Facultad 5

Centro de Informática Industrial

TÍTULO: COMPONENTES GRÁFICOS PARA LA REPRESENTACIÓN DE VÁLVULAS Y BOMBAS EN EL SCADA SAINUX.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Claudia María González Fernández.

Tutor: Ing. Luis Andrés Valido Fajardo.

MsC. Dayany Díaz Corona.

Curso docente: 2015-2016.

"La inteligencia consiste no solo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica"

Aristóteles

Declaración de Autoría

Declaro ser el autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste se firma la presente a los _____ días del mes de _____ del año _____.

Firma del autor

Claudia María González Fernández

Firma del tutor

Ing. Luis Andrés Valido Fajardo

Firma del tutor

Ms. Dayany Díaz Corona

AGRADECIMIENTOS

A nuestro comandante por brindarnos un futuro lleno de oportunidades para aportar nuestro grano de arena y ser hombres de bien.

A los profesores de toda esta carrera por educarnos y transmitir tanta experiencia.

A mis tutores:

- Dayany por sus consejos.
- Valido por su paciencia, por ser incondicional ante mis dudas.

A mis amigos incondicionales (Evelio, Laura, Diana) por ser fieles a nuestra amistad.

A mis amigos especiales (Roberto, Kiki, Oscar y Erduin).

A los que siempre han estado y a los que se han incorporado para ser parte de mis recuerdos.

Agradezco a Yirka (la solución de mis problemas), por ser mi gemela, mi otra hermana.

A la profesora Zaida por su paciencia, para asumir mis problemas como los de ella.

A la familia que no escogemos pero la vida te la incorpora (Carmen, Adelaida, Taimara, Lili, Nelsa, Yoyi), gracias a ellas el peso de esta universidad se sentía liviano.

A mi hermana y mi sobrina, porque no importa el espacio ni el tiempo, el esfuerzo siempre es grande y los resultados son bien merecidos.

A mis padres:

- Mi mamá por ser mi fortaleza, por hacer conmigo esta carrera que tantas lágrimas y alegrías ha dado. Por ser fuerte ante las caídas, sacudir el polvo y seguir intentándolo. Y hoy nos estamos graduando.
- Mi papá por ser el hombre más honesto, desinteresado y noble. Por ser guía de mis pensamientos, por ser mi mejor amigo.

- A mis padres porque son la única razón por la que ha valido la pena tanto esfuerzo.

DEDICATORIA

Este trabajo va dedicado a mis abuelos Amador Fernández y Ricardo González, por ser personas muy revolucionarias. Porque la única arma que siempre creyeron invencible era el estudio. Porque por sus cualidades de honestidad, desinterés, se ganaron el respeto de sus familia y amigos. Porque esas cualidades las heredaron mis padres y un día serán ejemplo de mis hijos.

RESUMEN

La Universidad de las Ciencias Informáticas (UCI) posee el Centro de Informática Industrial que tiene como objetivo desarrollar aplicaciones para informatizar y automatizar los procesos industriales. Uno de estos proyectos es el SCADA SAINUX que se basa en la supervisión, control y adquisición de datos de un sistema para proporcionar la información que se genera a los usuarios relacionados con el control y supervisión de dichos procesos.

Dentro de los módulos del SCADA SAINUX se encuentra el módulo Interfaz Hombre Máquina (HMI), que se encarga de representar en un ordenador los procesos que ocurren en el campo, permitiendo al operador estar en contacto directo con el sistema para realizar la supervisión y el control. El HMI provee una biblioteca de componentes gráficos (CG) que permite recrear de la forma más real posible el proceso que se desea automatizar, estos gráficos pueden ser figuras geométricas tan sencillas como líneas, curvas,etc o tan complejas como las que se utilizan en los procesos industriales.

Sin embargo se presentan dificultades para representar los dispositivos de control utilizados en la red de tuberías.

El presente trabajo se basa en el desarrollo de componentes gráficos para la representación de las paletas de componentes válvulas y bombas del SCADA SAINUX con el propósito de solucionar las limitaciones presentes hoy en los Componentes Gráficos del HMI. El desarrollo del mecanismo está sustentado por la metodología AUP UCI con la utilización de tecnologías como SVG y herramientas como el marco de trabajo Qt.

Palabras Clave: HMI, SCADA SAINUX, Componentes Gráficos.

CONTENIDO

INTRODUCCIÓN

El mundo de la ciencia y la tecnología juega un papel importante dentro de la sociedad, las Tecnologías de la Información y las Comunicaciones (TIC) han permitido que el ser humano transforme su forma de comunicarse a través de dispositivos más rápidos y fáciles de utilizar.

Hoy las grandes empresas compiten por desarrollar productos automatizados para facilitar el manejo de grandes monopolios a través de la computadora. Cuba no se queda atrás, la Universidad de las Ciencias Informáticas (UCI) es una universidad innovadora de excelencia, académica y productiva que forma año tras año profesionales integrales y comprometidos con la Revolución, siendo esta la principal fuerza para la informatización de la sociedad y la competitividad a nivel mundial de la industria cubana.

El Centro de Informática Industrial (CEDIN) de la UCI desarrolla productos que aportan ingresos de divisas al país. Tiene la misión de desarrollar productos y servicios informáticos de automatización industrial con un alto valor agregado y que cumplan las necesidades y expectativas de los clientes, potenciando la formación especializada y la investigación.

Uno de estos productos es el SCADA SAINUX, con el objetivo de proporcionar la información que se genera en el proceso productivo a diversos usuarios relacionados directamente con el control y supervisión de dichos procesos, así como a otros pertenecientes a niveles superiores dentro y fuera de la empresa como pueden ser control de calidad, supervisión o mantenimiento.

SAINUX (Sistema de Automatización Industrial basado en GNU/LINUX), es un sistema SCADA distribuido, orientado a componentes y en proceso de desarrollo por el Centro de Informática Industrial (CEDIN) de la Universidad de las Ciencias Informáticas (UCI). Para los procesos de desarrollo y prueba del módulo HMI perteneciente al SAINUX se hace indispensable que el desarrollador o probador según sean el caso tenga en su estación de trabajo el resto de los módulos que componen el SCADA SAINUX ejecutándose como procesos del sistema operativo para poder desarrollar su labor.

Un sistema de supervisión, control y adquisición de datos SCADA se refiere a un sistema central que monitorea y controla un sitio completo o un sistema que se extiende sobre una gran distancia (kilómetros / millas). La instalación de un sistema SCADA necesita un hardware de señal de entrada y salida, sensores y actuadores, controladores, HMI, redes, comunicaciones, base de datos entre otros.

El módulo de HMI en el SCADA SAINUX se encarga de representar, en un ordenador, los procesos que ocurren en el campo, muestra los componentes implicados, los sensores, las estaciones remotas, y el sistema de comunicación dándole al operador total control. Este módulo es el que permite al operador estar en contacto directo con el sistema, realizar la supervisión y el control del proceso en general.

El HMI constan de dos entornos: El entorno de configuración (EC), donde los mantenedores configuran la información específica del área que se desea supervisar y diseñan los despliegues, los cuales haciendo uso de los componentes gráficos permiten simular los procesos de campo; el entorno de visualización (EV) es donde el operador puede supervisar y controlar la configuración realizada en el EC, interactuando con los componentes gráficos para emitir control sobre el sistema, monitorear los cambios de estado de las variables, gestionar alarmas, generar reportes.

Tanto para el EC como para el EV el HMI provee una biblioteca de componentes gráficos (CG) que permiten recrear de una manera lo más real posible los procesos de campo. Estos gráficos pueden ser simples figuras geométricas como: línea, rectángulo, elipse, texto, polígono, polilínea, curvas; pero también muy complejas como los componentes de hardware utilizados en la industria que se desean supervisar. Dentro de estos componentes se encuentran los destinados al diseño y configuración de redes de tuberías.

Dentro de los componentes o dispositivos que componen una red de tuberías se encuentran aquellos que nos permiten controlar el fluido dentro de las tuberías como son las válvulas y bombas.

La biblioteca de componentes gráficos presenta deficiencias para proporcionarles a los mantenedores o encargados de realizar las configuraciones de los componentes gráficos,

que representen dichos elementos de una red de tuberías. Todo lo anterior expuesto trae como consecuencia:

1. La utilización de imágenes para representar dichos dispositivos, trae consigo la pérdida de calidad cuando esta se redimensiona.
2. Rompe con el diseño gráfico homogéneo los componentes usados para el diseño de una red de tuberías por lo que disminuye el grado visual de la representación.
3. La acción de realizar una operación de control sobre los dispositivos reales se torna engorrosa en sus variantes, la primera el mantenedor se ve obligado a conjugar la imagen con algún otro componente de control presente en la biblioteca (CG), mientras en la segunda el operador tendría que buscar el punto que indica el estado del dispositivo (válvula o bomba según sea el caso) en el sumario y desde ahí ejercer la acción deseada.

Para dar solución a esta problemática se propone como **problema de la investigación:** ¿Cómo contribuir a elevar los grados de configuración, visualización y operabilidad de los dispositivos de control utilizados en las redes de tuberías de la Interfaz Hombre Máquina del sistema SCADA SAINUX?

El **objeto de estudio** se centra en: la Interfaz Hombre Máquina del sistema SCADA.

Delimitando como **campo de acción**: Componentes gráficos de las redes de tuberías de la Interfaz Hombre Máquina.

Objetivo general: Desarrollar componentes gráficos que permitan elevar los grados de configuración, visualización y operabilidad de los dispositivos de control utilizados en las redes de tuberías de la Interfaz Hombre Máquina de sistemas SCADA SAINUX.

Posibles resultados:

1. Componentes gráficos para la representación de válvulas utilizadas en las redes de tuberías agrupadas en una paleta especializada.
2. Componentes gráficos para la representación de bombas utilizadas en las redes de tuberías agrupadas en una paleta especializada.

Tareas a cumplir por el estudiante:

1) Elaboración del marco teórico de la investigación a través del estudio del estado del arte que existe actualmente sobre el tema.
2) Identificación de las principales válvulas y bombas que componen las redes de tubería y su representación en sistemas SCADA.
3) Caracterización de las principales válvulas y bombas que componen las redes de tubería y su representación en sistemas SCADA.
4) Realización del levantamiento de requisitos funcionales y no funcionales.
5) Diseñar los componentes gráficos que brinden la solución al problema planteado.
6) Implementación de componentes gráficos brinde la solución al problema planteado.
7) Realización de pruebas para validar el cumplimiento de los requerimientos.

Idea a defender

Con el empleo de los componentes gráficos desarrollados en la investigación se contribuirá a elevar los grados de configuración, visualización y operabilidad de los dispositivos de control utilizados en las redes de tuberías de la Interfaz Hombre Máquina del Sistema SCADA SAINUX.

En el desarrollo del presente trabajo se utilizan los siguientes métodos de investigación:

Métodos Teóricos

- **Analítico-sintético:** Para el estudio de los conceptos empleados en los sistemas SCADA, analizando todos los documentos elaborados, para la extracción de los elementos más importantes.
- **Histórico-lógico:** Para la comprensión de los antecedentes y las tendencias actuales referidas la evolución en el mundo de los sistemas SCADA web.
- **Modelación:** Para la elaboración de múltiples diagramas para un mejor entendimiento del problema y solución.

Métodos Empíricos

- **Observación:** Se puso en práctica este método para conocer el funcionamiento existente en los despliegues del SCADA SAINUX mediante el comportamiento de los dispositivos de campo en las propiedades de los objetos gráficos y sumarios empleados para la toma de decisiones de los operadores.

- **Consulta bibliográfica:** Empleada para consultar las fuentes de información relacionados con los tipos de los sumarios y objetos gráficos visualizados en los entornos de escritorio de los HMI en sistemas SCADAS.

El presente documento está estructurado en tres capítulos:

- **Capítulo 1.Fundamentación teórica:** se realiza un esbozo teórico centrado en los conceptos y características fundamentales de los componentes gráficos para la representación de redes de tuberías.

- **Capítulo 2.Construcción de la solución:** se analiza e implementa la arquitectura propuesta y se explican las principales funcionalidades.

- **Capítulo 3.Implementación y prueba:** se dedica a las pruebas realizadas para garantizar la robustez de la aplicación desarrollada.

CAPÍTULO I

FUNDAMENTACIÓN TEÓRICA

1.1 INTRODUCCIÓN

En este capítulo se abordará el estado del arte y los principales conceptos asociados a un sistema SCADA, así como de los demás elementos involucrados en la investigación, tales como la representación de los componentes gráficos y sus componentes. Se analizarán las principales tecnologías y herramientas para dar solución al problema planteado.

1.2 CONCEPTOS ASOCIADOS

1.2.1 Variable Analógica: Una variable analógica es aquella que toma infinitos valores entre dos puntos cualesquiera de la misma. Un ejemplo de variable analógica es la temperatura. Si se analiza su variación durante un día, se observa que no puede pasar de un valor a otro dando un salto. Esto quiere decir que si la temperatura se incrementa de 11 grados a 17 grados, toma infinitos valores intermedios (1).

1.2.2 Variable Digital: Una variable digital es aquella que solamente puede tomar un determinado número de valores. Un ejemplo de variable digital puede ser una alarma lanzada en un sistema informático para indicar alguna anomalía. Dentro de los tipos de variables digitales se encuentran las variables binarias las cuales toman solamente un estado de dos posibles (1).

1.2.3 Señal Analógica: Una señal analógica es un voltaje o corriente que varía suave y continuamente. Una onda senoidal es una señal analógica de una sola frecuencia. Los voltajes de la voz y del video son señales analógicas que varían de acuerdo con el sonido o variaciones de la luz que corresponden a la información que se está transmitiendo (1).

1.2.4 Señal Digital: Las señales digitales, en contraste con las señales analógicas, no varían en forma continua, sino que cambian en pasos o en incrementos discretos. La mayoría de las señales digitales utilizan códigos binarios o de dos estados (1).

1.2.5 Componente Gráfico u Objeto Gráfico: Representación gráfica de cada dato, en correspondencia con sus valores.

1.2.6 Despliegue: Es un resumen esquematizado con la ventaja de permitir visualizar la estructura y organización de los elementos que componen el proceso ,así como el comportamiento de los mismos, los cuales pueden estar ubicados en un área geográficamente extensa para que el operador monitoree constantemente cada de uno de los componentes que conforman el proceso (2).

1.3 GENERALIDADES DE UN SISTEMA SCADA

El SCADA es un sistema de tiempo real, distribuido en módulos que trabajan de manera conjunta posibilitando el funcionamiento del sistema como un todo. Estos módulos se encuentran interconectados a través de un software para la distribución de los servicios en la red, conocido como software de comunicación entre aplicaciones. La distribución de los módulos existentes en el SCADA permite obtener configuraciones escalables en dependencia de los requisitos presentes en cada escenario. Los módulos se componen de (3):

- **HMI (Human Machine Interface):** El término Interfaz Hombre-Máquina es usado frecuentemente en el contexto de los sistemas de computación y los sistemas electrónicos. Proporciona al operador las funciones de control y supervisión de la planta, así como una mejor comprensión del estado del proceso al representarlo mediante gráficos sinópticos. Muestra lo que está ocurriendo en el campo en tiempo real, brinda interfaces para el envío de los comandos a las estaciones remotas, recibe los valores de las variables que se procesan y permite el manejo de alarmas y eventos.
- **Comunicación:** La capa de comunicación es la capa de software, que se encarga de la comunicación entre los diferentes procesos distribuidos del SCADA. Ofrece un conjunto de servicios a los módulos del SCADA para el envío asíncrono de información. Para ello hace empleo del paradigma publicación/suscripción haciendo

posible desplegar el SCADA de forma distribuida. Es el que proporciona interfaces a los demás para el intercambio de puntos, alarmas, eventos y comandos.

