, otro separa la Zona Desmilitarizada de la Red de Tiempo Real. En algunos casos se trata de un solo equipo que cumple ambas funciones, lo que simplifica la administración.

Las conexiones directas desde la Red Corporativa y hacia la Red de Tiempo Real no son permitidas o son extremadamente restringidas, dejando libres solo algunos puertos específicos de conexión y desde hosts determinados, esto aplica por ejemplo a las consolas de operación remota, que requieren tomar información fresca de la Red de Tiempo Real.

Para otros usos los usuarios de la Red Corporativa que requieran información de la Red de Tiempo Real se utiliza la Zona Desmilitarizada. El procedimiento consiste en el almacenamiento periódico de datos desde la Red de Tiempo Real hacia servidores de la Zona Desmilitarizada, por ejemplo en su servidor de históricos. Luego esta información queda disponible para ser accedida desde la Red Corporativa.

El propósito de esta estructura de red es brindar seguridad a la instalación, los sistemas de la red interna no están conectados de forma directa a Internet. Entonces cualquier ataque proveniente de Internet deberá ser capaz de superar las protecciones de tres Firewalls antes de llegar a la Red de Tiempo Real. Así aun cuando los puestos de trabajo de la Red Corporativa resulten infectados, situación probable teniendo en cuenta que estos requieren acceder regularmente a Internet por diversas cuestiones que hacen que el primer Firewall deba ser configurado de manera más laxa, es difícil que luego la infección se propague hacia la Zona Desmilitarizada y mucho más hacia la Red de Tiempo Real.

Si existiesen conexiones con infraestructuras lejanas, donde funcionen por ejemplo RTUs, estas deben implementarse mediante redes VPN punto a punto especialmente si estas se realizan a través de redes públicas. Los protocolos de telecontrol utilizados para la comunicación con las RTUs son encapsulados en general mediante el protocolo IPSEC.

La Figura 89 muestra un esquema con lo expuesto hasta el momento.

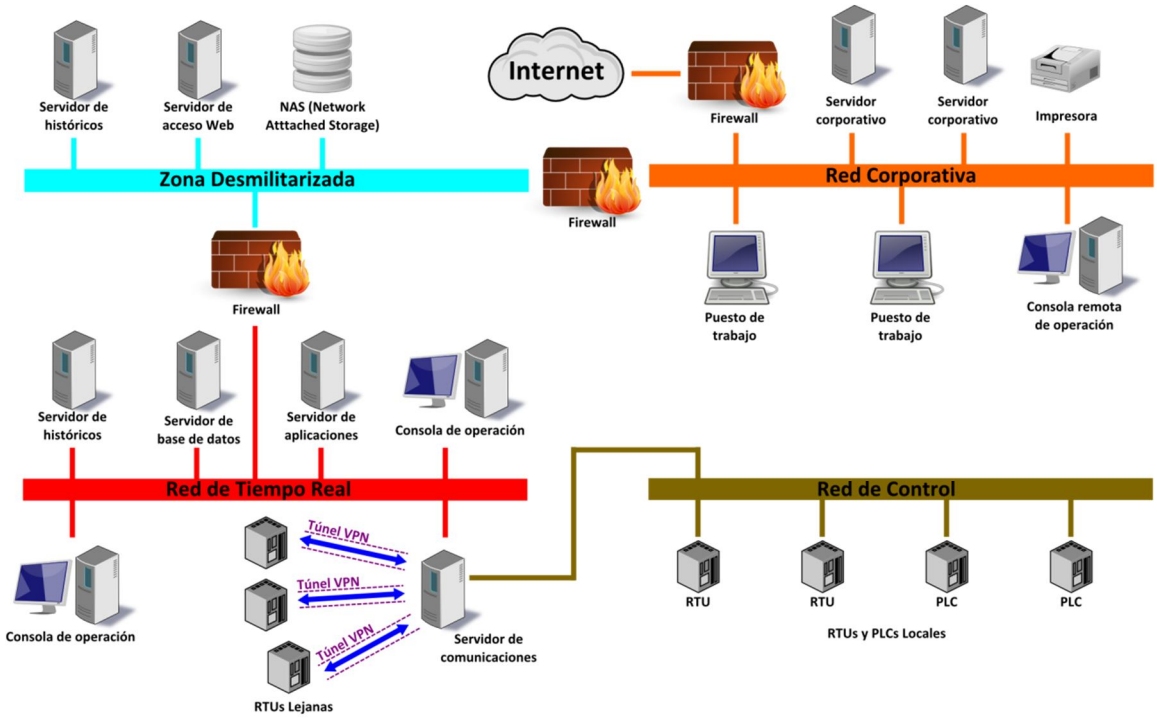


Figura 89 – Segmentación de redes en niveles de seguridad mediante utilización de firewalls

Falencias del esquema

En una primera aproximación este diseño de red puede parecer muy seguro, la existencia de Firewalls, la segmentación por niveles de seguridad y el aislamiento del sistema SCADA de Internet parece demostrarlo, sin embargo esto no siempre es así llevando a los administradores a una falsa sensación de seguridad no siendo conscientes de los riesgos que asumen.

Existen otros factores a tener en cuenta que transforman a este esquema en vulnerable:

- ✓ En la organización no existen o no se cumplen las políticas de uso aceptable de recursos, como memorias USB, computadoras portátiles, etc.
- ✓ Los sistemas antivirus no son actualizados con periodicidad.
- ✓ No se aplican las actualizaciones ni los parches de seguridad a los sistemas operativos.
- ✓ El acceso a Internet es libre y no se realiza ningún filtrado del contenido descargado de la web.

Por su parte la funcionalidad de los Firewalls podría no ser tan segura como se puede llegar a pensar, debido a que su tarea consiste fundamentalmente en el filtrado de tráfico de red en función de reglas que consisten simplemente en direcciones IP, dominios y puertos.

Imaginemos una situación en que un atacante externo toma el control de un puesto de trabajo de la Red Corporativa aprovechándose de reglas laxas de filtrado de tráfico web hacia esta red y de una vulnerabilidad del sistema operativo. Esta vulnerabilidad le permite inyectar un código malicioso que al ejecutarse se comunica con el atacante (“llama a casa”) y abre una línea de comandos con permisos de administrador.

Si este puesto de trabajo tiene permiso de acceso a la Zona Desmilitarizada el atacante, ahora con control total del sistema operativo del puesto de trabajo, puede repetir el procedimiento pero por ejemplo con el servidor de históricos aprovechándose de otra vulnerabilidad esta vez de este servidor.

Esto puede repetirse hasta alcanzar el sistema SCADA perpetrándose el ataque deseado y causando graves consecuencias.

9.2.2. Mejoras

Sistemas de prevención y de detección de intrusos

Estos sistemas conocidos como IPS e IDS respectivamente por sus siglas en inglés Intrusion Prevention System e Intrusion Detection System son elementos de seguridad que se encargan de monitorear en tiempo real el tráfico de redes en busca de patrones que evidencien comportamientos maliciosos. Una vez que la actividad maliciosa es identificada, de acuerdo al grado de criticidad o bien se registra, se reporta o se bloquea.

Esto mejora sustancialmente el nivel de seguridad de un Firewall que sólo se limita a analizar direcciones IP y puertos de origen y destino.

El funcionamiento habitual de estos dispositivos consiste en dejarlos correr durante un tiempo en un entorno sin anormalidades para que éste registre todo el tráfico de red en busca de patrones habituales de tráfico. Una vez terminada esta fase de aprendizaje se pasa al modo de operación normal en donde cualquier desviación del registro inicial genera alarmas.

También es posible definir cadenas de bytes que de ser encontradas dentro del tráfico dentro de un cierto contexto lanzan alertas, este tipo de detección requiere el ingreso constante de nuevos patrones o cadenas de bytes sospechosos.

Por su parte se pueden definir además políticas en base a direcciones IP, puertos, dominios o hosts como si se tratase de un Firewall.

Análisis de vulnerabilidades

Existen en el mercado diversos programas que realizan una auditoría de sistemas de todo tipo en busca de vulnerabilidades conocidas. Estos programas se dejan en ejecución durante un determinado tiempo en la red para que realicen un escaneo de todos los dispositivos. En cada uno de ellos buscará puertos de escucha y servicios como HTTP, FTP, SSH, etc.

A continuación se elabora un informe automático donde se detallan los hosts, sus sistemas operativos, sus aplicaciones, sus versiones, sus vulnerabilidades, etc. En base a esto el administrador puede determinar si usuarios están ejecutando aplicaciones no permitidas, si se requiere instalar parches de seguridad y en general cuáles son los puntos débiles de la red.

Es importante utilizar al menos dos de estas herramientas, hacerlo de forma periódica, mantenerlas actualizadas y por supuesto corregir las falencias detectadas lo antes posible.

9.2.3. Consideraciones respecto al sistema SCADA expuesto

Si bien se han planteado en este capítulo contramedidas de seguridad que podrían aplicarse de forma general a sistemas de telecontrol y automatización que funcionen sobre redes privadas y que tengan acceso a redes públicas, es menester también analizar ciertas características particulares y relativas al sistema que se ha expuesto a lo largo del presente trabajo.

Corrección de errores y vulnerabilidades

Como se ha visto el sistema se compone de módulos propios (núcleo, de configuración, HMI) y se apoya también en plataformas externas como PHP, el servidor web NGINX y el motor de base de datos PostgreSQL.

Todos estos elementos, debido a su naturaleza, son susceptibles a ser blanco de ataques informáticos que podrían poner en riesgo la operación global del SCADA aprovechándose de vulnerabilidades conocidas o desconocidas. Por ello es importante adoptar una política de mitigación de estas incidencias.

Habitualmente son publicadas nuevas versiones de las plataformas externas que incorporan corrección de errores de funcionamiento y solución de vulnerabilidades. Se ha adoptado como política que estas nuevas versiones reemplacen a las anteriores dentro del sistema, pasando primero por una etapa de pruebas en donde se verifica que el reemplazo no produzca ningún tipo de incompatibilidad con los módulos propios. Más tarde se genera un nuevo instalador actualizado y se distribuye entre los usuarios.

De la misma forma es habitual durante la operación normal de los módulos núcleo, de configuración o HMI, que surjan errores que no habían sido detectados previamente. Muchas veces son los mismos usuarios los que los descubren estos vicios ocultos cuando sin saberlo ponen a prueba el sistema al utilizarlo de una forma particular. Estos errores podrían ser aprovechados por un atacante para suspender la ejecución del sistema. Entonces ante reportes de usuarios de problemas de funcionamiento se procede a identificar claramente el problema y se realiza la corrección del mismo en el código fuente. Más tarde, en función de su criticidad, se decide si se lanza inmediatamente un nuevo instalador o se espera a que se realicen nuevas correcciones o mejoras.

El proceso de reinstalación del sistema es simple, sólo debe salvarse desde el módulo de desarrollo la configuración actual para luego proceder a la reinstalación. Posteriormente se carga la configuración antes salvada. El proceso toma en total unos cinco minutos, por lo que el impacto en la operación no es importante.