- **Base de Datos Históricos:** Es el módulo que implementa el mecanismo encargado del almacenamiento de la información recibida desde el campo, así como la sucesión de alarmas y eventos generados. La información almacenada es utilizada por varias aplicaciones del sistema para la generación de reportes, tendencias y la gestión de la producción.
- **Recolección:** Tiene como función la adquisición de datos de los dispositivos de campo. Es el encargado de la gestión de los drivers y de la planificación de la encuesta a los dispositivos. El recolector es responsable de construir las listas de los puntos asociados a los dispositivos de campo a consultar, agrupados según la frecuencia de recolección y entregar dichas listas a los manejadores para que estos construyan los bloques de encuesta.
- **Adquisición:** Es el “centro neurológico” del sistema, en él se integran la recolección y el procesamiento de datos. Como recolector es el encargado de la gestión de los drivers y de la planificación de la encuesta a los dispositivos. Durante el procesamiento analiza las alarmas configuradas, publicar los resultados a los demás módulos, mantiene la sincronización con el módulo de adquisición de respaldo y salva la data adquirida para su posterior uso por otras aplicaciones.

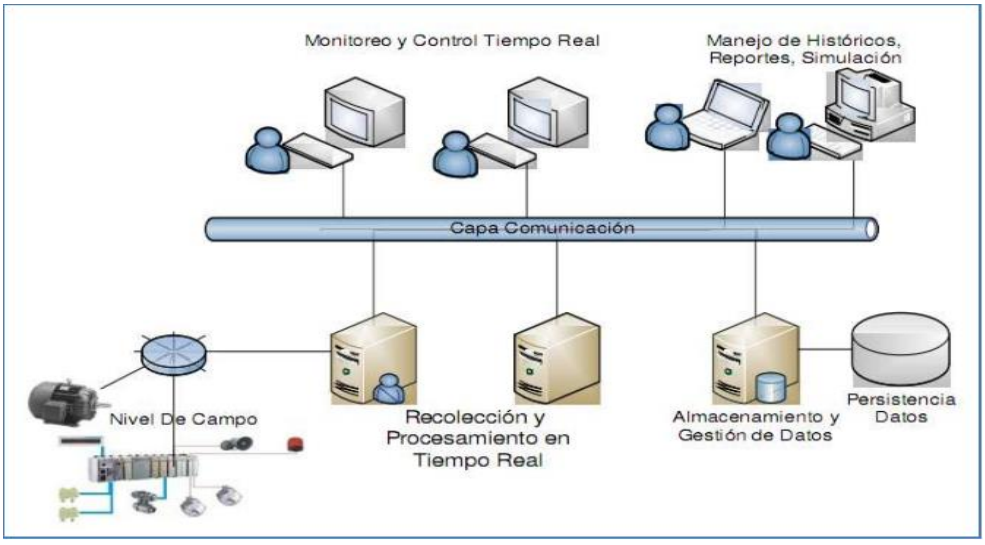


Fig.1 Sistema SCADA

1.3.1 MÓDULO INTERFAZ HOMBRE-MÁQUINA.

Los sistemas HMI podemos pensarlos como una “ventana” de un proceso. Esta ventana puede estar en dispositivos especiales como paneles de operador o una computadora. Estos sistemas, en computadoras se les conoce también como software de interfaz hombre máquina o de monitoreo y control de supervisión. Las señales de los procesos son conducidas al mismo por medio de dispositivos como tarjetas de entrada/salida en la computadora, PLC’s (Controladores lógicos programables), RTU (Unidades remotas de I/O) o DRIVE’s (Variadores de velocidad de motores). Todos estos dispositivos deben tener una comunicación que entienda el HMI (3).

Los sistemas HMI facilitan mucho las tareas de diseño debido a que incorporan protocolos para comunicarse con los dispositivos de campo más conocidos, tienen herramientas para crear bases de datos dinámicas, permiten crear y animar pantallas de forma sencilla y además incluyen gran cantidad de librerías de objetos para representar los diferentes dispositivos de la industria como motores, tanques, indicadores, interruptores válvulas y bombas.

En el SCADA SAINUX, el módulo HMI presenta dos ambientes, el ambiente de edición que permite configurar los procesos, definir y gestionar las variables, editar los controladores, los comandos, las alarmas y opciones adicionales. El editor es el que permite a un operador definir el ambiente de trabajo del SCADA, adaptándolo mejor a la aplicación particular que se desea desarrollar. Por otra parte, la edición gráfica proporciona al operador las funciones de control y supervisión de la planta. El proceso se representa mediante gráficos representativos, almacenados en el ordenador o bien importados desde otras aplicaciones durante la configuración del paquete de software; y el ambiente de ejecución que es el encargado de posibilitar al HMI el monitoreo y supervisión de los procesos del sistema previamente configurados en el ambiente de edición. Se refiere a las características o componentes que estarán disponibles al usuario cuando éste inicie al sistema en el ambiente de ejecución.

1.3.2 FUNCIONES DE UN SISTEMA SCADA.

- **Monitoreo:** Es la habilidad de obtener y mostrar datos en tiempo real. Estos datos se pueden mostrar como números, texto o gráficos que permitan una lectura más fácil de interpretar (4).
- **Supervisión:** Esta función permite junto con el monitoreo la posibilidad de ajustar las condiciones de trabajo del proceso directamente desde la computadora. Alarmas: Es la capacidad de reconocer eventos excepcionales dentro del proceso y reportar estos eventos. Las alarmas son reportadas basadas en límites de control preestablecidos (4).
- **Control:** Es la capacidad de aplicar algoritmos que ajustan los valores del proceso y así mantener estos valores dentro de ciertos límites. Este va más allá del control de supervisión, removiendo la necesidad de la interacción humana. Sin embargo, la aplicación de esta función desde un software corriendo en una PC puede quedar limitada por la confiabilidad que quiera obtenerse del sistema (4).
- **Históricos:** Es la capacidad de muestrear y almacenar en archivos, datos del proceso a una determinada frecuencia. Este almacenamiento de datos es una poderosa herramienta para la optimización y corrección de procesos (4).

1.4 REDES DE TUBERÍAS

El hombre ha ido adquiriendo y mejorando el legado de sus antecesores, perfeccionando sus técnicas, y acrecentando así cada vez más su demanda por conseguir una mejor calidad de vida. Fue así, como surgieron los tubos, quienes, organizados en sistemas, perduran en el tiempo como el medio de transporte de fluidos.

Los sistemas de tuberías están formados por tramos de tuberías y aditamentos que se alimentan fluido arriba por un depósito o una bomba y descargan fluido abajo libremente a la atmósfera o a otro depósito (5).

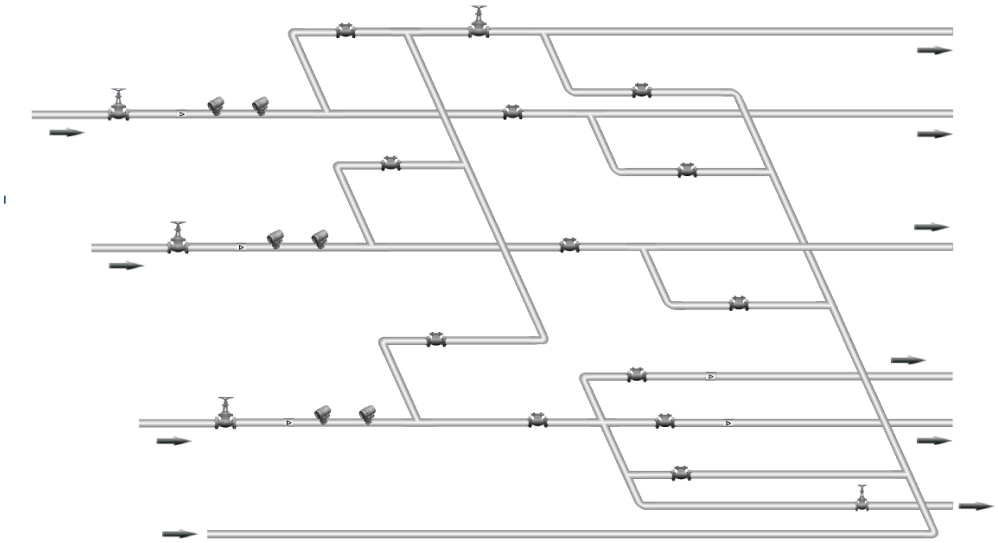


Fig.2 Representación de una red de tuberías

En el dibujo de tuberías se pueden emplear dos sistemas de representación:

- Trazo a línea doble o real (llamados bilineal)
- y Trazo de línea simple o simplificada (monolineal).

La diferencia entre ambas formas de representación es que la representación bilineal se emplean principalmente para tubos con longitudes muy grandes donde no se cortan, para especificar los detalles de los componentes de una red de tuberías esta sería la forma más conveniente, sin embargo la forma monolineal en dibujos a escalas pequeñas, como son, los planos arquitectónicos o los croquis. Cuando los detalles no son relevantes, se suelen simplificar los planos con símbolos a trazo simple, pero que representan de igual forma los componentes y accesorios (5).

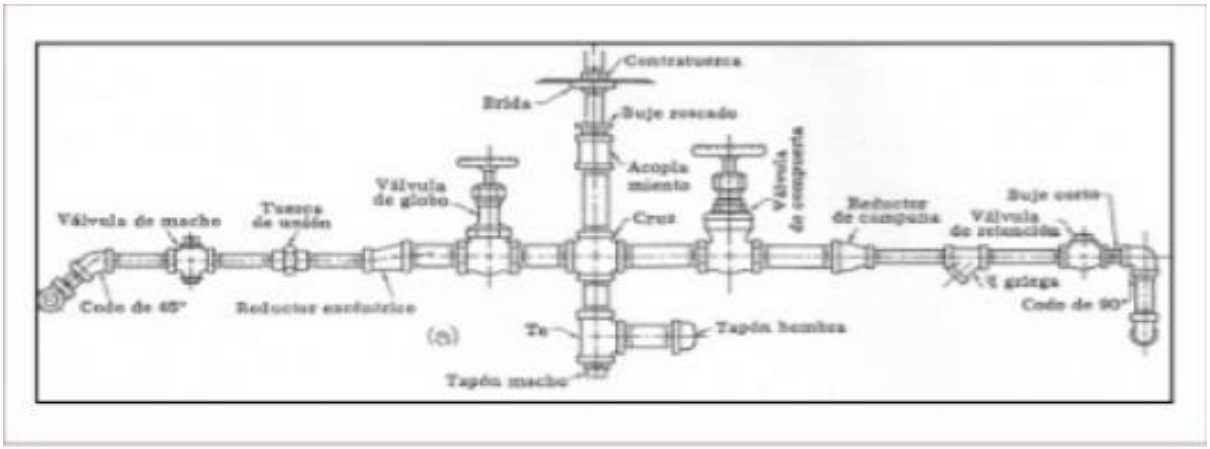


Fig.3 Trazo a línea doble o real

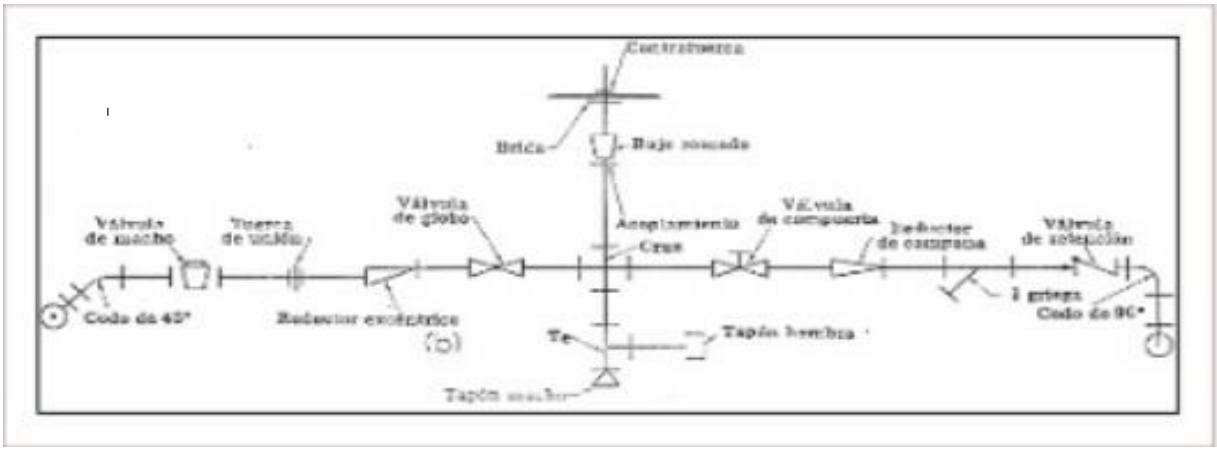
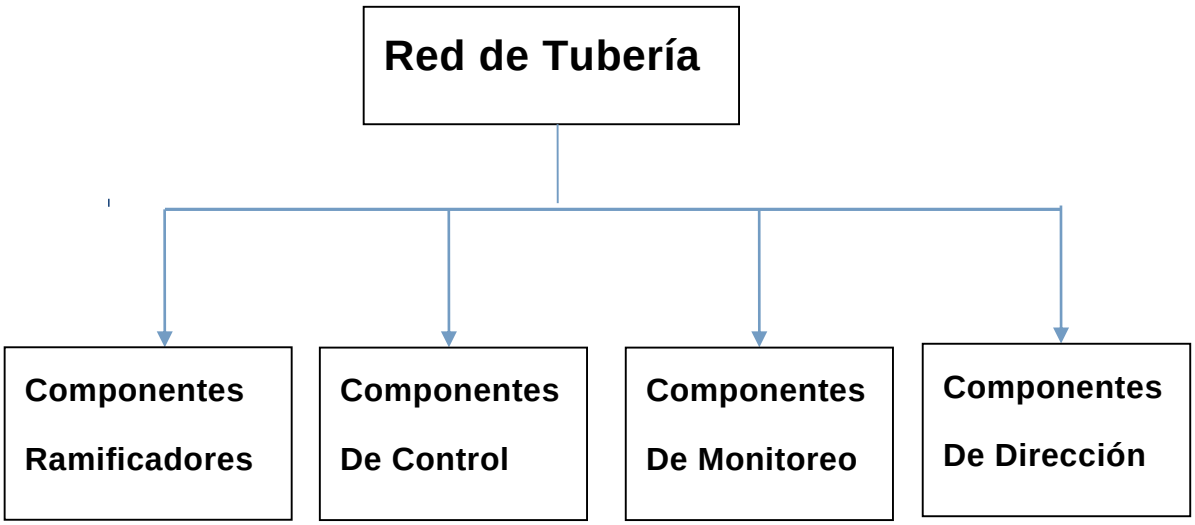


Fig.4 Trazo de línea simple o simplificada

1.4.1 COMPONENTES DE UNA RED DE TUBERÍAS

Una red de tuberías es aquella en la cual los conductos o tuberías que la componen se ramifican sucesivamente, conformando circuitos o anillos cerrados. Un circuito es cualquier trayectoria cerrada que puede recorrer una partícula fluida, partiendo desde un punto o nudo de la red, fluyendo por distintos tramos, hasta llegar al punto de partida (5).

Las redes de tuberías (fluidos) y conductos (aire) se diseñaban (y diseñan todavía) tradicionalmente para su operación a carga total (o caso más desfavorable). Pero con el advenimiento de bombas con variador de frecuencia y los ahorros energéticos que ello conlleva incluso en el tamaño de bombas y ventiladores, se requiere ahora un buen control para asegurar que todo funciona de manera estable cuando la instalación no trabaja a plena carga.



Componentes Ramificadores: se encarga de dividir o concentrar el flujo dentro de red**.**

Componentes de Control: permite controlar y regular el fluido que circula por la red.

Componentes de Monitoreo: son los componentes que se añaden dentro de una red de tuberías con el propósito de monitorear el estado del flujo que circula por la red, ejemplo: los flujómetro.

Componentes de Dirección: estos componentes tienen la responsabilidad de dirigir el fluido de acuerdo al sentido que está diseñado.

1.4.2 VÁLVULAS

El buen funcionamiento de un sistema de tuberías depende en gran parte de la elección adecuada y de la situación de las válvulas que controlan y regulan la circulación de los fluidos en la instalación. Las válvulas deben colocarse en lugares donde sea fácil su manejo y de modo que pueda atenderse la conservación de la instalación sin que interrumpa el funcionamiento de otros aparatos conectados. Una buena válvula debe diseñarse de manera que sus deformaciones debidas a las variaciones de temperatura y de presión, y las dilataciones de las tuberías conectadas, no deformen el asiento; su vástago y el collarín del sellado por comprensión deben permitir poner con facilidad y con rapidez la empaquetadura, y los discos y los asientos deben estar diseñados y hechos con materiales

que permitan que la válvula siga cerrando bien durante un periodo razonable de servicio activo.

Una válvula se puede definir como un aparato mecánico con el cual se puede iniciar, detener o regular la circulación (paso) de líquidos o gases mediante una pieza movible que abre, cierra u obstruye en forma parcial uno o más orificios o conductos.

Las válvulas son unos de los instrumentos de control esenciales en la industria. Debido a su diseño y materiales, las válvulas pueden abrir y cerrar, conectar y desconectar, regular, modular o aislar una enorme serie de líquidos y gases, desde los más simples hasta los más corrosivos o tóxicos.

Las válvulas son, después de las bombas y los motores, componentes importantes de los circuitos hidráulicos. Operaciones de control múltiples, complejas y automáticas se consiguen incorporando al circuito las válvulas más adecuadas. Pueden servir para realizar tres funciones distintas:

- Controlar la presión: limitan la presión del circuito para protegerlo o para reducir la fuerza o el par ejercido por el cilindro o un motor rotativo; limitan la presión en una rama de un circuito a un valor inferior a la presión de trabajo del circuito principal; controlan la sucesión de operaciones entre dos ramas de un circuito.
- Controlar el caudal: controlan, por ejemplo, la velocidad con que se mueve un cilindro hidráulico.
- Controlar la dirección: Bloquean el paso del fluido en un sentido, pero no en el sentido contrario.

1.4.3 PROCESOS AUTOMATIZADOS

La automatización es un sistema donde se trasfieren tareas de producción, realizadas habitualmente por operadores humanos a un conjunto de elementos tecnológicos.
Un sistema automatizado consta de dos partes principales:

- Parte de Mando
- Parte Operativa

La Parte Operativa es la parte que actúa directamente sobre la máquina. Son los elementos que hacen que la máquina se mueva y realice la operación deseada. Los elementos que forman la parte operativa son los accionadores de las máquinas, como motores, cilindros, compresores y los captadores.

La Parte de Mando suele ser un autómata programable (tecnología programada), aunque hasta hace bien poco se utilizaban relés electromagnéticos, tarjetas electrónicas o módulos lógicos neumáticos (tecnología cableada). En un sistema de fabricación automatizado el autómata programable está en el centro del sistema. Este debe ser capaz de comunicarse con todos los constituyentes de sistema automatizado (6).

1.4.4 VÁLVULAS AUTOMATIZADAS

Este tipo de válvulas presentan la característica de accionamiento asistido, siendo desplazado el obturador neumáticamente o con la ayuda de un servomotor. El tipo más extendido es el de accionamiento neumático, recurriendo al uso de servomotores en aplicaciones donde es necesario aplicar grandes esfuerzos, tal y como puede suceder en oleoductos (7).

Una vez que se automatiza una válvula se obtiene a través de la señal digital o analógica si está fluyendo fluido, así como su estado. Otro de los objetivos que se persigue con la automatización es la posibilidad de ejercer control cobre el estado de este a distancia ya sea con el uso de señales analógicas o digitales.

Fig.5 Válvulas

1.4.5 BOMBAS AUTOMATIZADAS

Una bomba hidráulica es una máquina generadora que transforma la energía (generalmente energía mecánica) con la que es transformada en energía del fluido incompresible que mueve. El fluido incompresible puede ser líquido o una mezcla de líquidos y sólidos como puede ser el hormigón antes de fraguar o la pasta de papel. Al incrementar la energía del fluido, se aumenta su presión, su velocidad o su altura. En general, una bomba se utiliza para incrementar la presión de un líquido añadiendo energía al sistema hidráulico, para mover el fluido de una zona de menor presión o altitud a otra de mayor presión o altitud (8). En diversos procesos químicos donde intervienen fluidos es necesaria la implementación de bombas industriales. El objetivo y funcionamiento principal de una bomba industrial es convertir energía mecánica en energía cinética por medio de la producción de presión y velocidad del fluido. El sistema de bombeo tiene diferentes resultados dependiendo de numerosos factores, por eso es importante saber escoger una bomba industrial de acuerdo a las aplicaciones. Entre los factores a considerar se encuentran: presión última, velocidad de bombeo, gases a bombear y presión de proceso.

Fig.6 Bombas

1.5 HERRAMIENTAS Y TECNOLOGÍAS

En este epígrafe se realiza un análisis de las herramientas y tecnologías actuales en el campo del desarrollo de aplicaciones. Este proyecto se desarrolla a partir de las herramientas que brinda el software libre, buscando la independencia tecnológica que estas brindan al posibilitar la libertad de uso y distribución de los programas sin incurrir en litigios de licenciamiento o asuntos legales.

1.5.1 SVG

Vector Grafico Escalable (Scalable Vector Graphics) es un formato vectorial poco conocido pero muy útil para su uso online por su flexibilidad y por la capacidad de ofrecer gráficos con calidad. SVG es vectorial, lo que supone tener todas las ventajas de cualquier formato vectorial. Es escalable, ocupa poco espacio en memoria y permite una definición mayor a tamaños reducidos, mucho mayor que los archivos bitmap. Esta herramienta se utilizó para el diseño gráfico de los componentes. Se utiliza la versión 1.1 (9).

1.5.2 INKSCAPE

Es un editor gráfico de vectores, gratuito y de código libre, puede crear y editar gráficos vectoriales como ilustraciones, diagramas, líneas, gráficos, logotipos, e ilustraciones complejas. El formato principal que utiliza el programa es Scalable Vector Graphics (SVG) versión 0.48. Se encuentra desarrollado principalmente para el sistema operativo GNU/Linux, pero es una herramienta multiplataforma y funciona en Windows, Mac OS X, y otros sistemas derivados de Unix. El objetivo principal de Inkscape es crear una herramienta de dibujo potente y cómoda, totalmente compatible con los estándares XML, SVG y CSS. También quieren mantener una próspera comunidad de usuarios y desarrolladores usando un sistema de desarrollo abierto y orientado a las comunidades, y estando seguros de que Inkscape sea fácil de aprender, de usar y de mejorar. Esta herramienta se utilizó como editor gráfico para imágenes y tablas (10).

1.5.3 LENGUAJE C++

Es un lenguaje de programación basado en C, al que se han añadido nuevos tipos de datos, clases, plantillas, mecanismo de excepciones, sistema de espacios de nombres, funciones, sobrecarga de operadores, referencias, operadores para manejo de memoria persistente, y algunas utilidades adicionales de bibliotecas. Nació para añadirle cualidades y características de las que carecía C. El resultado es que, como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción. El lenguaje de programación definido se utilizó para programar la lógica de los componentes (11).

En este trabajo se utilizó debido a sus múltiples ventajas, entre ellas el hecho de que, como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción. Además, la definición "oficial" del lenguaje nos dice que C++ es un lenguaje de propósito general basado en el C,

al que se han añadido nuevos tipos de datos, clases, plantillas, mecanismo de excepciones, sistema de espacios de nombres, funciones online, sobrecarga de operadores, referencias, operadores para manejo de memoria persistente, y algunas utilidades adicionales de librería (11).

1.5.4 ENTORNO DE DESARROLLO INTEGRADO (IDE)

Para el desarrollo de este producto se requiere de un entorno de desarrollo integrado (IDE, por sus siglas en inglés), el cual puede denominarse como un entorno de programación que consiste en un editor de código y un compilador (12).

El IDE seleccionado para el desarrollo de la aplicación es Qt Creator en su versión 2.5.0. Qt Creator es un IDE multiplataforma para el desarrollo de aplicaciones que pueden o no tener interfaz gráfica. Este se centra en proporcionar características que ayudan a los nuevos usuarios del IDE a aprender y comenzar a desarrollar rápidamente.

Existen otras características importantes que presenta este IDE como la disponibilidad de código fuente, la excelente documentación organizada que provee en el QtAssistant y un editor para el diseño de formularios denominado QtDesigner.

Qt Creator cuenta con:

- ✓ Un editor de código con soporte para C++.
- ✓ Herramientas para la rápida navegación por el código.
- ✓ Resaltado de sintaxis y auto-completado de código.
- ✓ Control estático de código y estilo a medida.
- ✓ Soporte para refactorización de código.
- ✓ Paréntesis coincidentes y modos de selección.

1.5.5 BIBLIOTECA QT

Qt es un marco de referencia de desarrollo de software que incluye clases, bibliotecas y herramientas para la producción de aplicaciones usando el lenguaje C++ de forma nativa y puede operar en varias plataformas incluyendo los sistemas Unix (Linux, MacOS X, Solaris) o incluso toda la familia de Windows. Esta poderosa herramienta permite desarrollar ricas aplicaciones gráficas e incluye soporte de nuevas tecnologías como OpenGL, XML, Bases de Datos y programación para redes. Diversos proyectos de renombre mundial usan este marco de referencia entre los que se encuentra, el entorno de escritorio KDE, incluyendo sus aplicaciones, así como el visor de videos VLC (Video LAN). Existen diferentes versiones de Qt que posibilitan su utilización con otros lenguajes de programación como Python (PyQt), Java (Qt Jambi), Perl (PelrQt), entre otros. Este marco de trabajo se detalló para utilizar las clases, bibliotecas y herramientas. (13).

1.5.6 UML

El Lenguaje Unificado de Modelado (UML) versión 2.0 es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software orientado a objetos (OO). Ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. El lenguaje UML se definió para representar gráficamente las clases. (14).

1.5.7 VISUAL PARADIGM

Visual Paradigm para UML versión 8.0 es una herramienta CASE que soporta el modelado mediante UML y proporción asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software. Se utilizó esta herramienta para la representación de diagramas y establecer coherencia entre ellos, además de generar el código partiendo de los diagramas y viceversa, otra de

las ventajas que presenta esta herramienta es la integración con otras aplicaciones y el trabajo multiusuario (15).

La misma fue muy importante en el desarrollo de este trabajo debido a que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas.

En este trabajo el uso de esta herramienta fue de mucha ayuda ya que esta tiene una gran disponibilidad en múltiples plataformas presenta un diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad, usa un lenguaje estándar UML que es común a todo el equipo de desarrollo lo que facilita la comunicación, posee capacidades de ingeniería directa e inversa, además el modelo y el código permanecen sincronizados en todo el ciclo de desarrollo, tiene soporte para varios idiomas, es fácil de instalar y actualizar, se pueden realizar diagramas de procesos de negocio - proceso, decisión, actor de negocio y documentos, tiene una gran interoperabilidad con modelos UML2 (metamodelos UML 2.x para plataforma Eclipse) a través de XMI, tiene gran integración con las bases de datos, se puede transformar de diagramas de Entidad-Relación en tablas de base de datos y de Sistemas Gestores de Bases de Datos existentes a diagramas de Entidad-Relación, además posee un potente generador de informes y un editor de figuras.

1.5.8 METODOLOGÍA DEL DESARROLLO DE SOFTWARE

Las metodologías de desarrollo de software constituyen un conjunto de procedimientos, técnicas y herramientas que ofrecen ayuda y brindan soporte a equipos de desarrollo de software para crear nuevos productos. Estas metodologías siguen los modelos del ciclo de vida del software, indicando cómo hay que obtener los distintos productos parciales y finales a lo largo del desarrollo de un proyecto.

Lograr la construcción de un sistema informático eficiente, que cumpla con los requerimientos planteados, es una tarea realmente intensa y sobre todo difícil de cumplir.

Las metodologías para el desarrollo del software imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Una metodología de desarrollo de software tiene como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. No existe una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Las metodologías de desarrollo se pueden dividir en dos grupos de acuerdo con sus características y los objetivos que persiguen: ágiles y robustas (16).

Roles definidos por AUP-UCI

1. Jefe de proyecto: Maneja a los miembros construye relaciones con los stakeholders, coordina interacciones con los stakeholders, planea, maneja y asigna los recursos.

2. Analista: Desarrolla, adapta y apoya sus materiales del proceso del software.

3. Desarrollador: Escribe, testea y construye software.

4. Arquitecto de software: Diseña, prueba, desarrolla, y apoya los esquemas de la BD.

6. Administrador de la configuración: Un encargado de la configuración es responsable de proporcionar la infraestructura total y el ambiente del CM al equipo de desarrollo.

7. Cliente/Proveedor de requisitos.

8. Administrador de calidad: Responsables del éxito de la prueba, incluyendo el planeamiento, la gerencia, y la defensa para la prueba y las actividades de la calidad.

9. Probador: Encargado de ejecutar las pruebas.

1.5.9 SISTEMA OPERATIVO

Debian GNU/Linux es un sistema operativo de libre distribución y uso. Permite a varios usuarios acceder al mismo tiempo a través de terminales, y distribuye los recursos

disponibles entre todos. Puede correr en la mayoría de plataformas del mercado (procesadores de la gama Intel y AMD, Motorola, Sun, Sparc, etc.). La versión empleada en la solución es 7.0 Wheezy (17).

1.6 CONCLUSIONES PARCIALES

- En este capítulo se realizó un esbozo de las principales características y conceptos importantes de los sistemas SCADA, haciendo énfasis en el módulo HMI.
- Se definieron y seleccionaron las tecnologías necesarias para llevar a cabo el cumplimiento del objetivo planteado en el trabajo.
- A partir de estos puntos se comenzará el desarrollo de la propuesta de solución.

CAPÍTULO II
ANÁLISIS Y DISEÑO DE LA SOLUCIÓN

2.1 INTRODUCCIÓN

Este capítulo tiene como objetivo describir las actividades que se realizaron durante los procesos de análisis, diseño. Se proponen los requisitos funcionales y no funcionales que regirán el desarrollo de la solución propuesta. A partir de los requerimientos se determinan las historias de usuarios que regirán el desarrollo de la solución propuesta y se muestran los diagramas que describen el funcionamiento del sistema, se diseña una arquitectura y se hace una descripción del diseño.

2.1.2 MODELO DE DOMINIO

Un modelo del dominio se utiliza con frecuencia como fuente de inspiración para el diseño de los objetos de software. El modelo del dominio muestra (a los modeladores) clases conceptuales significativas en un dominio de] problema; es el artefacto más importante que se crea durante el análisis orientado a objetos (18).

Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Pueden mostrar:

•Objetos del dominio o clases conceptuales.

•Asociaciones entre las clases conceptuales.

•Atributos de las clases conceptuales.