Otras consideraciones

Servidores

Dado que el sistema puede ser instalado en uno o más servidores es importante que los mismos cuenten con las medidas de protección necesarias para asegurar su correcta operación, esto comprende la instalación de antivirus actualizados y firewalls, así como también la aplicación de todos los parches de seguridad del sistema operativo y demás programas en operación.

Motor de base de datos

De forma predeterminada el motor de base de datos solo admite conexiones desde localhost, es decir que la misma debe originarse desde la misma computadora en donde el mismo se encuentre en operación, de esta forma los intentos de conexión desde otros nodos de la red serán rechazados. Esta medida de seguridad puede anularse en caso que se desee instalar el motor de base de datos en un servidor independiente si las características de la aplicación así lo requiriesen.

Almacenamiento de claves de usuario

La información de los usuarios es almacenada dentro de la base de datos en la tabla adm_users que más tarde será descripta junto al resto en un capítulo especial. Esta información incluye información sensible, como ser la contraseña de acceso al módulo de operación HMI.

Con el objetivo de proteger esta información las mismas son almacenadas utilizando el algoritmo de reducción criptográfico MD5. Cuando el usuario ingresa la contraseña en el módulo HMI se calcula el hash de la misma y se envía al servidor para ser comparado con el valor almacenado en la base de datos. De esta forma la contraseña permanece segura.

10. Capítulo X: Conceptos finales

10.1. Conclusiones

En el transcurso del presente trabajo se ha intentado proponer una respuesta a los interrogantes actuales en particular del sector de los sistemas HMI/SCADA y de la automatización y el control en general.

Se ha especificado, diseñado y expuesto un sistema HMI/SCADA multiplataforma, modular, interoperable con otros sistemas y redes de comunicaciones, con acceso web y basado en herramientas, bibliotecas y plataformas de código abierto.

El mismo pretendió subsanar en parte las falencias comunes que otros productos del mercado presentan en un sistema de bajo costo y al alcance de las pequeñas y medianas industrias y empresas de nuestro país. Las aplicaciones son muchas, se han enumerado reiteradas veces a lo largo del presente trabajo.

Al día de hoy ha alcanzado un grado de evolución suficiente para comenzar con su comercialización, bajo el nombre "EQUINOX Control Expert". Se publicita a través de la página web http://www.equinoxce.com/.

La empresa nacional MYEEL, líder local en la comercialización de medidores y equipos para aplicaciones de agua, gas y electricidad es uno de los distribuidores a nivel nacional del producto bajo el paraguas "MiDDe/SCADA".

Figura 90 – MiDDe SCADA, solución de telesupervisión de Myeel Basada en EQUINOX Control Expert

Puede consultarse información del producto a través del siguiente enlace: http://www.myeel.com.ar/novedad.php?novedad=MiDDE-Scada-nuevo-producto

Actualmente ha sido vendido e implementado en las cooperativas eléctricas de las ciudades argentinas de La Paz en la provincia de Entre Ríos, General Pico en provincia La Pampa y Plottier en la provincia de Neuquén para formar parte de los Sistemas de Telecontrol de las redes eléctricas de estas localidades.

Por su parte EIT Electric, una empresa del grupo EIT Group también posee una representación de EQUINOX y utiliza el producto en los proyectos de telemedición y control que la empresa lleva adelante. Consultar más información en http://eitelectric.co/es/category/servicios/.

Asimismo la empresa Draken de Uruguay, integra el producto en proyectos de telemedición de estaciones de bombeo y almacenamiento de la red de distribución de agua potable de ese país que la empresa OSE (Obras Sanitarias del Estado, http://www.ose.com.uy/) administra.

Estos son algunos de los ejemplos que prueban que el sistema posee la madurez necesaria para aplicaciones de esta índole. Se espera que con el tiempo se vayan incorporando nuevas funcionalidades y características que hagan posible afrontar desafíos mayores.

Haciendo una síntesis del estado actual y resumiendo las especificaciones del sistema podemos citar:

- ✓ Posee capacidad de definir hasta 1024 puntos o tags, lo cual es una limitación que podría extenderse utilizando servidores de hardware con mayores recursos.
- ✓ Puede usarse con sistemas operativos Windows y Linux debido a la característica multiplataforma de todas las herramientas involucradas en el desarrollo.
- ✓ Implementa protocolos de telecontrol Modbus RTU, Modbus TCP, DNP3 y DNP3 TCP/IP.
- ✓ Acceso multiusuario, hasta diez operadores pueden trabajar simultáneamente.
- ✓ Almacenamiento histórico de las indicaciones de todos los puntos del sistema cada diez minutos (valor predeterminado) hasta por seis meses.
- ✓ Motor de gráficas con hasta diez variables a representar por gráfica.

El desarrollo del sistema continúa día a día con el agregado de nuevas características muchas veces a partir de pedidos puntuales de clientes. Con el objetivo de establecer una visión a largo plazo se expone a continuación cuales son los factores que se deberían tener en cuenta como siguientes pasos en la evolución del producto.

10.2. Futuros pasos

Es notable cómo los sistemas SCADA son cada vez más importantes para varios sectores industriales por ejemplo el de la manufactura, los procesos industriales, así como en infraestructuras críticas como redes de distribución eléctrica inteligentes, sistemas inteligentes de transporte, etc.

Se analiza a continuación, teniendo en cuenta las tendencias y visiones expuestas a lo largo de la presente tesis, los factores claves en los que se considera que el sistema debería continuar su evolución, si tomamos en cuenta las posibilidades que se presentan en el ecosistema complejo colaborativo de dispositivos, sistemas y entidades interactuantes.

Supervisión por eventos

El monitoreo de activos es de clave importancia en una infraestructura heterogénea de alta complejidad. En sistemas de gran escala será imposible seguir adquiriendo información con los métodos tradicionales de interrogación de dispositivos. El enfoque más prometedor es tener una infraestructura impulsada por eventos unida a una arquitectura orientada a servicios. Entonces cada dispositivo o sistema será capaz de compartir la información que genera (datos, alarmas, etc.) como eventos a las entidades interesadas y también podrá componer y orquestar la información y los servicios de forma modelada, generando nuevos índices de monitoreo no previstos en la etapa de diseño de los sistemas componentes.

Gestión y mantenimiento a gran escala

La nueva generación de sistemas estará compuesta de cientos de dispositivos de control con diferentes configuraciones de hardware y software. Continuar manejando la configuración de estas infraestructuras de forma manual como se hace hoy en día será cada vez más engorroso. En ese sentido es imprescindible pensar en la automatización de esta tarea con una mecánica que permita descubrir dispositivos, sistemas y servicios ofrecidos por la infraestructura de forma dinámica. Debería ser posible realizar actualizaciones de software y tareas masivas de reprogramación y reconfiguración del sistema entero. Adicionalmente la visualización de la infraestructura real debe ser posible porque dará la oportunidad de un mejor entendimiento y mantenimiento de la misma. La creciente complejidad no permitirá la gestión por dispositivo, en consecuencia la automatización de tareas como la configuración, corrección, optimización, protección son deseables ya que facilitarían la gestión y el mantenimiento a gran escala.

Seguridad confianza y privacidad

Dado que los nuevos sistemas SCADA interactuarán cada vez más profundamente con otros sistemas y servicios en la nube, además de las actuales preocupaciones de seguridad, factores adicionales relacionados con las buenas prácticas en ese sentido, confianza y privacidad necesitan ser investigadas y tenidas en cuenta.

Actualmente la mayoría de los dispositivos y sistemas son operados bajo la órbita de una organización o entidad, una empresa o un usuario. Sin embargo cambiará en el futuro. La tendencia es que los sistemas se compongan de varios otros más simples administrados por diferentes entidades y posiblemente bajo distintos niveles de seguridad.

Procesamiento de información en tiempo real

La adquisición de información en tiempo real es una tarea difícil, sin embargo para que la próxima generación de aplicaciones pueda reaccionar en tiempo real se necesita además el procesamiento de información en tiempo real antes de ser transmitida. Esto incluye el posible pre-filtrado y pre-procesamiento de la información para un objetivo específico y el análisis complejo de eventos relevantes.

Ya que el procesamiento de eventos de tiempo real depende de varios pasos, se necesita abordar los desafíos planteados por estos como detección de patrones de eventos, abstracción de eventos, programación de eventos, filtrado de eventos, modelado de jerarquías de eventos, detección de relaciones (como causalidades, grupos o sincronismos) entre eventos, etc.

Simulación

Los ambientes industriales son complejos sistemas de sistemas. Por eso cualquier cambio a una parte de ellos puede tener resultados inesperados en otras partes dependientes o colaborativas. Sin embargo la evolución independiente de sistemas más pequeños es indispensable para lograr adaptabilidad y permitir la evolución. Entonces la emulación de los sistemas es altamente necesaria para poder identificar con suficiente anticipación los posibles conflictos y efectos secundarios. Estas simulaciones pueden ser usadas en el período pre-despliegue: evaluación del comportamiento ante cambios a ser aplicados y el monitoreo de ellos, así como también luego del despliegue y con el sistema en producción.

Mejor interoperabilidad

Como la siguiente generación de sistemas será altamente colaborativa y tendrá que intercambiar información, la interoperabilidad a través de comunicaciones abiertas y el intercambio estandarizado de datos es necesaria. La ingeniería de sistemas complejos e interoperables tiene un profundo impacto en su evolución, migración y futura integración con otros sistemas. Se espera que la futura infraestructura industrial evolucione constantemente. En consecuencia es importante que sea compatible hacia atrás para evitar corromper funcionalidades existentes así como compatible hacia adelante lo que implica diseñar interfaces e interacciones tan ricas como sea posible teniendo en cuenta futuras funcionalidades que pudieran surgir.

11. Anexo: Modelo de datos del sistema

La siguiente es una especificación de la estructura interna de la Base de Datos del sistema. Como ya se ha mencionado se ha escogido PostgreSQL como motor de base de datos. Dado que el mismo es un sistema de base de datos relacional el presente modelo debe seguir estos principios.

En la base de datos relacional existe un conjunto de tablas que contienen una serie de registros o datos. Los datos de ciertas tablas de conectan con datos de otras tablas a través de relaciones.

Las tablas pueden tener uno o más campos para identificar de forma única los registros que las componen. Es decir que no pueden existir dos registros con valores idénticos en estos campos. Los mismos se denominan "claves primarias". Asimismo existen las denominadas "claves foráneas" que no necesariamente son únicas pero que contienen referencias a claves de otras tablas.

Se han definido 38 tablas con diferentes funciones, para su mejor interpretación se han categorizado de la siguiente forma:

- ✓ Tabla de Puntos.
- ✓ Tablas de Pantallas del Sistema.
- ✓ Tablas de Adquisición de Datos.
- ✓ Tablas de Sets.
- ✓ Tablas de Telemandos.
- ✓ Tablas de Vistas.
- ✓ Tablas de Usuarios.
- ✓ Tablas de Listados del Sistema.
- ✓ Otras tablas.

Se presenta un diagrama con la estructura de cada tabla, detallando sus campos e indicando mediante flecha las relaciones entre las mismas. A continuación se explica en detalle el contenido de cada y la función de cada campo.