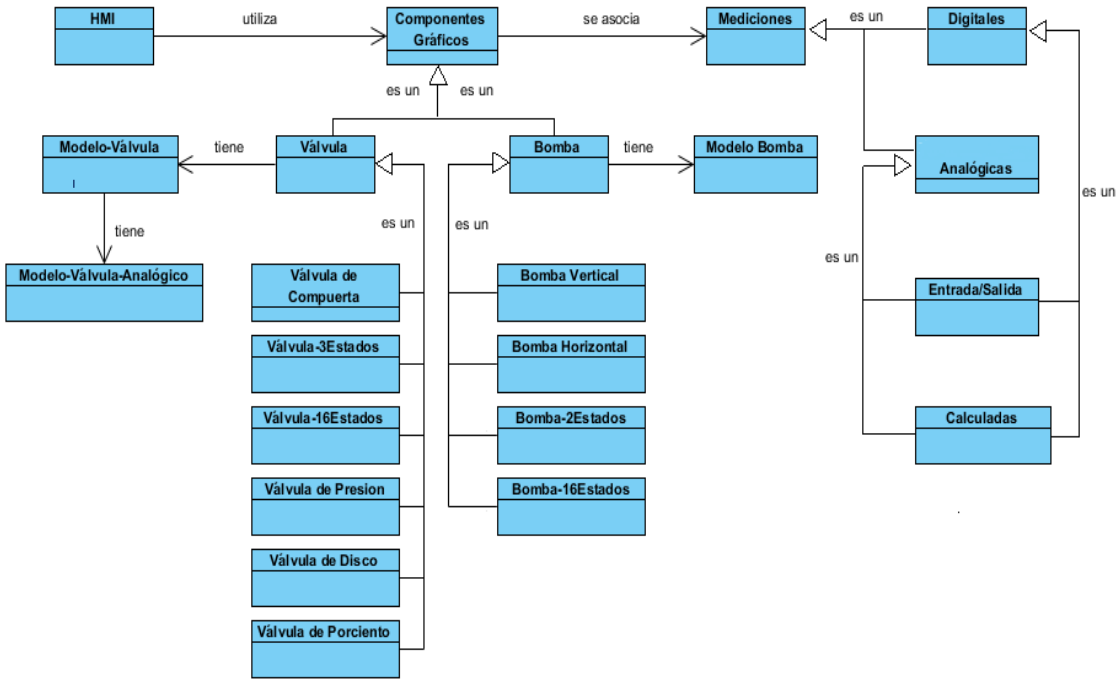


Fig.7 Modelo de Dominio

Para el modelo del negocio se tuvieron en cuenta la descripción de sus entidades:

- **HMI**: es el módulo que se encarga de representar al usuario u operador lo que sucede en el proceso.
- **Componentes Gráficos:** representación abstracta de algún dispositivo del proceso que se está supervisando.
- **Mediciones:** se le asocian a los componentes gráficos para obtener un valor ya sea analógico (entre dos puntos cualesquiera) o digital (determinados valores). Estos valores pueden ser Entrada/salida tomado directamente de un dispositivo de campo o Calculado dependiendo del valor que arroje una función matemática.
- **Válvulas:** Las válvulas varían de acuerdo a sus características. Válvula de tres estados define tres colores para indicar los tres estados que puede tomar según sus mediciones (abierto, semi-abierto y cerrado) y Válvula de dieciséis estados sus colores son para representar el nivel de llenado que presenta. Válvula de presión diseñada para aliviar la presión cuando un fluido supera un límite preestablecido. Válvula de Compuerta realiza solo dos acciones a través del cierre para liberar o no el paso de fluidos. Válvula de porciento indica a través del flujómetro el porciento de

llenado y la Válvula de Disco el nivel de cierre o apertura va a estar dado por el valor de un ángulo (entre 0 y 360 grados).

- **Bombas:** Bomba Horizontal, la disposición del eje de giro horizontal presupone que la bomba y el motor se hallan a la misma altura; éste tipo de bombas se utiliza para funcionamiento en seco, exterior al líquido bombeado que llega a la bomba por medio de una tubería de aspiración. En las bombas verticales, el motor puede estar inmediatamente sobre la bomba, o muy por encima de ésta. La Bomba de tres estados define tres colores para indicar el paso del fluido y la Bomba de dieciséis estados se le asocia una variable digital para indicar el estado en que se encuentra.
- **Modelo-Válvula** y **Modelo-Bomba:** son las clases que guardan los datos que deben persistir de los componentes.

2.1.3 ANÁLISIS DE LA SOLUCIÓN

Dentro de los componentes de control de una red de tuberías se pueden hallar las válvulas y bombas los cuales permiten controlar el flujo de líquidos y gases dentro de una red de tuberías. Para las bombas se controlan mayoritariamente los parámetros que controlan el flujo que circula por estas estructuras y sus estados de funcionamiento, para el caso de las válvulas se controlan el estado de apertura. Este último parámetro en dispositivos es dado bien por una variable digital o analógica de acuerdo a especificaciones propias del componente. La solución debe ser capaz de representar los dispositivos reales, y ser lo más configurable posible, además de representar el estado o valor en que se encuentra las mediciones asociadas a dichos dispositivos.

2.1.4 REQUISITOS

Un requisito es una “condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado”. También se aplica a las condiciones que debe cumplir o poseer un sistema o uno de sus componentes para satisfacer un contrato, una norma o una especificación (19).

Los requisitos cumplen una doble función:

- Son una oferta de contrato -> abiertos a la interpretación.
- Son el contrato en sí mismo -> deben definirse de forma detallada.

Los requerimientos puedes dividirse en requerimientos funcionales y requerimientos no funcionales. Los requerimientos funcionales definen las funciones que el sistema será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Los requerimientos no funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares, etc.

2.1.5 REQUISITOS FUNCIONALES

Los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer (20).

Muchos de los problemas de la ingeniería de software provienen de la imprecisión en la especificación de requerimientos. Para un desarrollador de sistemas es natural dar interpretaciones de un requerimiento ambiguo con el fin de simplificar su implementación. Sin embargo, a menudo no es lo que el cliente desea. Se tienen que estipular nuevos requerimientos y se deben hacer cambios al sistema, retrasando la entrega de éste e incrementando el costo.

En principio, la especificación de requerimientos funcionales de un sistema debe estar completa y ser consistente. La compleción significa que todos los servicios solicitados por el usuario están definidos. La consistencia significa que los requerimientos no tienen definiciones contradictorias.

Para el ambiente de **Edición–Configuración** se determinaron los siguientes requisitos funcionales, comunes para ambos grupos de componentes:

RF1. El sistema debe permitir definir nombre del componente.

RF2. El sistema debe permitir a rotación del componente.

RF.3 El sistema debe permitir configurar la posición del componente.

RF.4 El sistema debe permitir configurar la dimensión del componente.

RF.5 El sistema debe permitir asociar variables al componente.

RF.6 El sistema debe permitir una relación biyectiva entre las mediciones relacionadas con el componente a los parámetros medible de este.

RF.7 El sistema debe permitir configurar el tipo de control del componente.

RF.8 El sistema debe permitir configurar correspondencia entre los valores discretos.

RF 9. El sistema debe permitir configurar la opacidad del objeto del componente.

RF.10 El sistema debe permitir configurar color de la tubería del componente.

RF.11 El sistema debe permitir configurar color de la brida del componente.

RF.12 El sistema debe permitir mostrar u ocultar bridas del componente.

RF.13 El sistema debe permitir configurar el color que representa el estado indefinido del componente.

RF.14 El sistema debe permitir mostrar/ocultar por ciento.

- Para el componente Válvula de Porciento se declara el siguiente requisito:

RF.15 El sistema debe permitir configurar dimensiones del Meter Emergente.

- Para el componente Válvula de Reducción de Presión se declaran los siguientes requisitos:

RF.16 El sistema debe permitir configurar color del anillo.

RF.17 El sistema debe permitir configurar color del plato.

RF.18 El sistema debe permitir configurar color de la aguja.

RF.19 El sistema debe permitir configurar color de la posición del valor.

RF.20 El sistema debe permitir configurar color del valor.

RF.21 El sistema debe permitir configurar tamaño de la fuente del valor.

RF.22 El sistema debe permitir mostrar valor.

RF.23 El sistema debe permitir configurar color de la escala.

RF.24 El sistema debe permitir configurar tamaño de la fuente de la escala.

RF.25 El sistema debe permitir mostrar/ocultar escala numérica.

RF.26 El sistema debe permitir configurar posición de la unidad de medida.

RF.27 El sistema debe permitir configurar tamaño de fuente de la unidad.

RF.28 El sistema debe permitir mostrar unidad de medida.

RF.29 El sistema debe permitir configurar dimensión del flujómetro Emergente.

Para el ambiente de **Visualización - Ejecución** se determinaron los siguientes requisitos funcionales, comunes para ambos grupos de componentes:

RF.30 El sistema debe permitir ejecución de comando sobre las mediciones asociadas al componente.

RF.31 El sistema debe permitir representar en el componente el estado en el que se encuentra las mediciones asociadas.

2.1.6 REQUISITOS NO FUNCIONALES

Son aquellos requerimientos que no se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y la representación de datos que se utiliza en las interfaces del sistema (20).

Los requerimientos no funcionales surgen de la necesidad del usuario, debido a las restricciones en el presupuesto, a las políticas de la organización, a la necesidad de interoperabilidad con otros sistemas de software o hardware o a factores externos como los reglamentos de seguridad, las políticas de privacidad, entre otros.

Software

- El sistema debe funcionar en el Sistema Operativo GNU/Linux en su distribución Debian, específicamente la versión Wheezy.
- Qt en su versión 4.8 o una versión superior.

Hardware

- Memoria RAM: 2 Gigabytes.
- Microprocesador: Intel(R) Core(Tm) 2 Duo.

2.2 HISTORIAS DE USUARIOS

Es la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es

muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (21).

Historia de Usuario	
Número: 1	**Nombre de Historia:** Definir nombre del objeto
Actor: Usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos estimados:** 0.4
Nivel de Complejidad: Alta	**Puntos reales:** 0.4
Descripción: El sistema proporciona una interfaz al cliente donde puede escribir el nombre del objeto, el mismo no debe exceder de 32 caracteres, ni puede contener comillas simples (') y and per se and (&).	

Historia de Usuario # 1

La descripción de las demás historias de usuario se encuentra en el anexo 1.

2.3 PLANIFICACIÓN DEL DESARROLLO

Esta fase tiene como propósito establecer un acuerdo entre clientes y desarrolladores sobre el menor tiempo en que la mayor cantidad de historias de usuarios puedan ser utilizadas.

2.3.1 ESTIMACIÓN DE ESFUERZOS E ITERACIONES POR HISTORIA DE USUARIO.

Historias de Usuario	Puntos de Estimación	Iteración	Duración total de las iteraciones (semanas)
Definir nombre del objeto	0.4 semanas	1	3
Permitir la rotación del widget.	0.3 semanas		
Configurar la posición del objeto.	0.5 semanas		
Configurar la dimensión del objeto.	0.3 semanas		
Asociar variables al componente.	0.6 semanas		
Asociar mediciones relacionadas con el componente a los parámetros medible de este.	0.6 semanas		
Configurar el tipo de control del componente.	0.6 semanas	2	3
Configurar correspondencia entre los valores discretos.	0.6 semana		
Configurar la opacidad del objeto del componente.	0.2 semanas		
Configurar color de la tubería del componente.	0.4 semanas		

Configurar color de la brida del componente.	0.4 semana		
Mostrar u ocultar bridas del componente.	0.4 semanas		
Configurar el color que representa el estado indefinido del componente.	1 semanas	3	3
Mostrar/ocultar por ciento.	0.3 semanas		
Configurar dimensiones del Meter Emergente.	1.1 semanas		
Configurar color del anillo.	0.1 semanas		
Configurar color del plato.	0.5 semanas		
Configurar color de la aguja.	0.5 semana	4	3
Configurar color de la posición del valor.	0.5 semana		
Configurar color del valor.	0.6 semana		
Configurar tamaño de la fuente del valor.	0.4 semanas		
Mostrar valor.	1 semana		
Mostrar/ocultar escala numérica.	1 semana	5	3
Configurar posición de la unidad de medida.	1 semana		
Configurar color de la escala.	0.4 semana		

Configurar tamaño de la fuente de la escala.	0.6 semana		
Mostrar unidad de medida.	0.4 semana	6	3
Configurar tamaño de fuente de la unidad.	0.7 semanas		
Configurar tamaño de fuente de la unidad	0.6 semana		
Permitir ejecución de comando sobre las mediciones asociadas al componente.	1.5 semana	7	3
Representar en el componente el estado en el que se encuentra.	1.5 semanas		

2.3.2 PLAN DE ITERACIONES

Luego de definidas las Historias de Usuarios (HU) y estimado el esfuerzo propuesto para su realización, se realizó el sistema en ocho iteraciones, las cuales se describen detalladamente a continuación:

- **Iteración I:** Iteración que tiene como objetivo realizar las HU 1, 2, 3, 4, 5 y 6 las cuales se encargan de configurar las propiedades del componente seleccionado.
- **Iteración II**: Esta iteración tiene como objetivo realizar las HU 7, 8, 9, 10, 11 y 12 siendo estas las encargadas de añadir animaciones al accesorio de la red de tubería.
- **Iteración III:** Iteración tiene como objetivo realizar las HU 13, 14, 15, 16 y 17 las cuales son las referentes a cambiar el color de los elementos que componen el medidor de flujo.

- **Iteración IV:** La presente iteración tiene como objetivo realizar las HU 18, 19, 20, 21 y 22 siendo estas las encargadas de configurar la posición y el tamaño de los elementos que componen el medidor de flujo.
- **Iteración V y VI:** Son las iteraciones conformadas por las HU 23, 24, 25, 26, 27, 28 y 29 las cuales tienen como objetivo configurar las propiedades de los medidores de flujo digital y analógico.
- **Iteración VII:** Esta iteración está compuesta por las HU 30 y 31, en las cuales se ejecuta una acción para representan a través de un color el estado en que se encuentra.

2.4 DIAGRAMA DE PAQUETES

Un diagrama de paquetes en el Lenguaje Unificado de Modelado representa las dependencias entre los paquetes que componen un modelo. Es decir, muestra cómo un sistema está dividido en agrupaciones lógicas y las dependencias entre esas agrupaciones.

Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema.

Los paquetes están normalmente organizados para maximizar la coherencia interna dentro de cada paquete y minimizar el acoplamiento externo entre los paquetes. Con estas líneas maestras sobre la mesa, los paquetes son buenos elementos de gestión. Cada paquete puede asignarse a un individuo o a un equipo, y las dependencias entre ellos pueden indicar el orden de desarrollo requerido.