11.1. Diagrama de tablas y relaciones de la Base de Datos

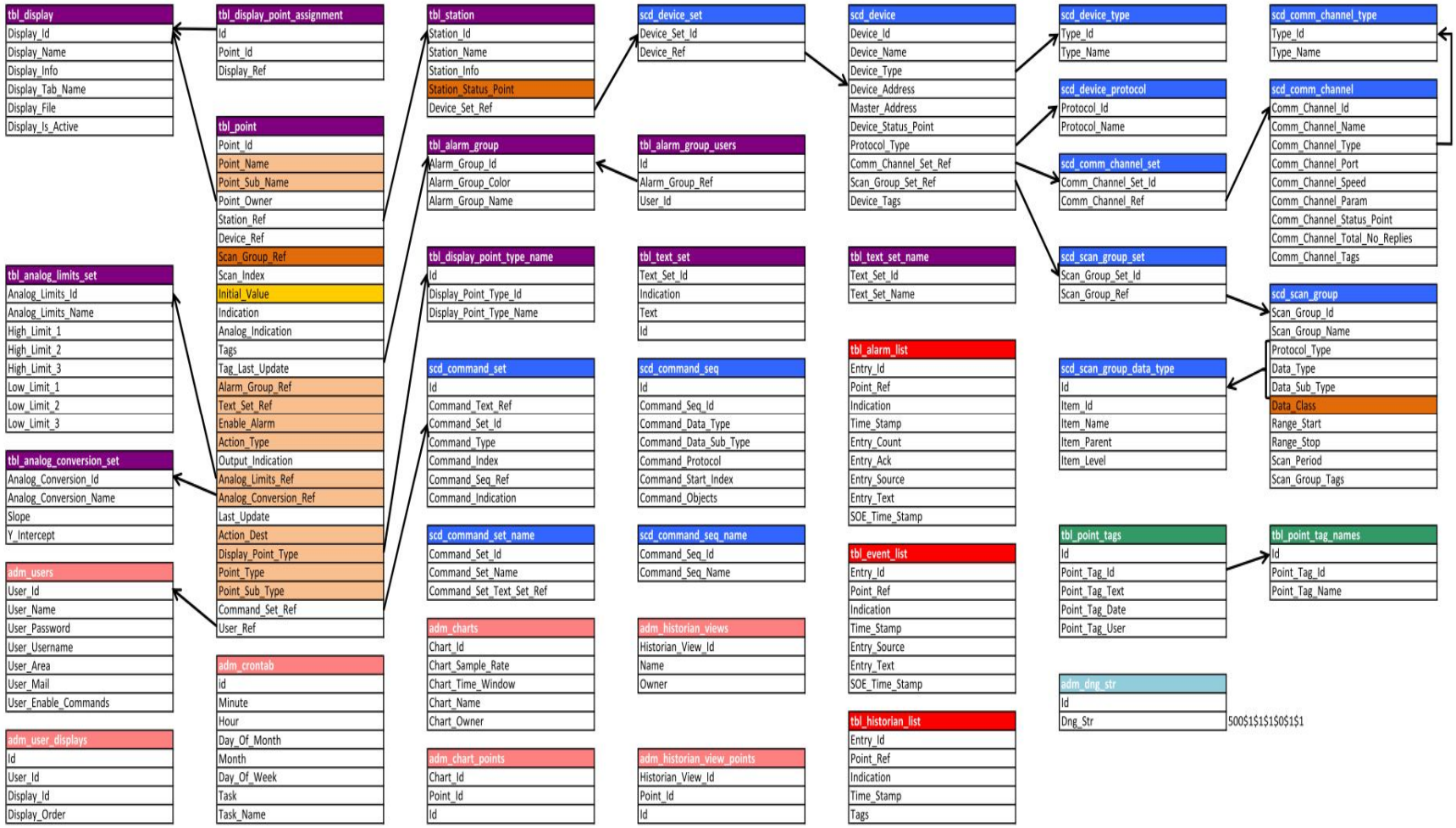


Figura 91 – Modelo de datos del sistema

11.2. Tabla de Puntos

Esta tabla contiene la configuración de todos los puntos del sistema, conteniendo para cada uno de ellos las definiciones necesarias para su normal funcionamiento. Su nombre es **tbl_point**. A continuación se detallan cada uno de estos campos y su función.

11.2.1. Campos generales

- ✓ **Point_Id**
  Este campo es la clave primaria de la tabla. Se trata de un valor numérico único que identifica a cada punto del sistema. Éste se incrementa a medida que se van creando puntos en el sistema. No pueden existir dos puntos que compartan el mismo **Point_Id**. Esto garantiza la identificación unívoca de cada uno de los puntos.

- ✓ **Point_Name / Point_Sub_Name**
  Campos utilizados para asignar una denominación al punto. Point_Name (Nombre de punto) se utiliza para nominar la categoría más general a la cual pertenece este punto y Point_Sub_Name (Sub-nombre de punto) a la más específica. Por ejemplo si queremos representar la temperatura, presión y altura de fluído en un tanque crearemos tres puntos asignando a estos campos lo siguiente:

	Point_Name	**Point_Sub_Name**
Punto Nº1	TANQUE 1	TEMPERATURA
Punto Nº2	TANQUE 1	PRESIÓN
Punto Nº3	TANQUE 1	ALTURA

  La combinación de nombre y sub-nombre es la que se utiliza para identificar los símbolos dentro del archivo gráfico de la pantalla a la cual pertenece el punto, es decir tanto los símbolos como los puntos son denominados igual, de manera que el sistema pueda ubicarlos dentro del archivo SVG de gráficos vectoriales para cambiar su aspecto de acuerdo a la indicación actual del punto al cual están relacionados.

- ✓ **Point_Owner**
  Define la pantalla que es propietaria del punto, es decir en la cual el punto se encuentra definido. Por su parte el sistema permite que el punto sea visualizado en pantallas adicionales, aunque siempre será propiedad de alguna de ellas.

  Para alterar la configuración del punto debemos hacerlo desde la pantalla propietaria del mismo.

11.2.2. Campos relacionados con la indicación del punto

- ✓ **Indication**
  Indicación o valor actual del punto. Se utiliza para puntos del tipo "**status**". Esta información se actualizará cuando se produzca un cambio en el valor leído desde el dispositivo al cual el punto se encuentra asignado.

- ✓ **Analog_Indication**
  Indicación o valor actual del punto. Se utiliza para puntos del tipo "**measure**", "**gauge**" o "**bar**". Esta información se actualizará cuando se produzca un cambio en el valor leído desde el dispositivo al cual el punto se encuentra asignado.

- ✓ **Tags**
  El campo Tags brinda información adicional acerca del estado actual del punto. A continuación e presenta la lista de posibles tags que pueden ser aplicados a los puntos:

  1. **Has Alarms**: El punto tiene alarmas activas, no reconocidas, relacionadas cambios en su estado.
  2. **Force Update**: Obliga al sistema a realizar una actualización del valor del punto aún cuando no haya ocurrido un cambio de estado en el mismo.
  3. **Execute**: La presencia de este tag indica que hay un telemando pendiente de ejecución sobre este punto.
  4. **Calculated**: El valor actual del punto ha sido calculado, es decir no corresponde al valor real leído desde el dispositivo.
  5. **Manual Entry**: El valor actual del punto ha sido introducido de forma manual por un operador del sistema, es decir no corresponde al valor real leído desde el dispositivo.
  6. **Information**: La presencia de este tag indica que existe un mensaje informativo asociado a este punto. Los mensajes son cargados por los operadores del sistema para dejar constancia de un suceso o situación de operación excepcional relacionada con este punto, por ejemplo: "fuera de servicio", "en reparación", "indica erróneamente", etc.
  7. **Command Inhibit**: Los telemandos asociados a este punto se encuentran inhibidos, por lo que no se permitirá su ejecución. Este tag es aplicado a discreción por los operadores del sistema.
  8. **Event Inhibit**: Los cambios de estado asociados al punto no serán registrados como eventos.
  9. **Alarm Inhibit**: Los cambios de estado asociados al punto no serán registrados como alarmas.
  10. **Not Renew**: Ha ocurrido un error al tratar de leer el valor de este punto, por lo que el valor actual es considerado como "no actualizado" o "no renovado".

- ✓ **Tags_Last_Update**
  Etiqueta de tiempo que indica cuando fueron actualizados por última vez los **Tags** del punto.

- ✓ **Last_Update**
  Etiqueta de tiempo de la última actualización que sufrió el punto. Puede referirse a un cambio en la indicación (campos **Indication** / **Analog_Indication**) o a un cambio de valor del campo **Tags**. Es utilizado por el sistema para actualizar el estado de los símbolos en las pantallas gráficas, donde sólo actualizará los que contengan una etiqueta de tiempo **Last_Update** que sea posterior o más reciente que la última registrada.

11.2.3. Campos relacionados con la adquisición de datos

- ✓ **Station_Ref / Device_Ref / Scan_Index**
  Estos tres campos relacionan al punto con la fuente de información que se utilizará para completar los campos **Indication** o **Analog_Indication** según corresponda.

  Esta información será leída a través de un canal de comunicación desde un dispositivo (que puede ser un PLC, una RTU, un IED, o de forma genérica cualquier elemento de adquisición de datos) especificado en el campo **Device_Ref**, este dispositivo pertenecerá a una estación especificada en el campo **Station_Ref**.

  El campo **Station_Ref** define la estación a la cual está asignado el punto. Las estaciones son entidades que determinan un área geográfica, de aplicación, o un tipo de elemento donde agrupar los distintos dispositivos del sistema. Su fin es simplemente organizativo permitiendo al operador del sistema tener una idea más acabada de la función de cada elemento de adquisición de datos conectado al sistema.

  Por último el campo **Scan_Index** define el índice, posición o registro dentro del dispositivo asignado desde el cual se leerá el valor o indicación actual del punto.

  La definición de **Scan_Index** se complementa con la definición del tipo de punto mediante los campos **Point_Type** y **Point_Sub_Type**. En ellos se determina la naturaleza del punto, es decir si se trata de un valor discreto de uno dos bits, un valor analógico de 16 o 32 bits, etc. Más adelante se explican estos campos con mayor detalle. Luego **Scan_Index** definirá específicamente, dentro del rango completo de entradas / salidas leídas desde el dispositivo, a cual estará asignado el punto.

11.2.4. Campos relacionados con la ejecución de acciones

- ✓ **Action_Type**
  Tipo de acción que el operador podrá realizar a través de este punto. Las opciones son "**Comando**" (para ejecutar una acción sobre un dispositivo conectado), "**Link**" (para acceder a otra pantalla del sistema al hacer clic sobre el punto) o "**Bloqueado**" para deshabilitar para inhibir cualquier tipo de acción.

- ✓ **Action_Dest**

Cuando **Action_Type** es definido como "**Link**" este campo especifica la pantalla a la cual se accederá al hacer clic sobre el punto.

- ✓ **Command_Set_Ref**
  Cuando **Action_Type** es definido como "**Comando**" este campo define el **Set de Comandos** asignado al punto. Un **Set de Comandos** es un conjunto de telemandos que se enviarán a un cierto dispositivo a través de un canal de comunicación cuando un operador del sistema lo desee.

- ✓ **Output_Indication**
  Campo usado para establecer el valor que debe enviarse en los telemandos.

- ✓ **User_Ref**
  Se utiliza para identificar al operador del sistema autor de la ejecución de un telemando.

11.2.5. Campos relacionados con la naturaleza del punto

- ✓ **Display_Point_Type**
  Naturaleza del punto desde el aspecto gráfico, determina el tipo de símbolo asociado al punto. Se presenta a continuación una lista de posibilidades:

1. **Status**: Símbolos de estado, su indicación puede tomar un número finito de estados, por ejemplo "0" y "1" o "0", "1", "2" y "3". Todos ellos tendrán una representación gráfica distinta, aunque solo uno es representado en pantalla en un determinado momento de acuerdo a la indicación actual del punto. Relacionados con entradas o salidas de naturaleza digital.
2. **Measure**: Símbolos correspondientes a representaciones numéricas de registros de entrada o salida del sistema. La indicación actual del punto se escala mediante el Set de Conversión definido para luego dibujarse en el símbolo. Relacionado con entradas o salidas de naturaleza analógica.
3. **Bar**: Símbolos esquematizados mediante barras o rectángulos cuyas dimensiones se ven alteradas en función de las variaciones en la indicación actual del punto. Relacionado con entradas o salidas de naturaleza analógica.
4. **Gauge**: Símbolos que representan medidores cuya manecilla cambia de ángulo en función de la indicación actual del punto. Relacionado con entradas o salidas de naturaleza analógica.
5. **Static**: Símbolos que no cambian de aspecto durante su utilización, aunque al relacionarse con un punto pueden utilizarse para accesos entre pantallas del sistema.
6. **Last Alarm**: Símbolo utilizado para mostrar en pantalla la última alarma ocurrida en un determinado dispositivo.

- ✓ **Point_Type / Point Sub Type**: Determinan la naturaleza del punto desde el punto de vista del dispositivo (PLC, RTU, etc.). A continuación se presenta una lista de combinaciones posibles de estos campos.

1. **Point_Type:** Digital Input / Digital Output
   Son puntos de naturaleza discreta, es decir que toman un número finito de estados posibles. Pueden asociarse sólo a símbolos del tipo "**Status**".
   a. **Point_Sub_Type:** Single Bit
      Tienen un bit de longitud, por lo tanto pueden tomar sólo dos estados: "0" o "1", correspondiendo cada uno de ellos a un "abierto" / "cerrado" de una válvula, a un "encendido" / "apagado" de un interruptor, etc.
   b. **Point_Sub_Type:** Double Bit
      Tienen dos bits de longitud, por lo tanto pueden tomar cuatro estados: "0", "1", "2", o "3". Un ejemplo de uso puede ser un motor con cuatro velocidades posibles, o un seccionador que puede estar tanto "abierto", "cerrado" como "en tránsito".
2. **Point_Type:** Register Input / Output 16
   Son registros de entrada o salida con una longitud de 16 bits, es decir de dos bytes. Pueden asociarse a símbolos del tipo "status", "measure", "bar" o "gauge".
   a. **Point_Sub_Type:** 8 Bits LSB
      Se aplica una máscara para tomar la parte baja (byte menos significativo) del registro de 16 bits.
   b. **Point_Sub_Type:** 8 Bits MSB
      Se aplica una máscara para tomar la parte alta (byte más significativo) del registro de 16 bits.
   c. **Point_Sub_Type:** 16 Bits
      Se toma el registro entero de 16 bits sin aplicar máscaras.
   d. **Point_Sub_Type:** 16 + 16 Bits
      Se toman dos registros consecutivos de 16 bits, por lo que se considerará un registro doble de 4 bytes de longitud.
   e. **Point_Sub_Type:** Bit 0 – Bit 15
      Se selecciona uno de los 16 bits de los dos bytes del registro.

   El ítem "e" podrá asignarse a símbolos "status". El resto podrán asignarse a símbolos "measure", "bar", o "gauge".

3. **Point_Type:** Register Input / Ouput 32
   Son registros de entrada o salida con una longitud de 32 bits, es decir de cuatro bytes. Pueden asociarse a símbolos del tipo "status", "measure", "bar" o "gauge".
   a. **Point_Sub_Type:** 8 Bits B0
      Se aplica una máscara para tomar el primer byte (menos significativo) del registro de 32 bits.

b. **Point_Sub_Type:** 8 Bits B1
   Se aplica una máscara para tomar el segundo byte del registro de 32 bits.
c. **Point_Sub_Type:** 8 Bits B2
   Se aplica una máscara para tomar el tercer byte del registro de 32 bits.
d. **Point_Sub_Type:** 8 Bits B3
   Se aplica una máscara para tomar el cuarto byte (más significativo) del registro de 32 bits.
e. **Point_Sub_Type:** 16 Bits Low
   Se aplica una máscara para tomar el primer y segundo byte del registro de 32 bits, formando un subregistro de 16 bits.
f. **Point_Sub_Type:** 16 Bits High
   Se aplica una máscara para tomar el tercer y cuarto byte del registro de 32 bits, formando un subregistro de 16 bits.
g. **Point_Sub_Type:** 32 Bits
   Se toma el registro entero de 32 bits sin aplicar máscaras.
h. **Point_Sub_Type:** Bit 0 – Bit 31
   Se selecciona uno de los 32 bits de los cuatro bytes del registro.

El ítem "h" podrá asignarse a símbolos "status". El resto podrán asignarse a símbolos "measure", "bar", o "gauge".

4. **Point_Type:** Calculation
   La indicación de estos puntos no es leída desde dispositivos sino que es calculada internamente por el sistema realizando operaciones matemáticas tomando como entrada de estos cálculos la indicación de otros puntos del sistema.

11.2.6. Otros campos

- ✓ **Alarm_Group_Ref**
  Referencia al Grupo de Alarma al cual pertenece el punto. Los Grupos de Alarma son categorías utilizadas para agrupar puntos del sistema de acuerdo a la criticidad que tenga un cambio de estado del mismo, por área o dispositivo al que pertenezcan, por tipo de señal, etc.
- ✓ **Text_Set_Ref**
  Referencia al Set de Texto asignado al punto. Los Sets de Texto determinan para cada estado posible de la indicación del punto un cierto texto que la defina, ejemplo: "0" es "ABIERTO" y "1" es "CERRADO" o "0" es "APAGADO" y "1" es "ENCENDIDO".
- ✓ **Analog_Limits_Ref**
  Referencia al Set de Límites asignado al punto. Este campo aplica solo para puntos del tipo "measure", "gauge" o "bar", es decir los de naturaleza analógica. Los Sets de Límites son conjuntos de valores que se contrastan en cada actualización con el valor actual del campo Analog_Indication. Si el valor de Analog_Indication supera o deja de superar a alguno de ellos se genera una alarma que alerta al operador del sistema de tal violación.

- ✓ **Analog_Conversion_Ref**
  Referencia al Set de Conversión asignado al punto. Este campo aplica solo para puntos del tipo "measure", "gauge" o "bar", es decir los de naturaleza analógica. Los Sets de Conversión son operaciones matemáticas que se aplican a los valores crudos (cuentas) leídos desde los dispositivos para convertirlos a unidades de ingeniería (m^3, kg, km/h, rpm, etc.) aptas para la interpretación del operador del sistema.
- ✓ **Enable_Alarm**
  Determina si los cambios en el punto generarán o no alarmas. Para el caso de los puntos del tipo "status" corresponde a cambios de estado y para los puntos del tipo "measure", "gauge", "bar" corresponde a violaciones de límites (de acuerdo al Set de Límites).

11.3. Tablas relacionadas con pantallas del sistema

11.3.1. Tabla tbl_display

Esta tabla define todas las características de los displays, es decir de las pantallas del sistema. Contiene los siguientes campos.

Campos

- ✓ **Display_Id**
  Valor numérico que identifica de forma única a una cierta pantalla del sistema. Este campo es referenciado desde **Point_Owner** en **tbl_point** y desde **Id** en **tbl_display_point_assignment**.
- ✓ **Display_Name**
  Nombre de la pantalla, usada para referencia del usuario.
- ✓ **Display_Info**
  Información sobre la pantalla, texto informativo usado para referencia del usuario.
- ✓ **Display_Tab_Name**
  En la interfaz gráfica para el acceso del operador las pantallas se pueden seleccionar mediante pestañas o tabs. Este campo determina al nombre que tendrá la pestaña correspondiente a la pantalla.
- ✓ **Display_File**
  Ruta del archivo de gráficos vectoriales SVG correspondiente a la pantalla.
- ✓ **Display_Is_Active**: Determina si la pantalla está activa o no, si no lo estuviera el operador del sistema no podrá utilizarla a pesar de tenerla asignada.

11.3.2. Tabla tbl_display_point_assignment

Esta tabla relaciona o asigna uno o más puntos con cada uno de los displays o pantallas del sistema. La tabla contendrá una entrada por punto que se asigne a una determinada pantalla. De esta forma el sistema podrá determinar al cargar un archivo gráfico vectorial SVG perteneciente a

un display, cuales son los puntos que deberá relacionar con los símbolos presentes en él, de forma de alterar su aspecto de acuerdo al valor o indicación actual de los puntos.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Point_Id**
  Identificación única de punto, coincidirá con la definida en el campo de nombre homónimo dentro de la tabla **tbl_point**.
- ✓ **Display_Ref**
  Referencia al campo **Display_Id** de la tabla **tbl_display**.

11.3.3. Tabla tbl_display_point_type_name

Tabla que contiene los nombres de los tipos de símbolos admitidos por el sistema, es decir "Measure", "Gauge", "Bar", "Status", "Static", "Last Alarm".

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Display_Point_Type_Id**
  Referencia al campo **Display_Point_Type** de la tabla **tbl_point.**
- ✓ **Display_Point_Type_Name**
  Nombre del tipo de símbolo.

11.4. Tablas relacionadas con la adquisición de datos

11.4.1. Tabla tbl_station

Esta tabla contiene la definición de las estaciones del sistema. Una estación es una categoría para organizar los dispositivos del sistema según su ubicación geográfica, área de aplicación, tipo de elemento, etc.

Campos

- ✓ **Station_Id**
  Valor numérico que identifica de forma única a la estación.
- ✓ **Station_Name**
  Nombre de la estación, usado para referencia del usuario.
- ✓ **Station_Info**
  Información sobre la estación, texto informativo usado para referencia del usuario.

- ✓ **Device_Set_Ref**
  Referencia al Set de Dispositivos asignados a la estación, según tabla **scd_device_set**.

11.4.2. Tabla scd_device

Tabla que contiene la definición de los dispositivos conectados al sistema mediando canales de comunicaciones. Un dispositivo puede ser un PLC, una RTU, un IED (Intelligent Electronic Device o Dispositivo Electrónico Inteligente) o cualquier otro elemento de adquisición de datos.