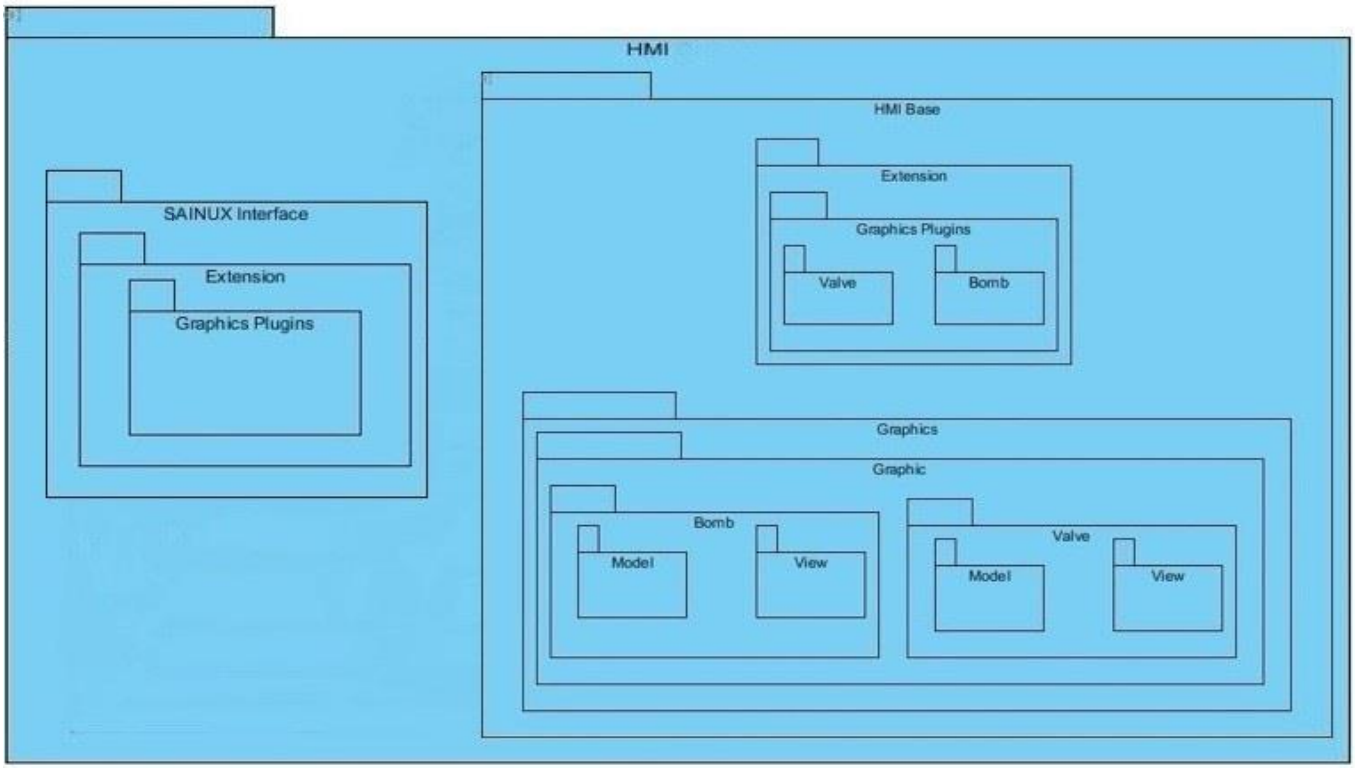


Fig.8 Diagrama de Paquetes

La solución está estructurada en dos partes; Componentes y Paletas de Componentes. La primera parte contiene la carpeta SAINUX Interface, esta contiene la subcarpeta Graphics Plugins que contiene las entidades referentes a las paletas de los componentes, pero de la interfaz SAINUX. En la segunda parte están las Paletas de Componentes divididas en la carpeta HMI donde se almacenan las propiedades comunes de cualquier módulo de un sistema SCADA SAINUX, este módulo contiene los paquetes Extensión y dentro de él la subcarpeta Graphics Plugins donde se colocan las entidades relacionadas con las paletas Valve y Bomb y dentro de ellas según el tipo de componente que representa la paleta. El segundo paquete lo conforma Graphics donde se guardan las entidades de la biblioteca de componentes gráficos, dentro de él la subcarpeta Graphic que agrupa las entidades que representan a los componentes Válvulas y Bombas, con sus entidades para representar la Vista y el Modelo.

2.5 DISEÑO DE CLASES

El diseño es un proceso de resolución de problemas cuyo objetivo es encontrar y describir una forma:

- Para implementar los requisitos funcionales del sistema.
- Respetando las restricciones impuestas por los requisitos no funcionales.
- Ajustándose a los principios generales de calidad.
- El proceso de diseño es, por tanto, un proceso iterativo, mediante el cual se va a realizar una traducción de los requisitos en una representación del software (28).

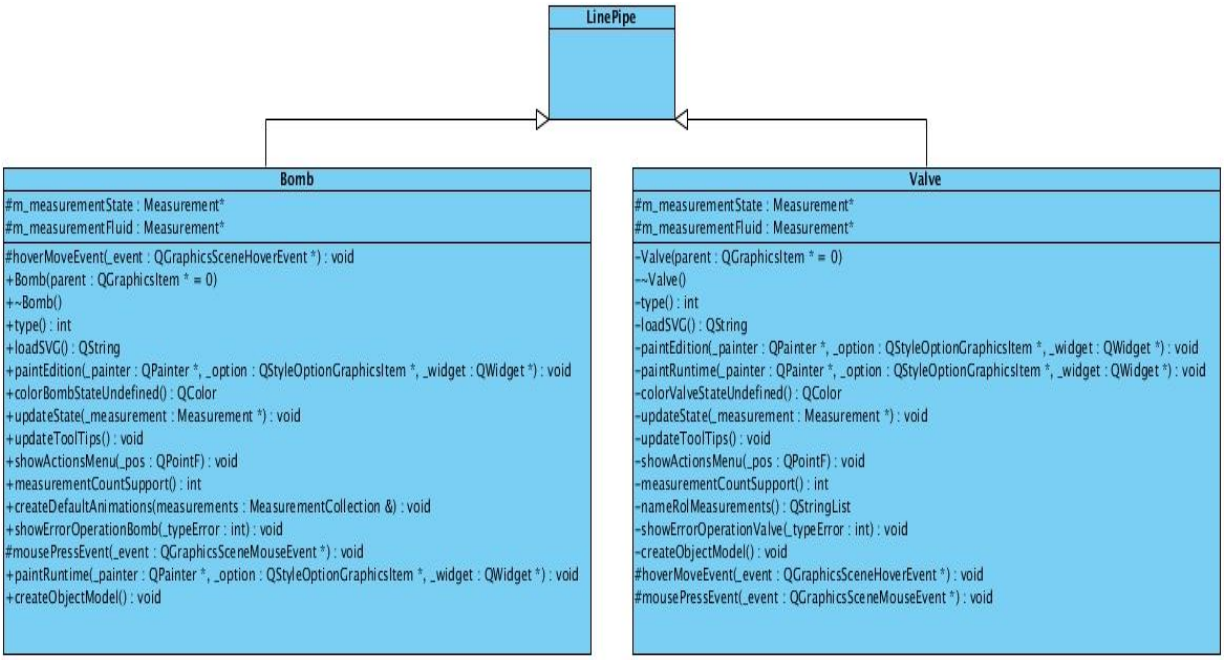


Fig.9 Diseño de clase para las entidades Válvula y Bomba

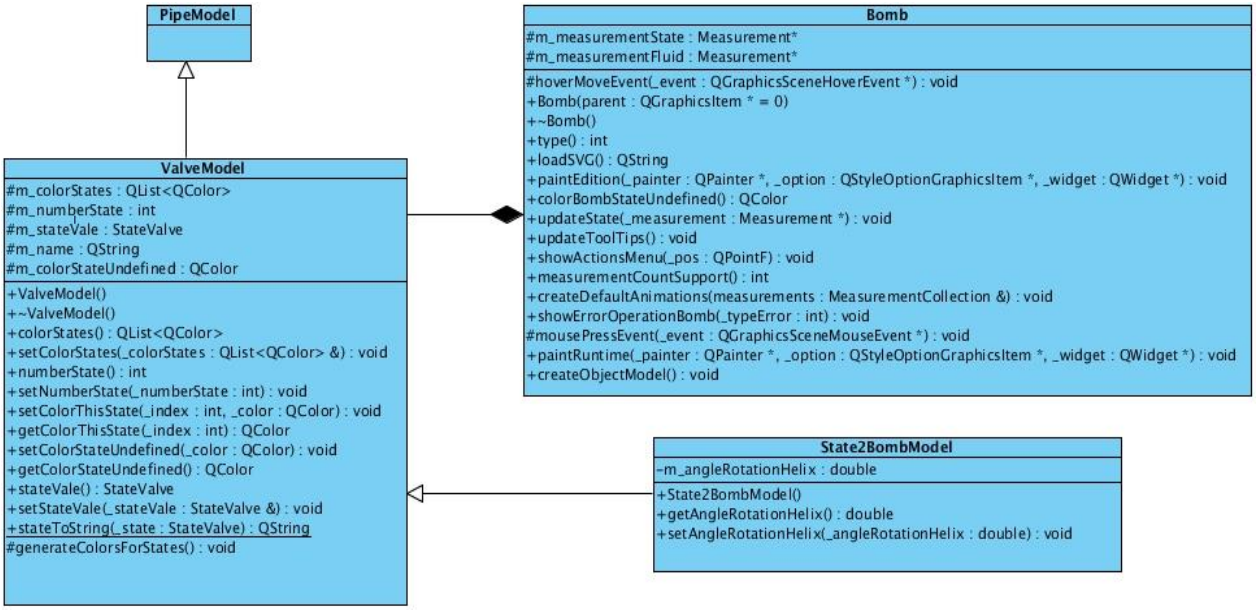


Fig.10 Diseño de clase para la entidad Bomba

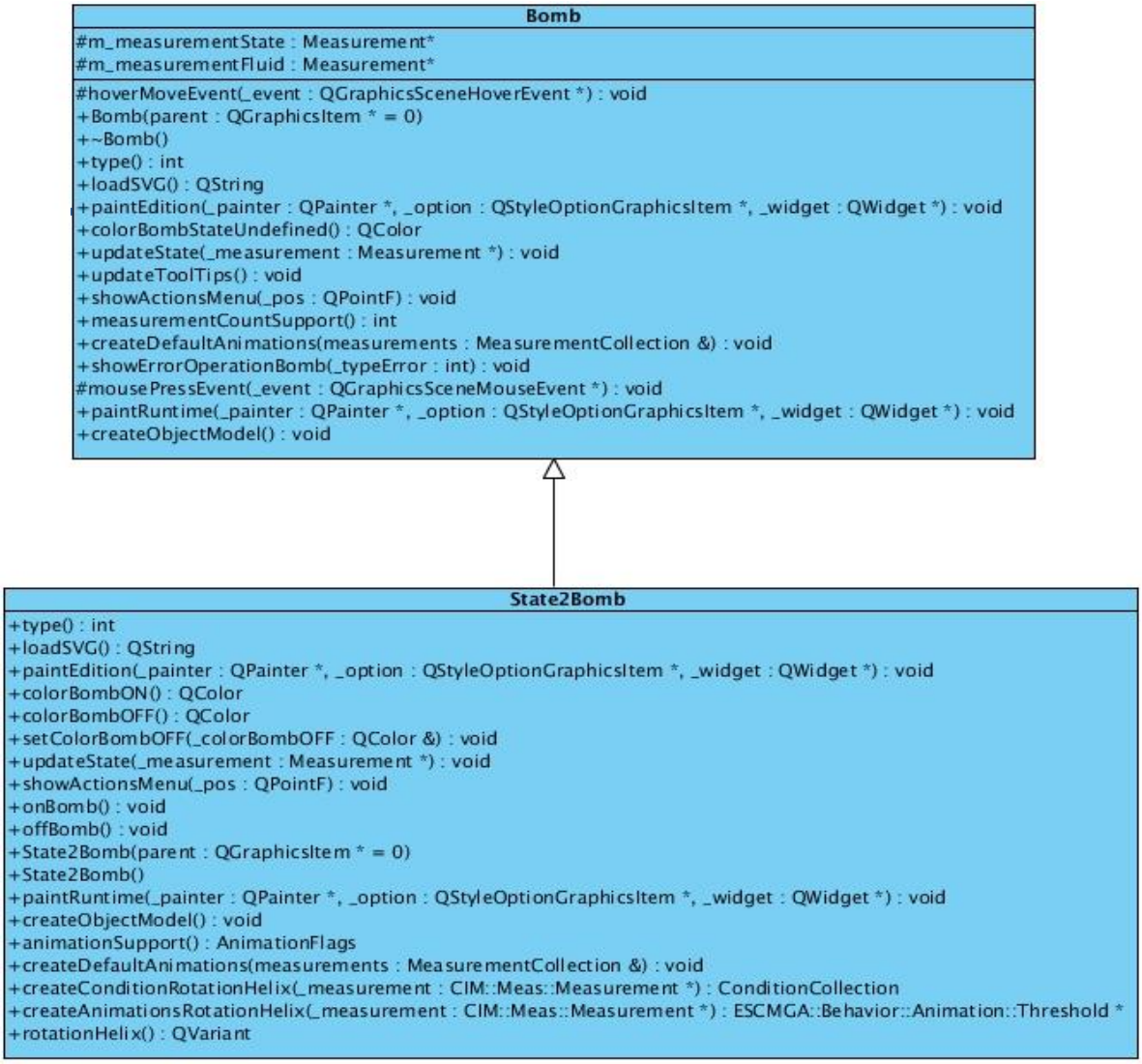


Fig11. Diseño de clases para la entidad Bombas y las clases que heredan

Los restantes diseños de clases se encuentran en el anexo 3.

Una vez que las clases fueron estructuradas, se diseñaron para establecer la relación entre ellas. Partiendo de la clase LinePipe, como clase base que contiene los diseños gráficos de las tuberías y bombas, hereda de ella la clase Bomb y la clase Valve. De la clase Bomb heredan los diferentes tipos de bombas: State2Bomb, State16Bomb, VerticalBomb y HorizontalBomb, y de la clase Valve.

- **Válvulas:** Válvula de tres estados define tres colores para indicar los tres estados que puede tomar según sus mediciones (abierto, semi-abierto y cerrado) y Válvula de dieciséis estados sus colores son para representar el nivel de llenado que presenta. Válvula de presión diseñada para aliviar la presión cuando un fluido supera un límite preestablecido. Válvula de Compuerta realiza solo dos acciones a través del cierre para liberar o no el paso de fluidos. Válvula de porciento indica a través del flujómetro el porciento de llenado y la Válvula de Disco el nivel de cierre o apertura va a estar dado por el valor de un ángulo (entre 0 y 360 grados).

- **Bombas:** Bomba Horizontal, la disposición del eje de giro horizontal presupone que la bomba y el motor se hallan a la misma altura; éste tipo de bombas se utiliza para funcionamiento en seco, exterior al líquido bombeado que llega a la bomba por medio de una tubería de aspiración. En las bombas verticales, el motor puede estar inmediatamente sobre la bomba, o muy por encima de ésta. La Bomba de tres estados define tres colores para indicar el paso del fluido y la Bomba de dieciséis estados se le asocia una variable digital para indicar el estado en que se encuentra.

- **Modelo-Válvula** y **Modelo-Bomba:** son las clases que guardan los datos que deben persistir de los componentes.