Campos

- ✓ **Device_Id**
  Valor numérico que identifica de forma única al dispositivo en el sistema.
- ✓ **Device_Name**
  Nombre del dispositivo, usado para referencia del usuario.
- ✓ **Device_Type**
  Referencia al tipo de dispositivo, según tipos definidos en tabla **scd_device_type**
- ✓ **Device_Address**
  Dirección de hardware del dispositivo
- ✓ **Master_Address**
  Dirección de hardware con que se debe interrogar a este dispositivo
- ✓ **Device_Status_Point**
  Este campo contiene el **Point_Id** (ver **tbl_point**) del punto de estado donde se reflejará el estado actual del dispositivo, es decir si se encuentra conectado, desconectado, tiene errores de definición, etc.
- ✓ **Protocol_Type**
  Referencia al protocolo de telecontrol con el que se interrogará al dispositivo, según tipos definidos en tabla **scd_device_protocol.**
- ✓ **Comm_Channel_Set_Ref**
  Referencia al Set de Canales asignado al dispositivo, según tabla **scd_comm_channel_set.**
- ✓ **Scan_Group_Set_Ref**
  Referencia al Scan Group o Grupo de Escaneo asignado al dispositivo, según tabla **scd_scan_group_set.**
- ✓ **Device_Tags**
  Etiquetas de propósito general que se utilizan para por ejemplo determinar si se mostrarán o no en la pantalla del módulo principal del sistema el intercambio de paquetes (frames) entre el dispositivo y el sistema.

11.4.3. Tabla scd_device_set

Esta tabla contiene la definición de un conjunto de dispositivos, conocidos como Sets de Dispositivos para ser asignados a las estaciones del sistema.

Campos

- ✓ **Device_Set_Id**
  Valor numérico identificativo al Set.
- ✓ **Device_Ref**
  Referencia al **Device_Id** del dispositivo de la tabla **scd_device**.

11.4.4. Tabla scd_device_type

Contiene un listado de tipos de elementos de adquisición de datos que son apropiados para su vinculación al sistema. Algunos ejemplos son “PLC”, “RTU”, “IED”, “Recloser”, “Line Protection”, etc.

Campos

- ✓ **Type_Id**
  Valor numérico que identifica de forma única el tipo de elemento.
- ✓ **Type_Name**
  Nombre o descripción del tipo de elemento.

11.4.5. Tabla scd_device_protocol

Lista de protocolos de telecontrol que entiende el sistema y mediante los cuales se comunica con los dispositivos. Por ejemplo Modbus TCP, Modbus RTU, DNP3, DNP3 over TCP/IP, etc.

Campos

- ✓ **Protocol_Id**
  Valor numérico que identifica de forma única el protocolo de telecontrol.
- ✓ **Protocol_Name**
  Nombre o descripción del protocolo de telecontrol.

11.4.6. Tabla scd_comm_channel

Tabla que contiene la definición de los canales de comunicación mediante los cuales el sistema se comunica con los dispositivos.

Campos

- ✓ **Comm_Channel_Id**
  Valor numérico que identifica de forma única al canal de comunicación.
- ✓ **Comm_Channel_Name**
  Nombre del canal de comunicación, usado para referencia del usuario.
- ✓ **Comm_Channel_Type**

Referencia al tipo de canal de comunicación, según tipos definidos en tabla **scd_comm_channel_type**

- ✓ **Comm_Channel_Port**
  Puerto de comunicaciones correspondiente al canal, por ejemplo "COM2", "/dev/tty0" para canales seriales o "192.168.0.33:502" para canales tipo TCP/IP.
- ✓ **Comm_Channel_Speed**
  Velocidad del canal de comunicaciones en baudios, este campo aplica sólo a los canales del tipo seriales.
- ✓ **Comm_Channel_Param**
  Parámetros adicionales a definir para el canal, por ejemplo en el caso de puertos seriales se define la paridad, los bits de datos y los bits de stop.
- ✓ **Comm_Channel_Status_Point**
  Este campo contiene el **Point_Id** (ver **tbl_point**) del punto de estado donde se reflejará el estado actual del canal de comunicación, es decir si se encuentra conectado, desconectado, tiene errores de definición, etc.
- ✓ **Comm_Channel_Total_No_Replies**
  El sistema cuenta las no respuestas desde los dispositivos a los cuales interroga por cada canal posible de comunicación con los mismos. Este valor se guarda en este campo y tiene fines estadísticos.
- ✓ **Comm_Channel_Tags**
  Etiquetas de propósito general que se utilizan para por ejemplo determinar si se mostrarán o no en la pantalla del módulo principal del sistema el intercambio de paquetes (frames) a través de este canal.

11.4.7. Tabla scd_comm_channel_set

Esta tabla contiene la definición de un conjunto de canales de comunicación, conocidos como Sets de Canales de Comunicación para ser asignados a los dispositivos del sistema.

Campos

- ✓ **Comm_Channel_Set_Id**
  Valor numérico identificativo del Set.
- ✓ **Comm_Channel_Ref**
  Referencia al **Comm_Channel_Id** del canal de comunicación de la tabla **scd_comm_channel.**

11.4.8. Tabla scd_comm_channel_type

Contiene un listado de tipos de canales de comunicación. Por ejemplo: "Serial Port", "TCP/IP Socket".

Campos

- ✓ **Type_Id**
  Valor numérico que identifica de forma única el tipo de canal.
- ✓ **Type_Name**
  Nombre o descripción del tipo de canal.

11.4.9. Tabla scd_scan_group

Tabla que contiene la definición de los grupos de escaneo. Los grupos de escaneo son definiciones de formatos de interrogaciones o peticiones que el sistema envía a los dispositivos para obtener información específica de ellos.

Campos

- ✓ **Scan_Group_Id**
  Valor numérico que identifica de forma única al grupo de escaneo.
- ✓ **Scan_Group_Name**
  Nombre del canal de comunicación, usado para referencia del usuario.
- ✓ **Protocol_Type**
  Protocolo de telecontrol que se utilizará para realizar la interrogación.
- ✓ **Data_Type**
  Campo que define el tipo de dato a requerir desde el dispositivo, genéricamente pueden ser entradas o salidas tanto analógicas como digitales.
- ✓ **Data_Sub_Type**
  Campo que determina el sub-tipo de dato completando la definición del campo anterior. Por ejemplo entradas analógicas de 16 bits o 32 bits. Este campo solo aplica a algunos protocolos de telecontrol.
- ✓ **Range_Start**
  Primer índice, número o posición del registro o entrada digital de entrada o salida requerido/a desde el dispositivo.
- ✓ **Range_Stop**
  Último índice, número o posición del registro o entrada digital de entrada o salida requerido/a desde el dispositivo.
- ✓ **Scan_Period**
  Tiempo en segundos que el sistema esperará para enviar una nueva interrogación.
- ✓ **Scan_Group_Tags**
  Etiquetas de propósito general que se utilizan para por ejemplo determinar si se mostrarán o no en la pantalla del módulo principal del sistema el intercambio de paquetes (frames) relacionados a este grupo de escaneo.

11.4.10. Tabla scd_scan_group_set

Esta tabla contiene la definición de un conjunto de grupos de escaneo, conocidos como Sets de Grupos de Escaneo para ser asignados a los dispositivos del sistema definiendo así los formatos de las interrogaciones o peticiones que el sistema envía.

Campos

- ✓ **Scan_Group_Set_Id**
  Valor numérico identificativo del Set.
- ✓ **Scan_Group_Ref**
  Referencia al **Scan_Group_Id** del Grupo de Escaneo de la tabla **scd_scan_group.**

11.4.11. Tabla scd_scan_group_data_type

Esta tabla contiene el detalle de los nombres de los protocolos de telecontrol y de sus tipos y sub-tipos de datos. Estos nombres se utilizan para referencia del usuario en la definición de los grupos de escaneo.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Item_Level**
  Determina si el Item es corresponde a un protocolo de telecontrol (valor 0), un tipo de dato (valor 1) o un sub-tipo de dato (valor 2).
- ✓ **Item_Id**
  Identificación numérica del protocolo, tipo o sub-tipo de dato, su interpretación depende del protocolo de telecontrol. Por ejemplo: en los tipos de dato Modbus este valor coincide con los códigos de función del protocolo.
- ✓ **Item_Parent**
  Especifica el Item_Id del padre de este Item. En los sub-tipos de datos este valor corresponderá a un tipo de dato, por su parte en los tipos de datos el valor corresponderá a un protocolo de telecontrol.
- ✓ **Item_Name**
  Nombre del tipo, sub-tipo o protocolo según corresponda.

11.5. Tablas de Sets

11.5.1. Tabla tbl_alarm_group

Contiene la definición de los Grupos de Alarma. Los **Grupos de Alarma** son categorías utilizadas para agrupar puntos del sistema de acuerdo a la criticidad que tenga un cambio de estado del mismo, por área o dispositivo al que pertenezcan, por tipo de señal, etc.

Campos

- ✓ **Alarm_Group_Id**
  Valor numérico que identifica de forma única al Grupo de Alarma.
- ✓ **Alarm_Group_Color**
  Color del Grupo de Alarma. El mismo se utilizará para colorear el texto de las alarmas relacionadas con este ítem en los listados de alarmas del sistema.
- ✓ **Alarm_Group_Name**
  Nombre del Grupo de Alarma, usado solo para referencia del usuario.

11.5.2. Tabla tbl_alarm_group_users

Se utiliza para alertar a los usuarios mediante correo electrónico de la aparición de nuevas alarmas pertenecientes a un determinado Grupo.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Alarm_Group_Ref**
  Referencia a un cierto Grupo de Alarma
- ✓ **User_Id**
  Referencia al usuario que será alertado por correo electrónico

11.5.3. Tabla tbl_analog_limits_set

Cada entrada de esta tabla define un Set de Límites. Los **Sets de Límites** son conjuntos de valores que se contrastan en cada actualización con el valor actual del campo **Analog_Indication**. Si el valor de **Analog_Indication** supera o deja de superar a alguno de ellos se genera una alarma que alerta al operador del sistema de tal violación. Estos Sets aplican solo para puntos del tipo "**measure**", "**gauge**" o "**bar**", es decir los de naturaleza analógica.

Cada Set de este tipo define en total seis valores: seis límites superiores y seis inferiores.

Campos

- ✓ **Analog_Limits_Id**
  Valor numérico que identifica de forma única al Set de Límites.
- ✓ **Analog_Limits_Name**
  Nombre del Set de Límites, usado solo para referencia del usuario.
- ✓ **High_Limit_1**
  Límite Superior 1
- ✓ **High_Limit_2**
  Límite Superior 2
- ✓ **High_Limit_3**
  Límite Superior 3
- ✓ **Low_Limit_1**
  Límite Inferior 1
- ✓ **Low_Limit_2**
  Límite Inferior 2
- ✓ **Low_Limit_3**
  Límite Inferior 3

11.5.4. Tabla tbl_analog_conversion_set

Cada entrada de esta tabla define un Set de Conversión. Los **Sets de Conversión** son operaciones matemáticas que se aplican a los valores crudos (cuentas) leídos desde los dispositivos para convertirlos a unidades de ingeniería (m^3, kg, km/h, rpm, etc.) aptas para la interpretación del operador del sistema. Estos Sets solo aplican para puntos del tipo "**measure**", "**gauge**" o "**bar**", es decir los de naturaleza analógica.

Por el momento la única operación matemática definida es la de una recta, es decir y = ax + b, siendo "a" la pendiente y "b" la ordenada al origen. La variable "x" corresponde al valor leído desde el dispositivo e "y" es el que será presentado en pantalla.

Campos

- ✓ **Analog_Conversion_Id**
  Valor numérico que identifica de forma única al Set de Conversión.
- ✓ **Analog_Conversion_Name**
  Nombre del Set de Límites, usado solo para referencia del usuario.
- ✓ **Slope**
  Pendiente de la recta
- ✓ **Y_Intercept**
  Ordenada al origen de la recta

11.5.5. Tabla tbl_text_set

Tabla que contiene las definiciones de los Sets de Texto. Los **Sets de Texto** determinan para cada estado posible de la indicación de un punto un cierto texto que la defina, ejemplo: "0" es "ABIERTO" y "1" es "CERRADO" o "0" es "APAGADO" y "1" es "ENCENDIDO".

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Text_Set_Id**
  Valor numérico de identificación del Set de Conversión.
- ✓ **Indication**
  Valor numérico al que se desea asignar un cierto texto.
- ✓ **Text**
  Texto correspondiente a la indicación.

11.5.6. Tabla tbl_text_set_name

Se utiliza para almacenar los nombres de los Sets de Texto definidos en la tabla **tbl_text_set**.

Campos

- ✓ **Text_Set_Id**
  Valor numérico que identifica de forma única al Set de Conversión.
- ✓ **Text_Set_Name**
  Nombre del Set de Límites, usado solo para referencia del usuario.

11.6. Tablas de relacionadas a telemandos

11.6.1. Tabla scd_command_set

Esta tabla contiene definiciones de Sets de Comandos. Un **Set de Comandos** es un conjunto de telemandos que se enviarán a un cierto dispositivo a través de un canal de comunicación cuando un operador del sistema lo desee.

Existen dos tipos de telemandos que pueden definirse dentro de un Set de Comandos: Los **Comandos simples** permiten realizar cambios de salidas digitales del dispositivo, por su parte los **Comandos complejos secuenciales** son ejecuciones secuenciales de uno o más telemandos complejos.

Los telemandos complejos se utilizan para alterar el estado de bits en registros de salida o para establecer su valor en función de otros puntos del sistema o de valores fijos, o para cambiar de estado varias salidas digitales.

Cada Set de Comandos contiene una combinación de **Comandos Simples** y **Comandos complejos secuenciales** aunque podría ser un conjunto de sólo uno de ellos.

Los Sets de Comandos utilizan un Set de Texto pero solo para nominar cada uno de los telemandos que lo integran. Así por ejemplo se le presentarán al operador las opciones de "ARRANQUE" y "PARADA" relativos a una bomba y ambos estarán relacionados ya sea a un **Comando Simple** o a un **Comando complejo secuencial**. Esta definición se establece en la tabla **scd_command_set_name**.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Command_Set_Id**
  Valor numérico de identificación del Set de Conversión.
- ✓ **Command_Type**
  Tipo de comando: Comando simple o Comando complejo secuencial.
- ✓ **Command_Text_Ref**
  Posición del Set de Texto (definido en tabla **scd_command_set_name**) que se utilizará para nominar este comando.
- ✓ **Command_Index**
  Este campo aplica solo para Comandos Simples, especifica el índice, número de orden o posición de la salida digital sobre la que se desea actuar.
- ✓ **Command_Indication**
  Este campo aplica solo para Comandos Simples, especifica el valor numérico o indicación que debe ser escrito en la salida digital especificada en el campo anterior.
- ✓ **Command_Seq_Ref**
  Este campo solo aplica para Comandos complejos secuenciales y almacena una referencia a un Comando complejo secuencial de la tabla.

11.6.2. Tabla scd_command_set_name

Esta tabla contiene la definición de los nombres de cada Set de Comandos y el Set de Texto utilizado para nominar a los telemandos que lo integren.

Campos

- ✓ **Command_Set_Id**
  Valor numérico que identifica de forma única al Set de Comandos.
- ✓ **Command_Set_Name**

Nombre del Set de Límites, usado solo para referencia del usuario.

- ✓ **Command_Set_Text_Set_Ref**
  Referencia al Set de Texto que será utilizado para nominar a cada uno de los telemandos que formen parte del Set de Comandos.

11.6.3. Tabla scd_command_seq

Definición de los Sets de Comandos complejos secuenciales del sistema. Estos comandos son un grupo de telemandos que ejecutan acciones complejas sobre dispositivos del sistema.

Estas acciones comprenden:

- ✓ Alterar el estado de una salida digital (funcionalidad equivalente a la de los Comandos Simples)
- ✓ Alterar el estado de varias salidas digitales (en una sola transacción de datos con el dispositivo)
- ✓ Alterar el estado de ciertos bits de uno o varios registros de salida.
- ✓ Establecer el contenido de uno o varios registros de salida con valores fijos.
- ✓ Establecer el contenido de uno o varios registros de salida a partir del valor de otros puntos o de valores ingresados por el operador del sistema.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Command_Seq_Id**
  Valor numérico que identifica de forma única al Set de Comandos.
- ✓ **Command_Data_Type**
  Tipo de telemando complejo que se ejecutará sobre el dispositivo
- ✓ **Command_Start_Index**
  Índice, posición o número de orden del primer registro o salida digital sobre la que se realizará la acción.
- ✓ **Command_Objects**
  Información adicional necesaria de acuerdo al tipo de telemando complejo.

11.6.4. Tabla scd_command_seq_name

Esta tabla contiene la definición de los nombres de cada Set de Comandos complejos secuenciales del sistema.

Campos

- ✓ **Command_Seq_Id**
  Valor numérico que identifica de forma única al Set de Comandos.

- ✓ **Command_Seq_Name**
  Nombre del Set de Límites, usado solo para referencia del usuario.

11.7. Tablas relacionadas a vistas

11.7.1. Tabla adm_charts

Tabla utilizada para almacenar las definiciones de Vistas Gráficas del sistema. Una **Vista Gráfica** es una representación temporal en un diagrama de ejes cartesianos con la evolución de la indicación de uno o varios puntos del sistema.

Campos

- ✓ **Chart_Id**
  Valor numérico que identifica de forma única a la gráfica.
- ✓ **Chart_Name**
  Nombre de la gráfica, usado solo para referencia del usuario.
- ✓ **Chart_Sample_Rate**
  Tasa de refresco de la gráfica, determina cada cuanto tiempo se tomará una muestra de valores de los puntos que lo integran.
- ✓ **Chart_Time_Window**
  Ventana de tiempo de la gráfica que será presentada en pantalla.
- ✓ **Chart_Owner**
  Referencia al operador o usuario creador de la gráfica.

11.7.2. Tabla adm_charts_points

En esta tabla se almacena la lista de puntos asignados a cada Vista Gráfica del sistema.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Chart_Id**
  Valor numérico para identificación de la Vista Gráfica.
- ✓ **Point_Id**
  Referencia a uno de los puntos integrantes de la Vista Gráfica.

11.7.3. Tabla adm_historian_view

Tabla utilizada para almacenar las definiciones de Vistas Históricas del sistema. Las Vistas Históricas son representaciones en formato de tabla de valores de la indicación pasada de uno o más puntos del sistema.

Campos

- ✓ **Historian_View_Id**
  Valor numérico que identifica de forma única a la Vista Histórica.
- ✓ **Name**
  Nombre de la Vista Histórica, usado solo para referencia del usuario.
- ✓ **Owner**
  Referencia al operador o usuario creador de la Vista Histórica.

11.7.4. Tabla adm_historian_view_points

En esta tabla se almacena la lista de puntos asignados a cada Vista Histórica del sistema.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Historian_View_Id**
  Valor numérico para identificación de la Vista Histórica.
- ✓ **Point_Id**
  Referencia a uno de los puntos integrantes de la Vista Histórica.

11.8. Tablas de usuarios

Tabla administrativa utilizada para almacenar los usuarios u operadores autorizados para utilizar el sistema.

11.8.1. Tabla adm_users

Campos

- ✓ **User_Id**
  Valor numérico que identifica de forma única al usuario.
- ✓ **User_Name**
  Nombre completo del usuario.
- ✓ **User_Username**
  Nombre de usuario para ingreso a través de la interfaz web.
- ✓ **User_Password**
  Contraseña para ingreso a través de la interfaz web.
- ✓ **User_Area**
  Área de la organización donde el usuario trabaja.
- ✓ **User_Mail**
  Correo electrónico del usuario

- ✓ **User_Enable_Commands**
  Habilita o deshabilita la ejecución de telemandos para el usuario

11.8.2. Tabla adm_user_displays

Esta tabla define los displays o pantallas gráficas a las que cada usuario u operador del sistema tendrá acceso.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **User_Id**
  Referencia al usuario u operador.
- ✓ **Display_Id**
  Referencia al display o pantalla gráfica.
- ✓ **Display_Order**
  Número de orden de la pestaña relacionada con a la pantalla.

11.9. Tablas de listados del sistema

11.9.1. Tabla tbl_alarm_list y tbl_event_list

La tabla **tbl_alarm_list** es utilizada para almacenar la Lista de Alarmas del sistema. Esta lista es un registro de todos los cambios de indicación en los puntos del tipo "**status**" y las violaciones de límites de los puntos del tipo "**measure**", "**gauge**" o "**bar**". Esto aplica a todos los puntos de esta condición excepto aquellos que tengan específicamente inhibida la generación de alarmas. La Lista de Alarmas brinda al operador del sistema información de situaciones que requieran atención inmediata.

Por su parte la tabla tbl_event_list almacena la Lista de Eventos. Esta lista contiene los mismos registros de la Lista de Alarmas pero además agrega cualquier otra información relativa a la operación del sistema sea o no crítica incluyendo los cambios de estado de los elementos que tienen inhibida la generación de alarma. Ejemplos de generación de eventos que no general alarma son por ejemplo ante la aplicación de tags a puntos, entrada o salida de usuarios del sistema, etc.

Campos comunes tbl_alarm_list y tbl_event_list

- ✓ **Entry_Id**
  Valor numérico que identifica de forma única cada entrada de la tabla, en este caso particular a cada alarma o evento de la tabla.
- ✓ **Point_Ref**

Referencia al punto del sistema sobre el que se generó la alarma o el evento

- ✓ **Indication**
  Para los puntos del tipo "status" representa la indicación del mismo luego del cambio de estado. Para los puntos "measure", "gauge" o "bar" es un valor numérico que indica cuál de los seis valores límites (tres superiores y tres inferiores) fue violado.
- ✓ **Time_Stamp**
  Etiqueta de tiempo que indica la fecha y la hora cuando se produjo la alarma o el evento. Se toma del reloj interno del hardware donde se encuentre funcionando el sistema.
- ✓ **Entry_Source**
  Fuente de la alarma, expresa el módulo del sistema desde el cual se generó la alarma.
- ✓ **Entry_Text**
  Texto adicional que amplía la información de la condición de alarma.
- ✓ **SOE_Time_Stamp**
  Para los protocolos de telecontrol que soporten manejo de etiquetas de tiempo se utiliza este campo para almacenar la fecha y hora registrada en el dispositivo cuando se produjo la alarma, lo que supone una precisión mayor a la hora de determinar el momento exacto en que se produjo el suceso.