Fig.11 Diseño de Clases para los Componentes Gráficos Bombas

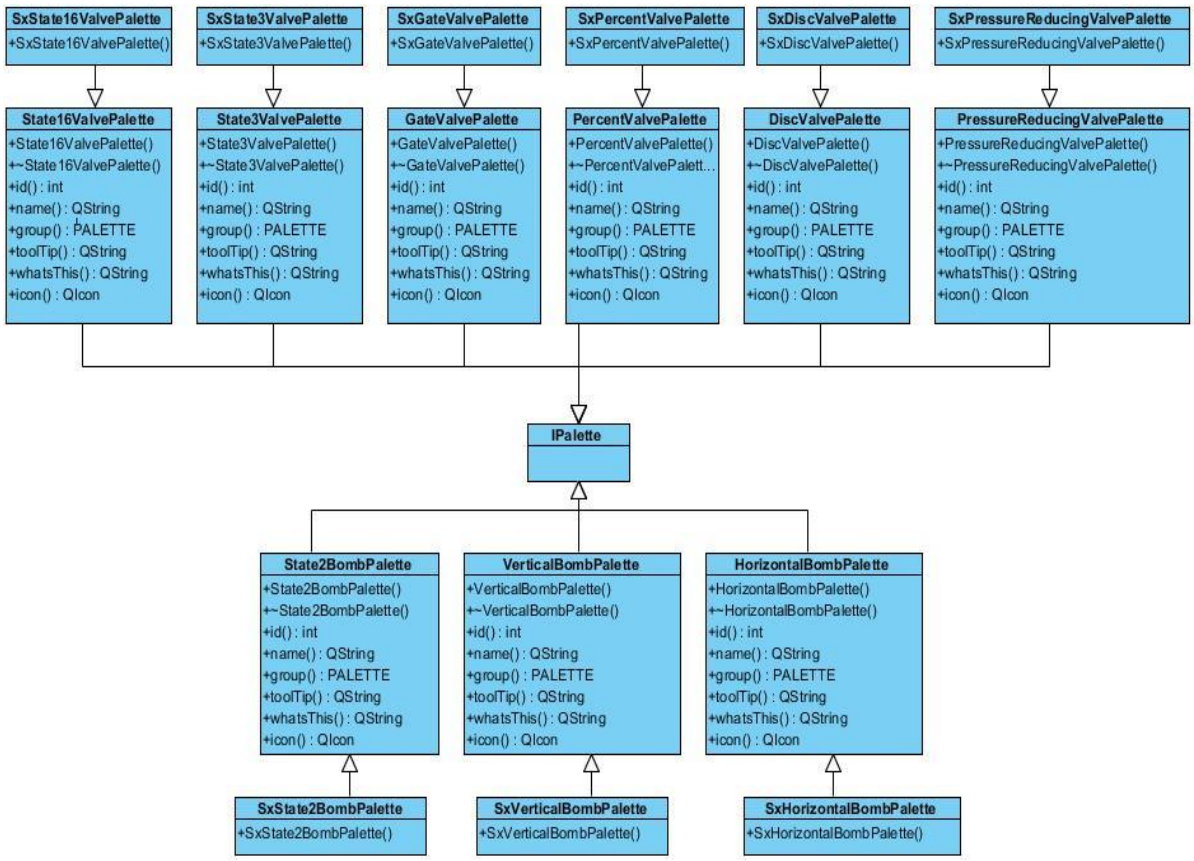


Fig.12 Diseño de Clases para las Paletas de Componentes

2.7 PATRONES DE ARQUITECTURA

Son patrones de diseño de software que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Dan una descripción de los elementos y el tipo de relación que tienen junto con un conjunto de restricciones sobre cómo pueden ser usados. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones. En comparación con los patrones de diseño, los patrones arquitectónicos tienen un nivel de abstracción mayor (22).

El patrón de arquitectura usado es el Modelo / Vista que propone Qt donde el modelo comunica con una fuente de datos, proporcionando una interfaz para los otros componentes en la arquitectura. La naturaleza de la comunicación depende del tipo de fuente de datos,

y la forma en que se implementa el modelo. La vista obtiene índices de modelo; estos son referencias a objetos de datos. Mediante el suministro de los índices del modelo, la vista puede recuperar los elementos de datos del origen de datos. En vistas estándar, un delegado hace que los elementos de datos. Cuando se edita un elemento, el delegado se comunica con el modelo usando directamente los índices modelo (23).

En general, las clases de modelo / vista se pueden separar en los tres grupos descritos anteriormente: modelos, vistas y delegados. Cada uno de estos componentes se define por las clases abstractas que proporcionan interfaces comunes y, en algunos casos, las implementaciones por defecto de características. Las clases abstractas están destinados a ser una subclase con el fin de proporcionar el conjunto completo de funcionalidad esperada por otros componentes; esto también permite que los componentes especializados puedan ser escritos.

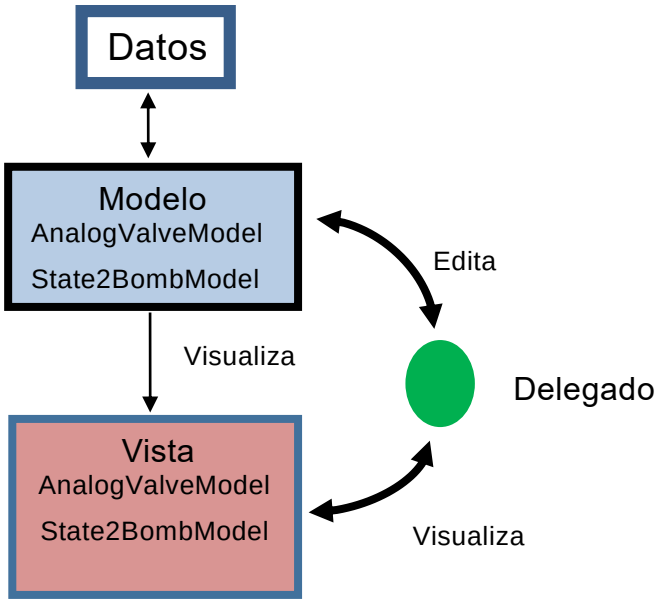


Fig.13 Patrón de Arquitectura Modelo/Vista

De las clases que componen el Modelo se encuentran:

- Para el caso de las Válvulas, se definieron las clases AnalogValveModel, PressureReducingValveModel y ValveModel.
- Para el caso de las Bombas esta la clase State2BombModel.

De las clases que componen las vistas se encuentran:

- Para las Válvulas, se definieron las siguientes clases: AnalogVariablesValves, DiscValve, GateValve, PercentValve, PressureReducingValve, State3Valve, State16Valve y Valve.
- Para el caso de las Bombas se precisaron estas: Bomb, HorizontalBomb, State2Bomb, State16Bomb y VerticalBomb.

2.8 PATRONES DE DISEÑO

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Debemos tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios) (24).

Los patrones de diseño son **soluciones para problemas típicos y recurrentes** que nos podemos encontrar a la hora de desarrollar una aplicación (24).

Existen diversas maneras de agrupar los patrones de diseño. Quizá la más extendida es agruparlos según su propósito (24). En este caso tendríamos las siguientes categorías:

- **Patrones creacionales:** utilizados para instanciar objetos, y así separar la implementación del cliente de la de los objetos que se utilizan. Con ellos intentamos separar la lógica de creación de objetos y encapsularla.
- **Patrones de comportamiento:** se utilizan a la hora de definir como las clases y objetos interaccionan entre ellos.

- **Patrones estructurales:** utilizados para crear clases u objetos que incluidos dentro de estructuras más complejas.

Los patrones GRASP utilizados fueron:

- **Experto:** Asigna una responsabilidad a la clase que cuenta con la información necesaria para cumplirla (25). Un ejemplo de la aplicación de este patrón se encuentra en la implementación de todas las clases, porque solo ellas tienen las responsabilidades que les corresponde en función de la información que trabajan.

- **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos (25). Se aplica en todos los casos donde una clase tiene la responsabilidad de crear una nueva instancia de la otra. Un ejemplo donde se utiliza este patrón es en la clase *Valve* encargada de crear la clase *ValveModel*.

- **Alta Cohesión:** Asigna una responsabilidad de modo que la cohesión siga siendo alta (25). La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Este patrón se evidencia en todas la clases ya que ellas almacenan la información necesaria para trabajar, sin involucrar otras clases.

- **Bajo Acoplamiento:** El uso de los patrones Experto y Creador contribuyen al bajo acoplamiento entre las clases del sistema. Es la idea de tener las clases lo menos ligadas entre sí que se pueda, de tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases (25). Este patrón se evidencia en todas las clases porque existe la mínima dependencia entre ellas para realizar modificaciones sin alterar su comportamiento.

Los patrones GoF utilizados fueron:

- **Observador (Observer):** Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambia de estado se notifica y actualizan automáticamente todos los objetos interesados (26). El uso de este patrón se ve

reflejado en la funcionalidad *updateState()***,** la cual es utilizada para notificarle al componente que la variable asociada a él ha sufrido cambios.

- **Plantilla (Template Method):** Este sencillo patrón resulta útil en casos en los que podamos implementar en una clase abstracta el código común que será usado por las clases que heredan de ella, permitiéndoles que implementen el comportamiento que varía mediante la reescritura (total o parcial) de determinados métodos. Este patrón se ve reflejado en los métodos *paintRuntime()*, *loadSVG()* y *paintEdition()*, dichas funcionalidades se encuentran evidenciadas en las clases de tipo vista.

2.9 CONCLUSIONES PARCIALES

- En este capítulo se comenzó a profundizar en el desarrollo de la propuesta de solución, partiendo de la elaboración del Modelo de Dominio a partir de los principales conceptos asociados obteniéndose una lista de las funcionalidades que debe tener el sistema, las mismas fueron representadas mediante historias de usuarios, y luego fueron descritas.
- Se confeccionó el Diagrama de Paquetes para representar las dependencias entre los paquetes que componen el modelo, el Diseño de Clases para los Componentes Gráficos y las Paletas de Componentes.
- Se definió como arquitectura el patrón Modelo/Vista para mostrar una interfaz de cada componente definido y como patrones el Observador y Plantilla para solucionar problemas más comunes.
- A partir de aquí se puede comenzar a construir el sistema, cumpliendo con todos los requerimientos y las funcionalidades que se consideraron en este capítulo.

CAPÍTULO III

IMPLEMENTACIÓN Y PRUEBAS

3. INTRODUCCIÓN

En este capítulo se pondrán en práctica las pruebas que detectarán posibles fallas de los componentes gráficos ya diseñados e implementados. Una vez corregidos los errores podrán ser desplegados para su utilización.

3.1 MODELO DE IMPLEMENTACIÓN

Un componente es una parte física de un sistema (módulo, base de datos, programa ejecutable, etc.). Se puede decir que un componente es la materialización de una o más clases, porque una abstracción con atributos y métodos pueden ser implementados en los componentes (29).

3.1.1 DIAGRAMA DE COMPONENTES

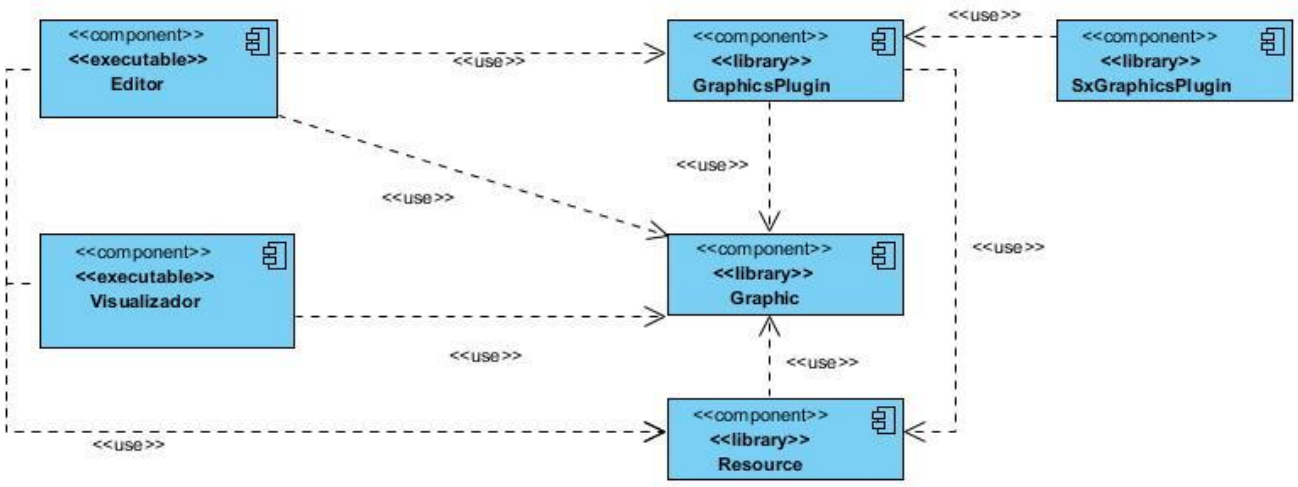


Fig.14 Diagrama de Componentes

El diagrama de componentes antes dibujado está estructurado por ejecutables y bibliotecas:

- Editor: Es la aplicación donde se trabaja en el entorno de configuración para representar los procesos del campo.

- Visualizador: Es una aplicación donde se supervisa el área configurada para interactuar con los componentes gráficos.
- Biblioteca Graphics Plugins: En esta biblioteca se almacenan los Plugins de los objetos gráficos.
- Biblioteca Graphics: Aquí están almacenados objetos gráficos que se utilizan para representar lo componentes gráficos, ya sea de forma simple o componentes complejos.
- SxGraphics Plugins: Es una interfaz de la biblioteca Graphics Plugins específica para los componentes de SAINUX.
- Resource: Biblioteca donde se encuentran agrupadas cada una de las entidades encargadas de brindar las imágenes, los iconos y los diseños de los componentes en SVG.

3.2 DIAGRAMA DE DESPLIEGUE

Un diagrama de despliegue, es un diagrama que muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos. Además muestra, las relaciones físicas entre los componentes hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos). Estarán formados por instancias de los componentes software que representan manifestaciones del código en tiempo de ejecución. A continuación se muestra el diagrama de despliegue del sistema (30).

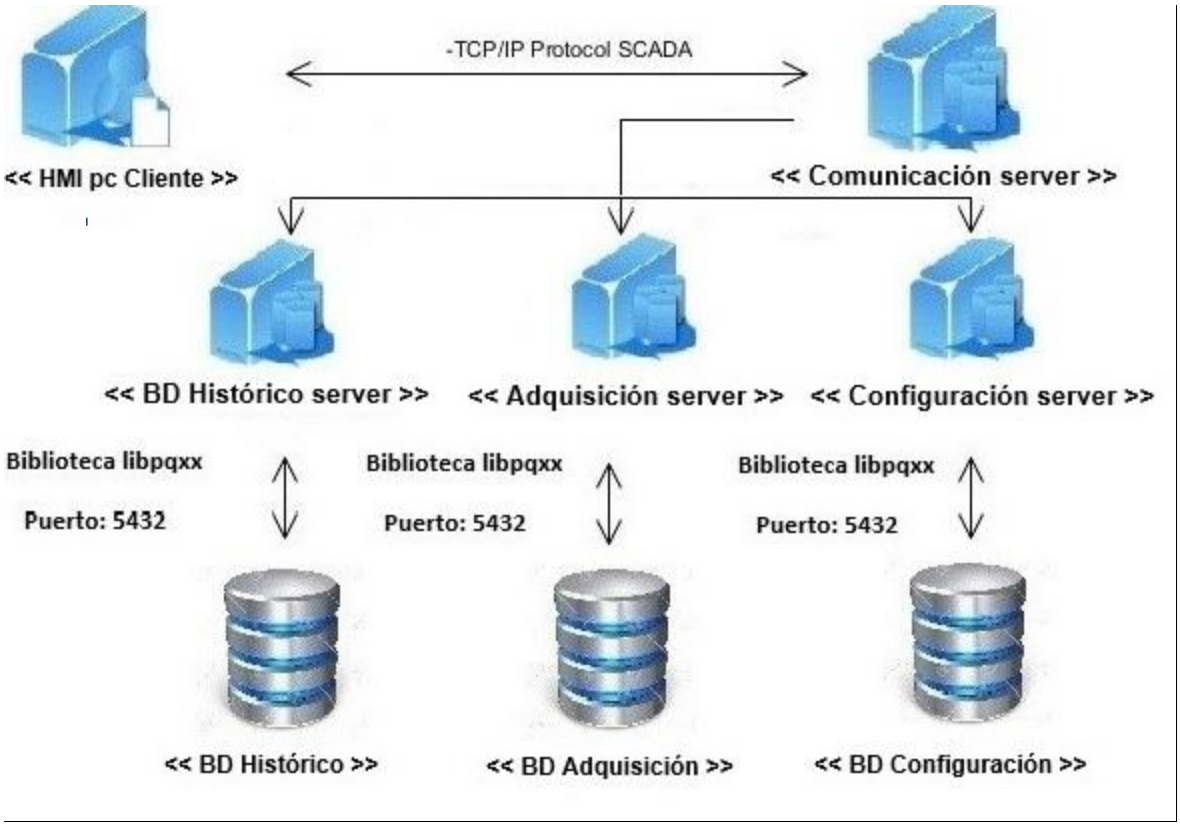


Fig.15 Diagrama de Despliegue

- **PC HMI:** En esta Pc se instala el módulo HMI, donde se encuentran desplegados el Ambiente Configuración y el Ambiente Edición, o solo uno de ellos.
- **Servidor Adquisición, Servidor Base de Datos Históricos y Servidor Configuración:** Estos módulos se ejecutan en distintas Pc que pueden servir como servidores poniendo en evidencia la arquitectura distribuida que presenta el SCADA SAINUX.
- **BD Histórico y Configuración:** Los módulos Históricos y Configuración se encuentran instalados con sus respectivas Bases de Datos conectados a través de los puertos 5432, utilizando la biblioteca libpqxx.
- La conexión entre estos módulos se realiza utilizando comunicación TCP/IP con el protocolo SCADA desarrollado por el centro.

3.3 ESTÁNDAR DE CODIFICACIÓN

Los estándares de codificación, también llamados estilos de programación o convenciones de código, son convenios para escribir código fuente en ciertos lenguajes de programación. Estos estándares facilitan el mantenimiento del código, sirven como punto de referencia para los programadores, mantienen un estilo de programación y ayudan a mejorar el proceso de codificación, entre otras cosas, haciéndolo más eficiente (31).

Como la solución propuesta en este trabajo es parte del sistema SCADA SAINUX el estándar de codificación utilizado fue definido por el proyecto:

- Es importante especificar el nombre del autor, para ello se utilizan los comandos ***@autor***.
- El código será escrito en inglés y la documentación en español.
- Las variables y funciones comienzan con letra minúscula. Cada palabra consecutiva en el nombre comienza con letra mayúscula.

  Ejemplo: void createObjectModel ();
- Los atributos de las clases deben empezar con ***m_*** seguido del nombre del atributo, en el caso de atributos compuestos, la inicial de la segunda palabra debe comenzar con mayúscula.

  Ejemplo: m_posScreen
- Las funciones utilizan la nomenclatura camello.
- Los parámetros que recibe una función debe empezar con _ (guion bajo).

  Ejemplo: void paintRuntime (QPainter *_painter, const QStyleOptionGraphicsItem *_option, QWidget *_widget);
- Todas las funcionalidades y atributos deben seguir el siguiente formato: para documentar ***@brief Nombre del método***.
- Ninguna función debe tener más de 200 líneas.
- Los valores de los numerativos deben ser con letras mayúsculas.

- Las secciones public, protected y private son declaradas en el orden expuesto.

3.4 PRUEBAS

Las pruebas de software son un elemento crítico para la garantía de la calidad del software y representan una revisión final de las especificaciones, del diseño y de la codificación. El objetivo fundamental de las pruebas es descubrir diferentes clases de errores con la menor cantidad de tiempo y de esfuerzo. Aunque las pruebas no pueden asegurar la ausencia de defectos; sí pueden demostrar que existen defectos en el software (32).

Estas actividades se planean con anticipación y se realizan de manera sistemática. Cuando se aplican pruebas a un software es necesario tener en cuenta el objetivo que se persigue, debido a que las pruebas son agrupadas por niveles que se encuentran en distintas etapas del proceso de desarrollo.

3.4.1 TIPOS DE PRUEBAS

De los distintos tipos de pruebas que se mencionan a continuación, se exponen sus objetivos y las principales características (33):

- **Pruebas Unitarias:** tiene como objetivo (busca asegurar que el código funciona de acuerdo con las especificaciones y que el módulo lógico es válido).Esta prueba se describe de la siguiente forma, se particionan los módulos en unidades lógicas fáciles de probar. Por cada unidad hay que definir los casos de prueba (pruebas de caja blanca).Para esto los casos de prueba deben diseñarse de forma tal que se recorran todos los caminos de ejecución posibles dentro del código bajo prueba; por lo tanto el diseñador debe construirlos con acceso al código fuente de la unidad a probar. Los aspectos a considerar son los siguientes: Rutinas de excepción, Rutinas de error, Manejo de parámetros, Validaciones, Valores válidos, Valores límites, Rangos, Mensajes posibles.
- **Prueba de Integración:** tiene como objetivo (Identificar errores introducidos por la combinación de programas probados unitariamente. Determina cómo la base de

datos de prueba será cargada. Verificar que las interfaces entre las entidades externas (usuarios) y las aplicaciones funcionan correctamente).Descripción: Describe cómo verificar que las interfaces entre las componentes de software funcionan correctamente. Determina cómo la base de datos de prueba será cargada. Decide qué acciones tomar cuando se descubren problemas.

- **Prueba de Regresión:** su objetivo es determinar si los cambios recientes en una parte de la aplicación tienen efecto adverso en otras partes. Descripción: En esta prueba se vuelve a probar el sistema a la luz de los cambios realizados durante el debugging, mantenimiento o desarrollo de la nueva versión del sistema buscando efectos adversos en otras partes.

- **Pruebas de Seguridad y Control de Acceso:** tiene como objetivo verifica que un actor solo pueda acceder a las funciones y datos que su usuario tiene permitido. Nivel de Seguridad del Sistema: Verificar que solo los actores con acceso al sistema y a la aplicación están habilitados para accederla. Descripción: Las pruebas de seguridad y control de acceso se centran en dos áreas claves de seguridad:

Una vez descritas las pruebas anteriores se llega a la conclusión que las pruebas de aceptación son las indicadas para evaluar el software, porque es la prueba planificada y organizada formalmente para determinar si se cumplen los requisitos de aceptación marcados por el cliente. Sus características principales son las siguientes (30):

- Participación del usuario.
- Está enfocada hacia la prueba de los requisitos de usuarios especificados.
- Está considerada como la fase final del proceso para crear una confianza en que el producto es el apropiado para su uso en explotación.

Caso de Prueba Aceptación	
Número: 1	**Historia de usuario: 1**

Nombre: Definir nombre del componente.
Descripción: Comprobar que el componente se nombra de forma correcta sin utilizar números o símbolos.
Condiciones de Ejecución: El usuario debe comprobar que se puede configurar el nombre del componente.
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Clic derecho con el ratón. - Seleccionar la opción Propiedades. - Cambiar el nombre.
Resultado esperado: El nombre ha sido cambiado.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 1. Caso de Prueba Aceptación

La descripción de las restantes pruebas de aceptación se encuentra en el anexo 2

3.4.2 DISEÑO DE CASO DE PRUEBA

Pruebas del sistema	HU	Iteración	NC	Cerrada	No Procede
	31	1ra	8	8	0
		2da	5	5	0
		3ra	1	1	0

Tabla.2 Pruebas del sistema

Una vez realizados los casos de Pruebas para un total de 31 historias de usuarios, con tres iteraciones se eliminaron las 14 No Conformidades.

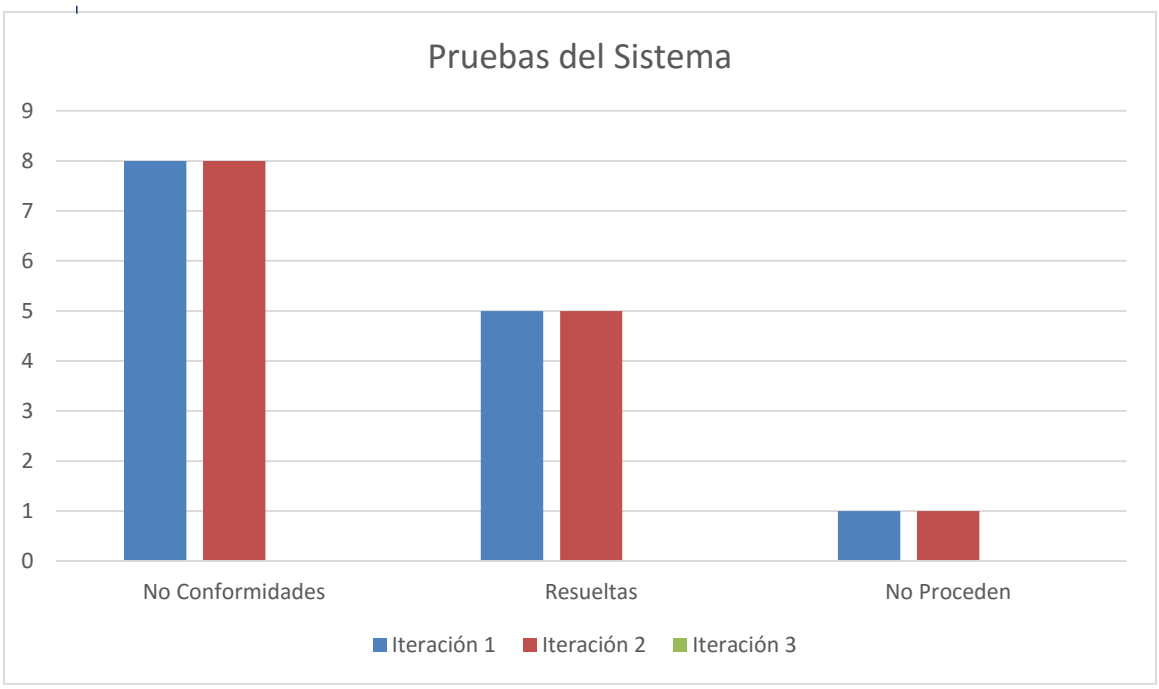


Fig.16 Plan de Iteraciones

Como resultado se detectó que los errores encontrados pueden clasificarse según su tipo:

- Para una cantidad de ocho No Conformidades se revelan errores de Presentación porque no se visualiza correctamente toda la información que se desea mostrar.
- Para las siguientes cinco No Conformidades los errores fueron de Redacción porque se localizaron varios errores ortográficos siendo estos generalmente de acentuación.
- Para la restante No conformidad el error detectado fue de Ejecución, durante el despliegue del sistema fueron detectadas diferentes fallas, siendo estas ocasionadas porque el sistema no realizaba todos los procedimientos definidos.

3.5 CONCLUSIONES PARCIALES

- En este capítulo se definió el estándar de codificación a utilizar. Se ejecutaron las pruebas de aceptación para darle validez a los requisitos descritos en las historias de usuarios.
- Se presentó la distribución física del sistema y sus componentes mediante el diagrama de componentes, permitiendo un mejor entendimiento de la distribución física y lógica del sistema.
- A partir del resultado arrojado por las pruebas realizadas se llegó a la conclusión de que el mecanismo se encuentra listo para su puesta en funcionamiento.

CONCLUSIONES GENERALES

Una vez concluida la presenta investigación se logró:

- Contribuir a elevar los grados de configuración, visualización y operabilidad de los dispositivos de control utilizados en las redes de tuberías de la Interfaz Hombre Máquina del sistema SCADA SAINUX con el desarrollo de componentes gráficos para la representación de válvulas y bombas.
- La validación del sistema propuesto en este trabajo a través de un proceso de prueba que arrojo que la solución está lista para ser usada.
- Un mecanismo que permite asociar varias mediciones a un componente, definiendo dentro de este las variables o los parámetros que van a representar dichas mediciones dentro de ese componente.

Destacar que todo el desarrollo del trabajo se utilizó herramientas y tecnologías libres.

RECOMENDACIONES

Se recomiẻnda seguir ampliando la gama de componentes gráficos que representan válvulas y bombas.

Incorporar los componentes gráficos en el visor Web que está desarrollado en el Centro CEDIN.

BIBLIOGRAFÍA

1. **García, Darlin Mercedes Blanco. *Simulador de Variables Analógicas y Digitales para la Interfaz Hombre Máquina (HMI) del SCADA Guardián del Alba.* 2013.**

2. **Ingeniería de Software. [En línea] 15 de marzo de 2016. http://clases3gingsof.wikifoundry.com/page/Proceso+de+Despliegue+de+RUP.**

3. Dagoberto Monteros, David B. Barrantes y José M. Quirós. *Introducción a los sistemas de control, supervisión y adquisición de datos (SCADA).* 2004.

4. es.scrib.com. [En línea] 15 de octubre de 2016. http://es.scribd.com/doc/284387390/2-4-Sistema-HMI#scribd.

5. Prezi. [En línea] 21 de abril de 2016. https://prezi.com/yzgen1zgicz5/sistemas-de-redes-de-tuberias/.

6. fluidos.eia.cu. [En línea] http://fluidos.eia.edu.co/hidraulica/articuloses/flujoentuberias/metodohardycross/introduccion.html.

7. www.sc.ehu.es. [En línea] http://www.sc.ehu.es/sbweb/webcentro/automatica/WebCQMH1/PAGINA%20PRINCIPAL/Automatizacion/Automatizacion.htm.

8. www.quiminet.com. [En línea] http://www.quiminet.com/articulos/conozca-el-funcionamiento-de-una-valvula-automatica-2747620.html.

9. ALEGSA.com.ar. DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA. Definición de SVG. [En línea] 17 de enero de 2016. http://www.alegsa.com.ar/Dic/svg.php.

10. Mad, Freepress S. Coop. Freepress S. Coop. Inkscape-software-libre-para-diseño-vectorial. [En línea] 2 de diciembre de 2015. http://www.freepress.coop/recursos/inkscape-software-libre-para-diseno-vectorial/.

11. Oliag, Sergio Talens. Curso de Programación en C++. [En línea] 30 de octubre de 2015. http://www.uv.es/~sto/cursos/c++/curso95.pdf.

12. Raydel Raúl Viñolo Sosa, Alexander Roquero Figueroa. Sistema Gestor de Proceso de Media v2. [En línea] 2 de diciembre de 2015. http://publicaciones.uci.cu/index.php/SC | seriecientifica@uci.cu.

13. CORPORATION, NOKIA. Qt. [En línea] 2008. [Citado el: 2 de diciembre de 2015.] http://qt.nokia.com/products/developer-tools/.

14. Cornejo, José Enrique González. DOCIRS. . [En línea] enero de 2008. [Citado el: 10 de diciembre de 2015.] http://www.docirs.com/uml.htm.

15. CMM. CMM.net. [En línea] 3 de diciembre de 2015. http://es.ccm.net/download/descargar-28127-visual-paradigm-for-uml-enterprise-edition.

16. Tamara Rodríguez Sánchez. MEJORA, PROGRAMA DE. Metodología de Desarrollo para la Actividad Productiva de la UCI. [En línea] http://excriba.prod.uci.cu/page/context/shared/sharedfiles/Metodologiauci.pdf.

17. Debian. The Universal Operating System. [En línea] http://www.debian.org.

18. Larman, Craig. *UML y Patrones. UML y Patrones. Modelo de Dominio. . Prentice Hall: s.n.* 2003.

19. bligoo. Ingeniería de Software. [En línea] [Citado el: 26 de febrero de 2016.] http://ingenieriadesoftware.bligoo.com.mx/requerimientos-funcionales-y-no-funcionales-rf-rnf.

20. tecnicas-para-identificar-requisitos-funcionales-y-no-funcionales. [En línea] https://sites.google.com/site/metodologiareq/capitulo-ii/tecnicas-para-identificar-requisitos-funcionales-y-no-funcionales.

21. Juan Quijano. GENBETA: dev. Historias de usuario, una forma natural de análisis funcional. [En línea] http://www.genbetadev.com/metodologias-de-programacion/historias-de-usuario-una-forma-natural-de-analisis-funcional.

22. Patrones de arquitectura VS patrones de diseño. [En línea] https://arlethparedes.wordpress.com/2012/08/27/patrones-de-arquitectura-vs-patrones-de-diseno/.