Campo adicional de tbl_alarm_list

- ✓ **Entry_Ack**
  Indica si algún operador del sistema efectuó el reconocimiento de la alarma, lo que debería ocurrir al tomar conocimiento de la situación de alarma detallada. Este campo es el que separa en la interfaz web

11.9.2. Tabla tbl_historian_list

En esta tabla se registra en forma periódica una fotografía del estado actual de todos los puntos definidos en el sistema, esto incluye la indicación y los tags. El período de forma predeterminada es de 10 minutos. Tomando datos desde esta tabla se construyen las Vistas Históricas, siendo éstas arreglos de puntos para la presentación de información sobre los mismos a lo largo del tiempo en formato de tabla.

Campos

- ✓ **Entry_Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Point_Ref**
  Referencia al punto del que se realiza el registro.
- ✓ **Indication**

Indicación del punto al momento de tomar el registro. Para puntos del tipo “measure”, “gauge”, o “bar” corresponde al campo **Analog_Indication** de la tabla tbl_point, mientras que para los puntos “status” corresponde al campo **Indication** de la misma tabla.

- ✓ **Tags**
  Tags del punto al momento de tomar el registro.
- ✓ **Time_Stamp**
  Etiqueta de tiempo que especifica la fecha y hora cuando se realizó el registro

11.10. Otras tablas

11.10.1. Tabla adm_crontab

Tabla utilizada para la ejecución de tareas programadas relativas a la operación del sistema. Consiste en la ejecución de scripts o de programas accesorios que realizan tareas variadas, como chequeos de integridad de la base de datos, operaciones de limpieza de archivos de datos viejos, vaciado de tablas, etc. Se buscó que el modo de operación fuera similar al del comando “crontab” de Unix.

Campos

- ✓ **Id**
  Valor numérico que identifica de forma única cada entrada de la tabla.
- ✓ **Task**
  Ruta al script o archivo ejecutable de la tarea.
- ✓ **Task_Name**
  Nombre de la tarea, usado solo para referencia del usuario.
- ✓ **Minute**
  Minuto en la que se ejecutará la tarea (“*” equivale a todos los minutos).
- ✓ **Hour**
  Hora en la que se ejecutará la tarea (“*” equivale a todas las horas).
- ✓ **Day_Of_Month**
  Día del mes en que se ejecutará la tarea (“*” equivale a todos los días).
- ✓ **Month**
  Mes en que se ejecutará la tarea (“*” equivale a todos los meses).
- ✓ **Day_Of_Week**
  Día de la semana en que se ejecutará la tarea (“*” equivale a todos los días).

12. Bibliografía

El siguiente es el listado de libros, artículos y documentación consultada para la elaboración del presente trabajo de tesis.

12.1. Libros

[1] Boyer, Stuart A. SCADA : Supervisory Control and Data Acquisition. Research Triangle Park, NC: ISA-The Instrumentation, Systems, and Automation Society, 2004.

[2] Bailey, David, and Edwin Wright. Practical SCADA for industry. Amsterdam London: Elsevier, 2003.

[3] Geschwinde, Ewald, and Schönig. PostgreSQL developer's handbook. Indianapolis, IN: Sams, 2002.

[4] Blanchette, Jasmin, and Mark Summerfield. C++ GUI programming with Qt 4. Upper Saddle River, NJ: Pearson Hall in association with Trolltech Press, 2006.

[5] Lane, Jonathan. Foundation Website creation with HTML5, CSS3, and JavaScript. Berkeley, Calif. New York: friends of ED Distributed to the book trade worldwide by Springer Science+Business Media, 2012.

[6] Flanagan, David. JavaScript : the definitive guide. Beijing Sebastopol, CA: O'Reilly, 2011.

[7] Lengstorf, Jason. Pro PHP and jQuery. New York, N.Y: Apress Distributed by Springer Science+Business Media, LLC, 2010.

[8] Kirsanov, Dmitry. The book of Inkscape the definitive guide to the free graphics editor. *San Francisco: No Starch Press, 2009.*

[9] Hudson, Paul. PHP in a nutshell. Sebastopol, Calif: O'Reilly, 2006.

[10] Tatroe, Kevin, Rasmus Lerdorf, and Peter MacIntyre. Programming PHP. Sebastopol, CA: O'Reilly Media, 2013

[11] Peña, Joan. Comunicaciones en el entorno industrial. Barcelona: Editorial UOC, 2003.

[12] Perez, Enrique. Tecnologías y redes de transmisión de datos. México: Limusa/Noriega, 2003.

12.2. Artículos

[1] Stamatis Karnouskos and Armando Walter Colombo. Architecting the next generation of service-based SCADA/DCS system of systems. *Proceedings IECON 2011 37th Annual Conference of the IEEE Industrial Electronics Society, Crown Conference Centre, Melbourne, Australia, 07-10 November, 2011*. Piscataway, NJ: IEEE, 2011, páginas 360 y 362 a 364.

[2] Z. A. Vale, Member IEEE, H. Morais, Student Member IEEE, M. Silva and C. Ramos, Member IEEE 1. Towards a future SCADA. *IEEE Power & Engineering Society General Meeting 2009 Conference Theme : Investment in workforce and innovation for power systems : July 26-30, 2009, Calgary Telus Convention Centre, Calgary, Alberta, Canada*. Piscataway, N.J: IEEE, 2009, páginas 1 a 4.

[3] Jian WANG, Wei WANG, Yingnan WANG,Weifu QI,Xianjun GAO. A Study on SCADA Graphics Elements Library Based on SVG and Batik, publicado en *2010 Asia-Pacific Power and Energy Engineering Conference proceedings : March 28-31, 2010, Chengdu, China*. Piscataway, NJ: IEEE, 2010, páginas 1 a 3.

[4] Ning Cai, Jidong Wang and Xinghuo Yu. SCADA System Security: Complexity, History and New Developments. Proceedings, INDIN 2008 6th IEEE International Conference on Industrial Informatics : July 13-16, 2008, Daejeon Convention Center, Daejeon, Korea. Piscataway, N.J: IEEE Xplore, 2008, páginas 569 a 572.

12.3. Documentación en línea

[1] Recomendación SVG
http://www.w3c.org/TR/2011/REC-SVG11-20110816/
Último acceso: Abril de 2015.

[2] Estándar ECMAScript
http://www.ecma-international.org/ecma-262/5.1/
Último acceso: Abril de 2015.

[3] Documentación JSON
http://json.org/
Último acceso: Abril de 2015.

[4] Documentación PostgreSQL
http://www.postgresql.org/docs/9.4/interactive/index.html

http://www.postgresql.org/message-id/20041203184254.F6767@cookie.varlena.com

http://www.postgresql.org/about/

Último acceso: Abril de 2015.

[5] Documentación PHP
http://php.net/manual/es/
Último acceso: Abril de 2015.

[6] Documentación Qt Framework
http://doc.qt.io/
Último acceso: Abril de 2015.

[7] Documentación Inkscape
http://wiki.inkscape.org/wiki/index.php/Inkscape

Último acceso: Abril de 2015.

12.4. Normas

[1] IEC/TR 60870-1-1 ed1.0. Telecontrol equipment and systems. Part 1: General considerations. Section One: General principles

13. Glosario de términos

Adquisición de datos: Procedimiento mediante el cual un dispositivo electrónico toma cierta información de magnitudes físicas del mundo real mediante el uso de sensores apropiados.

Almacenamiento Histórico: Característica de los sistemas HMI/SCADA que permite el registro periódico de mediciones, estados e informaciones varias para su posterior análisis.

ATM: Asynchronous Transfer Mode o Modo de Transferencia Asíncrono, redes sucesoras de la tecnología Frame Relay. Tecnología de conmutación de banda ancha orientada a conexión y con capacidad de multiplexado.

Backplane: Placa de circuito impreso utilizada para interconectar y comunicar módulos o tarjetas que se enchufan a él.

Biblioteca/framework: Ambiente de software universal que mediante módulos y artefactos provee herramientas y funcionalidades que facilitan el desarrollo de aplicaciones de software.

C++: Lenguaje de programación que extiende la funcionalidad del lenguaje C orientándolo al objetos. Diseñado en la década del ochenta por Bjarme Stroustrup.

Centro de Control: Infraestructura edilicia donde se encuentran varios de los elementos que conforman un Sistema de Telecontrol, entre ellos el sistema SCADA.

Ciberseguridad: Conjunto de medidas y políticas de seguridad que una organización puede implementar para defenderse de ataques cibernéticos a través de las redes de telecomunicaciones modernas.

CME: El Centro de Movimiento de Energía era el Centro de Control de alta tensión de la empresa Segba dedicado a la tele-supervisión de la red de distribución de la ciudad de Buenos Aires y el conurbano. En la década del noventa se produjo la privatización de la red, se constituyó una sociedad anónima privada.

Consola de Operación: Computador utilizado por los operadores de sistemas SCADA para interactuar con los procesos que el sistema supervisa.

DCE: Data Communication Equipment o Equipo de Comunicaciones de Datos, es un módem que realiza tareas de conversión de señales, codificación, sincronismo, modulación, demodulación, etc.

Disponibilidad: Porcentaje de tiempo en que un sistema está operativo y listo para ser utilizado en toda su capacidad.

DNP3: Protocolo de telecontrol para comunicaciones entre RTUs y Estaciones Maestras o Centros de Control.

DTE: Data Terminal Equipment o Equipo Terminal de Datos, es una terminal o computadora que se conecta mediante un cable con un DCE. Convierte información del usuario en señales y viceversa.

ECMA: European Computer Manufacturers Association o Asociación Europea de Fabricantes de Computadores es una organización internacional privada sin fines de lucro que promueve estándares para sistemas de información y comunicación.

Enlace de Fibra Óptica: Medio de enlace que emplea fibras ópticas, hilo extremadamente fino de materiales plásticos o vidrio, y que mediante el envío de pulsos de luz por este material permite transmitir información entre dos puntos distantes con enorme ancho de banda, pequeña atenuación e inmunidad al ruido.

Enlace de Radio: Método de conexión implementado para transmitir información mediante el uso de transmisores y receptores de ondas de radio utilizando uno o más métodos de modulación y demodulación.

Enlace GPRS: Método de conexión para transmitir información que utiliza el servicio General Packet Radio Service o Servicio General de Paquetes vía Radio (GPRS) brindado por las empresas de redes móviles.

Enlace multipunto: Conexión de comunicaciones entre varios nodos que comparten una línea de transmisión de datos. Existe un nodo maestro que interroga a los demás nodos que cumplen el rol de esclavos. Cada nodo posee una dirección que lo identifica de los demás.

Enlace Par de Cobre: Medio de enlace para transmitir información entre equipos de redes de datos que emplea conductores eléctricos de cobre de grosor entre 0,3 y 3mm recubiertos de una vaina protectora. Para mejorar la capacidad de transmisión pueden trenzarse y apantallarse.

Enlace punto a punto: Conexión de comunicaciones únicamente entre dos nodos. Cada nodo puede actuar como esclavo o maestro en un determinado instante, recibiendo o transmitiendo información por el enlace.

Enlace Satelital: Método de conexión para transmitir información que usa un satélite como medio de enlace.

Entradas Analógicas: Entradas eléctricas a un PLC o RTU que a través de contactos especiales se conectan mediante cableados a válvulas, motores, etc. y se utilizan para determinar diferentes variables del mismo dentro de un intervalo continuo definido (velocidad, temperatura, etc.).

Entradas Discretas o Digitales: Entradas eléctricas a un PLC o RTU que a través de contactos especiales se conectan mediante cableados a válvulas, motores, etc. y se utilizan para determinar el su estado actual de funcionamiento dentro de un conjunto finito de posibilidades (encendido, apagado, etc.).

ERP: Enterprise Resource Planning o Planificación de Recursos Empresarios son sistemas orientados a producir información útil para las áreas gerenciales de la organización sobre aspectos operativos, contables, de existencias o stock y de logística entre otros.

Escalabilidad: Posibilidades de crecimiento de la capacidad del sistema para acompañar una expansión o un aumento de la complejidad del proceso bajo supervisión.

Estación Maestra: Estación desde la cual se colecta información de telemetría interrogando a una o varias RTUs a través de vínculos de comunicaciones.

Frame Relay: Sucesor del protocolo X.25 en la década del 90 y muy similar a éste pero con mayor capacidad de transporte y menores controles de verificación de errores que ya no fueron necesarios por la evolución de los medios de enlace.

GUI: Graphic User Interface o Interfaz Gráfica de Usuario es un programa que permite la interacción de un usuario con una aplicación mediante imágenes y objetos gráficos y periféricos de entrada y salida.

HMI: Human Machine Interface o Interfaz Hombre Máquina. Sección de los Sistemas de Telecontrol destinada a la interacción entre el operador del sistema y el proceso bajo supervisión mediante periféricos como pantallas, teclados, mouses, tableros lumínicos, señales visuales, etc.

IED: Intelligent Electronic Device o Dispositivo Electrónico Inteligente. Sigla utilizada genéricamente para definir a dispositivos computarizados con capacidades de realizar funciones automáticas y transmitir magnitudes y estados a otros sistemas.

IETF: Internet Engineering Task Force o Grupo de Trabajo de Ingeniería de Internet. Organización sin fines de lucro auspiciada por la Internet Society que desarrolla y promueve estándares de Internet abiertos. No posee miembros formales y sus publicaciones están disponibles para el público en general.

Interfaz de Usuario: Herramienta mediante la cual el operador del sistema interactúa con el proceso en análisis. Nexo entre el hombre y la máquina. Ver HMI.

JavaScript: Estándar de facto para la programación de aplicaciones Web del lado cliente permitiendo interactuar con el usuario, comunicarse asincrónicamente y controlar el navegador. Estandarizado por el comité de ECMA como ECMAScript.

JQuery: Biblioteca JavaScript multiplataforma destinada a simplificar la programación de scripts del lado cliente. Su primera versión fue creada por John Resig en el año 2006.

JSON: JavaScript Object Notation o Notación de Objetos JavaScript. Formato abierto de intercambio de datos, liviano y también fácil de entender y escribir por personas.

Licencia GPL: General Public License o Licencia Pública General es una licencia de software libre que garantiza a los usuarios la libertad de usar, estudiar, compartir y modificar el código fuente de

la aplicación. Fue escrita por Richard Stallman en 1989. Es la licencia de software libre más difundida del mundo.

Licencia LGPL: Lesser General Public License o Licencia Pública General Reducida es una licencia de software libre similar a GPL excepto que permite a los desarrolladores de software integrar en sus productos código LGPL sin el requerimiento de revelar el código fuente de sus aplicaciones.

Listado de Alarmas: Listado o registro de situaciones de distinto grado de criticidad que hayan ocurrido dentro del ámbito de supervisión del Sistema de Telecontrol y que requieren atención de un operador.

Listado de Eventos: Listado o registro de situaciones críticas que hayan ocurrido dentro del ámbito de supervisión del Sistema de Telecontrol.

MES: Manufacturing Execution System o Sistema de Ejecución de Manufactura son plataformas diseñadas para mejorar las actividades operativas mediante la generación de informes de producción, mantenimiento, entre otros.

Modbus: Protocolo de telecontrol de arquitectura cliente-servidor desarrollado por la empresa Modicon en 1979.

Motor de Base de Datos: Herramienta de software utilizada para el almacenamiento sistemático de toda la información recolectada por el sistema del proceso industrial en análisis para su posterior uso.

MPLS: Multiprotocol Label Switching o Conmutación Multiprotocolo mediante Etiquetas es una tecnología de transporte de datos que está reemplazando paulatinamente a las redes ATM y Frame Relay definida en la publicación RFC3031 de la IETF.

Operador: Persona encargada de llevar a cabo las tareas operativas de un Sistema de Telecontrol, es decir la supervisión de pantallas, el envío de telemandos, la toma de decisiones ante condiciones de alarmas, entre otras tareas.

Pantalla de operación: Representación gráfica de un determinado proceso industrial bajo supervisión la cual es alterada dinámicamente en función de cambios en el mismo.

PHP: Lenguaje de programación de scripts que opera del lado servidor, procesando peticiones de clientes para generar páginas HTML. Su primera versión fue creada por Rasmus Lerdorf en 1995.

PLC: Programmable Logic Controller o Controlador Lógico Programable. Dispositivo computarizado utilizado para la automatización de procesos industriales.

PostgreSQL: Sistema de gestión de base de datos relacional lanzado por primera vez en 1986. Su arquitectura es reconocida por su confiabilidad y por su robustez en cuanto a integridad de la información.

Punto: En los sistemas SCADA los puntos son unidades de información leídas desde los dispositivos de automatización, como PLCs o RTUs, conectados al sistema para su posterior representación gráfica asociándolo a un símbolo.

Qt: Biblioteca de desarrollo de aplicaciones multiplataforma con interfaz de usuario que usa C++ estándar con extensiones. Actualmente es propiedad de la empresa Digia quien lo distribuye con de forma comercial y también con licencia LGPL.

RDBMS: Relational Database Management System o Sistema de Gestión de Bases de Datos Relacionales es un programa de gestión de bases de datos relacionales, es decir que contienen tablas que se conectan entre sí mediante relaciones.

Redundancia: Característica mediante la cual se dispone de varios equipos o procesos para mejorar la performance de operación o para reducir el riesgo de indisponibilidad o mal funcionamiento.

RS-232: Estándar de comunicaciones, también conocido como EIA-232, seriales definido por la EIA (Electronic Industries Alliance o Alianza de Industrias Electrónicas) en 1969. Define las características eléctricas, tiempos y las señales usadas para conectar un DTE y un DCE, asimismo establece los conectores y su pinout.

RS-422: Estándar de comunicaciones seriales definido por la EIA (Electronic Industries Alliance o Alianza de Industrias Electrónicas). Define las características eléctricas de transmisores y receptores usados en sistemas de comunicaciones digitales balanceados multipunto. Los vínculos RS-422 permiten cubrir distancias de hasta 1500 metros y trabajan bien en ambientes eléctricamente ruidosos. Su diferencia respecto a RS-485 es que se admite sólo un transmisor y hasta 10 receptores.

RS-485: Estándar de comunicaciones seriales definido por la EIA (Electronic Industries Alliance o Alianza de Industrias Electrónicas). Define las características eléctricas de transmisores y receptores usados en sistemas de comunicaciones digitales balanceados multipunto. Los vínculos RS-485 permiten cubrir distancias de hasta 1200 metros y trabajan bien en ambientes eléctricamente ruidosos. Extiende las capacidades de RS-422 permitiendo la comunicación de hasta 32 dispositivos en una misma línea de datos, donde todos ellos pueden transmitir y recibir.

RTU: Remote Terminal Unit o Unidad Terminal Remota. Dispositivo computarizado usado para transmitir telemetría de magnitudes físicas hacia sistemas SCADA.

Salidas Analógicas: Salidas eléctricas que un PLC o de una RTU que de acuerdo a una orden interna o externa pueden, mediante el uso de señales PWN (Pulse Width Modulation o Modulación de Ancho de Pulso), maniobrar equipamiento como válvulas, motores, interruptores, etc. y alterar variables de los mismos como caudal, velocidad, etc.

Salidas Discretas o Digitales: Salidas eléctricas que un PLC o de una RTU puede poner en estado energización o des-energización de acuerdo a una orden interna o externa para maniobrar equipamiento como válvulas, motores, interruptores, etc. y así encenderlo, apagarlo, abrirlo, cerrarlo, etc.

SCADA: Supervisory Control and Data Acquisition o Control de Supervisión y Adquisición de datos.

Segba: Servicios Eléctricos del Gran Buenos Aires fue una empresa pública que hasta su privatización fue la encargada de las tareas de generación, transporte y distribución de energía eléctrica en un área geográfica que incluía a la ciudad de Buenos Aires y el conurbano.

Símbolo: En los sistemas SCADA un símbolo es una agrupación de elementos gráficos destinados a representar de forma pictórica un determinado elemento, variable, parámetro o magnitud del mundo real.

Sistema de Comunicaciones: Conjunto equipamiento electrónico, como redes, switches, routers, transmisores, receptores, DTEs, DCEs, etc., que utilizando diversos medios de enlace permite la interoperación y conexión de otros sistemas.

SQL: Structured Query Language o Lenguaje Estructurado de Consulta es un lenguaje especialmente desarrollado para el manejo de información en bases de datos del tipo RDBMS.

SVG: Scalable Vector Graphic o Gráficos Vectoriales Escalables. Un formato de imágenes vectoriales basado en XML con soporte para interactividad y animación desarrollado como estándar abierto por la W3C en 1999.

Tag: En algunos sistemas SCADA se hace referencia a Tags en lugar de Puntos, sin embargo el concepto es mismo, ver “Puntos”.

Telemetría: Acción de medición de magnitudes físicas a distancia para su transmisión a un sitio lejano.

Terminal Server: Equipamiento de comunicaciones que permite conectar dispositivos seriales RS-232 o RS-485 a redes de área local (LAN). Son útiles para la adaptación de dispositivos de automatización antiguos a redes de transporte y comunicaciones modernas.

UTR: Remote Terminal Unit o Unidad Terminal Remota. Dispositivo computarizado usado para transmitir telemetría de magnitudes físicas hacia sistemas SCADA.

WWW: La World Wide Web o Red Mundial es el sistema de intercambio de documentos de hipertexto que son accesibles vía Internet.

X25: Protocolo definido por el CCITT en 1974 para el intercambio de datos entre un DTE y un DCE y aplicable en redes basadas en tecnología de conmutación de paquetes.