23. Qt Documentation. [En línea] [Citado el: 16 de marzo de 2016.] http://doc.qt.io/qt-4.8/model-view-programming.html.

24. Marcello Visconti, Hernán Astudillo. Fundamentos de Ingeniería de Software. [En línea] https://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.

25. [En línea] Gamma, E., y otros. Design Patterns. Elements of Reusable Object-Oriented Software.

26. [En línea] 2008. Timothy G. Mattson, Beverly A. Sanders, Berna L. Massingill. Patterns for Parallel Programming. S.l.: Addison-Wesley,

27. Microsoft. Diagramas de secuencia UML: Referencia. [En línea] [Citado el: 10 de marzo de 2016.] https://msdn.microsoft.com/es-es/library/dd409377.aspx.

28. Principiosdeldisenodelsoftware. [En línea] http://ocw.usal.es/ensenanzas-tecnicas/ingenieria-delsoftware/contenidos/Tema5-Principiosdeldisenodelsoftware-1pp.pdf.

29. Arquitectura de software. Diagrama de componentes. [En línea] [Citado el: 20 de marzo de 2016.] http://es.scribd.com/doc/7884665/Arquitectura-de-Software-II-Diagrama-de-Componentes-yDespliegue.

30. Diagrama de Despliegue. ANALISIS Y DISEÑO DE SISTEMAS II. [En línea] [Citado el: 26 de mayo de 2016.] virtual.usalesiana.edu.bo/web/practica/archivo/despliegue.doc.

31. Estándares de Codificación .Net v.1.0.0.0. [En línea] (42. Estándares de Codificación .Net v.1.0.0.0. http: //serk.kualtus.com/codigo.htm.).

32. PRESSMAN, R. S. *Ingeniería de Software, un enfoque práctico. Sexta edición ed.*

33. EliuMM. [En línea]
http://javablog.eliumontoya.com/home/tipodepruebasparadesarrollodesoftware.

34. [En línea] http: //serk.kualtus.com/codigo.htm.).

ANEXO 1 HISTORIAS DE USUARIOS

	Historia de Usuario
Número: 2	**Nombre de Historia:** Permitir la rotación del componente.
Actor: Usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Medio	**Puntos estimados:** 0.3
Nivel de Complejidad: Medio	**Puntos reales:** 0.3
Descripción: Establece un ángulo de rotación. Definir donde localizar el centro de rotación. El centro de rotación se puede localizar en distintas posiciones, en la parte inferior del widget, en la parte superior, en el centro, a la derecha o a la izquierda.	

Historia de usuario #2

	Historia de Usuario
Número: 3	**Nombre de Historia:** Configurar la posición del componente.
Actor: Usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Medio	**Puntos estimados:** 0.5
Nivel de Complejidad: Media	**Puntos reales:** 0.5

Descripción: El sistema proporciona una interfaz al usuario donde puede definir la posición en la que se traslada el componente sobre el eje X y el eje Y, definido en el rango (0; 2147483647).

Historia de usuario #3

Historia de Usuario	
Número: 4	**Nombre de Historia:** Configurar la dimensión del componente.
Actor: Usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Medio	**Puntos estimados:** 0.3
Nivel de Complejidad: Medio	**Puntos reales:** 0.3
Descripción: El sistema proporciona una interfaz al usuario donde puede configurar el ancho y el alto que tendrá el componente.	

Historia de usuario #4

Historia de Usuario	
Número: 5	**Nombre de Historia:** Asociar variables al componente.
Actor: Usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos estimados:** 0.6
Nivel de Complejidad: Alta	**Puntos reales:** 0.6
Descripción: Permitir la selección de las mediciones (digitales o analógicas) que indican el estado (abierto, semiabierto, cerrado) o parámetros que se controlan del componente.	

Historia de usuario #5

Historia de Usuario	
Número: 6	**Nombre de Historia:** Asociar mediciones relacionadas con el componente a los parámetros medible de este.
Actor: Usuario	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos estimados:** 0.6
Nivel de Complejidad: Alta	**Puntos reales:** 0.6
Descripción: El sistema brinda una interfaz al usuario donde puede asociar mediciones (analógicas o digitales) al componente según el estado que presenta y si está fluyendo líquido o gases.	

Historia de usuario #6

Historia de Usuario	
Número: 7	**Nombre de Historia:** Configurar el tipo de control del componente.
Actor: Usuario	**Iteración Asignada:** 2
Prioridad en Negocio: Alta	**Puntos estimados:** 0.6
Nivel de Complejidad: Alta	**Puntos reales:** 0.6
Descripción: El sistema brinda una interfaz al usuario donde puede definir el tipo de control (manual o automático) que pueda efectuarse sobre el componente, desde el inspector de propiedades**.**	

Historia de usuario #7

Historia de Usuario	
Número: 8	**Nombre de Historia:** Configurar correspondencia entre los valores discretos.

Actor: Usuario	**Iteración Asignada:** 2
Prioridad en Negocio: Alta	**Puntos estimados:** 0.6
Nivel de Complejidad: Alta	**Puntos reales:** 0.6
Descripción: El sistema brinda una interfaz al usuario donde puede establecer el valor que representa el estado en que se encuentra.	

Historia de usuario #8

Historia de Usuario	
Número: 9	**Nombre de Historia:** Configurar la opacidad del componente.
Actor: Usuario	**Iteración Asignada:** 2
Prioridad en Negocio: Baja	**Puntos estimados:** 0.2
Nivel de Complejidad: Baja	**Puntos reales:** 0.2
Descripción: Encargado de definir la transparencia que tendrá el componente representado en el rango de 0 a 100.	

Historia de usuario #9

Historia de Usuario	
Número: 10	**Nombre de Historia:** Configurar color de la tubería del componente.
Actor: Usuario	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos estimados:** 0.4
Nivel de Complejidad: Media	**Puntos reales:** 0.4

Descripción: El usuario configura el color de la tubería para representar cuando está fluyendo o no líquido.

Historia de usuario #10

	Historia de Usuario
Número: 11	**Nombre de Historia:** Configurar color de la brida del componente.
Actor: Usuario	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos estimados:** 0.4
Nivel de Complejidad: Media	**Puntos reales:** 0.4
Descripción: El sistema brinda una interfaz al usuario donde puede configurar el color de la brida para representar cuando se han unido dos componentes.	

Historia de usuario #11

	Historia de Usuario
Número: 12	**Nombre de Historia:** Mostrar u ocultar bridas del componente.
Actor: Usuario	**Iteración Asignada:** 3
Prioridad en Negocio: Media	**Puntos estimados:** 0.4
Nivel de Complejidad: Media	**Puntos reales:** 0.4
Descripción: El sistema brinda una interfaz al usuario donde puede mostrar u ocultar las bridas pertenecientes al componente.	

Historia de usuario #12

Historia de Usuario	
Número: 13	**Nombre de Historia:** Configurar el color que representa el estado indefinido del componente.
Actor: Usuario	**Iteración Asignada:** 3
Prioridad en Negocio: Alta	**Puntos estimados:** 1
Nivel de Complejidad: Alta	**Puntos reales:** 1
Descripción: El sistema brinda una interfaz donde puede configurar el color que representa el estado indefinido como negro para expresar que ha ocurrido un evento extraño.	

Historia de usuario #13

Historia de Usuario	
Número: 14	**Nombre de Historia:** Mostrar/ocultar por ciento.
Actor: Usuario	**Iteración Asignada:** 3
Prioridad en Negocio: Alta	**Puntos estimados:** 0.3
Nivel de Complejidad: Alta	**Puntos reales:** 0.3
Descripción: El usuario puede mostrar u ocultar el por ciento de llenado definido para este componente, en el rango de 0 a 100.	

Historia de usuario #14

Historia de Usuario	
Número: 15	**Nombre de Historia:** Configurar dimensiones del Meter Emergente.
Actor: Usuario	**Iteración Asignada:** 3

Prioridad en Negocio: media	**Puntos estimados:** 1.1
Nivel de Complejidad: Media	**Puntos reales:** 1.1
Descripción: El usuario configura las dimensiones del Meter Emergente para mostrarlo en diferentes tamaños. El rango no debe exceder de (0; 2147483647).	

Historia de usuario #15

Historia de Usuario	
Número: 16	**Nombre de Historia:** Configurar color del anillo.
Actor: Usuario	**Iteración Asignada:** 3
Prioridad en Negocio: Media	**Puntos estimados:** 0.1
Nivel de Complejidad: Baja	**Puntos reales:** 0.1
Descripción: El usuario configura el color del anillo según desee.	

Historia de usuario #16

Historia de Usuario	
Número: 17	**Nombre de Historia:** Configurar color del plato.
Actor: Usuario	**Iteración Asignada:** 4
Prioridad en Negocio: Media	**Puntos estimados:** 0.5
Nivel de Complejidad: Baja	**Puntos reales:** 0.5
Descripción: El usuario configura el color del plato según desee.	

Historia de usuario #17

Historia de Usuario	
Número: 18	**Nombre de Historia:** Configurar color de la aguja.
Actor: Usuario	**Iteración Asignada:** 4
Prioridad en Negocio: Baja	**Puntos estimados:** 0.5
Nivel de Complejidad: Baja	**Puntos reales:** 0.5
Descripción: El usuario configura el color de la aguja según desee.	

Historia de usuario #18

Historia de Usuario	
Número: 19	**Nombre de Historia:** Configurar color de la posición del valor.
Actor: Usuario	**Iteración Asignada:** 4
Prioridad en Negocio: Baja	**Puntos estimados:** 0.5
Nivel de Complejidad: Baja	**Puntos reales:** 0.5
Descripción: El usuario configura el color de la posición del valor según desee.	

Historia de usuario #19

Historia de Usuario	
Número: 20	**Nombre de Historia:** Configurar color del valor.
Actor: Usuario	**Iteración Asignada:** 4
Prioridad en Negocio: Baja	**Puntos estimados:** 0.6

Nivel de Complejidad: Baja	**Puntos reales:** 0.6
Descripción: El usuario configura el color de la posición del valor según desee.	

Historia de usuario #20

	Historia de Usuario
Número: 21	**Nombre de Historia:** Configurar tamaño de la fuente del valor.
Actor: Usuario	**Iteración Asignada:** 4
Prioridad en Negocio: Media	**Puntos estimados:** 0.4
Nivel de Complejidad: Media	**Puntos reales:** 0.4
Descripción: El usuario establece un tamaño adecuado de fuente para trabajar con el valor, en el rango de 8 a 72.	

Historia de usuario #21

	Historia de Usuario
Número: 22	**Nombre de Historia:** Mostrar valor.
Actor: Usuario	**Iteración Asignada:** 5
Prioridad en Negocio: Alta	**Puntos estimados:** 1
Nivel de Complejidad: Alta	**Puntos reales:** 1
Descripción: El usuario debe mostrar el valor que indica el estado de por ciento de fluido que atraviesa la tubería.	

Historia de usuario #22

Historia de Usuario	
Número: 23	**Nombre de Historia:** Configurar color de la escala.
Actor: Usuario	**Iteración Asignada:** 5
Prioridad en Negocio: Baja	**Puntos estimados:** 0.4
Nivel de Complejidad: Baja	**Puntos reales:** 0.4
Descripción: El usuario configura el color de la posición del valor según desee.	

Historia de usuario #23

Historia de Usuario	
Número: 24	**Nombre de Historia:** Configurar tamaño de la fuente de la escala.
Actor: Usuario	**Iteración Asignada:** 5
Prioridad en Negocio: Media	**Puntos estimados:** 0.6
Nivel de Complejidad: Media	**Puntos reales:** 0.6
Descripción: El usuario configura el tamaño de la fuente con la que desea trabajar, en el rango de 8 a 72.	

Historia de usuario #24

Historia de Usuario	
Número: 25	**Nombre de Historia:** Mostrar/ocultar escala numérica.
Actor: Usuario	**Iteración Asignada:** 6

Prioridad en Negocio: Media	**Puntos estimados:** 1
Nivel de Complejidad: Media	**Puntos reales:** 1
Descripción: El usuario puede mostrar la escala que indican los valores del flujómetro u ocultarlos.	

Historia de usuario #25

Historia de Usuario	
Número: 26	**Nombre de Historia:** Configurar posición de la unidad de medida.
Actor: Usuario	**Iteración Asignada:** 6
Prioridad en Negocio: Alta	**Puntos estimados:** 1
Nivel de Complejidad: Alta	**Puntos reales:** 1
Descripción: El usuario configura la posición de la unidad de medida.	

Historia de usuario #26

Historia de Usuario	
Número: 27	**Nombre de Historia:** Configurar tamaño de fuente de la unidad.
Actor: Usuario	**Iteración Asignada:** 6
Prioridad en Negocio: Alta	**Puntos estimados:** 0.6
Nivel de Complejidad: Media	**Puntos reales:** 0.6

Descripción: El usuario configura el tamaño de la unidad para mostrar el valor, en el rango de 8 a 72.

Historia de usuario #27

	Historia de Usuario
Número: 28	**Nombre de Historia:** Mostrar unidad de medida.
Actor: Usuario	**Iteración Asignada:** 7
Prioridad en Negocio: Alta	**Puntos estimados:** 0.4
Nivel de Complejidad: Alta	**Puntos reales:** 0.4
Descripción: El usuario debe mostrar la unidad de medida con la que trabaja el flujómetro.	

Historia de usuario #28

	Historia de Usuario
Número: 29	**Nombre de Historia:** Configurar tamaño de fuente de la unidad.
Actor: Usuario	**Iteración Asignada:** 7
Prioridad en Negocio: Alta	**Puntos estimados:** 0.7
Nivel de Complejidad: Alta	**Puntos reales:** 0.7
Descripción: El usuario configura el tamaño de fuente de la unidad, en el rango de 8 a 72.	

Historia de usuario #29

	Historia de Usuario

Número: 30	**Nombre de Historia:** Permitir ejecución de comando sobre las mediciones asociadas al componente.
Actor: Usuario	**Iteración Asignada:** 8
Prioridad en Negocio: Alta	**Puntos estimados:** 1.5
Nivel de Complejidad: Alta	**Puntos reales:** 1.5
Descripción: El usuario debe permitir ejecutar comandos que se reciben a través de la medición analógica para representar el estado del componente.	

Historia de usuario #30

Historia de Usuario	
Número: 31	**Nombre de Historia:** Representar en el componente el estado en el que se encuentra.
Actor: Usuario	**Iteración Asignada:** 9
Prioridad en Negocio: Alta	**Puntos estimados:** 1.5
Nivel de Complejidad: Alta	**Puntos reales:** 1.5
Descripción: El usuario debe representar los estados en que se encuentra el componente.	

Historia de usuario #31

ANEXO 2 PRUEBAS DE ACEPTACIÓN

Caso de Prueba Aceptación	
Número: 2	**Historia de usuario: 2**
Nombre: Permitir la rotación del componente.	
Descripción: Comprobar que el widget rota 15° sobre el eje z.	
Condiciones de Ejecución: El usuario debe comprobar que el componente rota sobre el eje z.	
Entradas/ Pasos de Ejecución: - Seleccionar el objeto. - Seleccionar en el menú contextual la opción rotar izquierda. - Seleccionar en el menú contextual la opción rotar derecha.	
Resultado esperado: El usuario pudo rotar el componente sobre cualquiera de las dos opciones.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #2

Caso de Prueba Aceptación	
Número: 3	**Historia de usuario: 3**

Nombre: Configurar la posición del componente.
Descripción: Comprobar que el objeto se traslada sobre cualquier punto, definido en el rango (- 2147483647; 2147483647).
Condiciones de Ejecución: El usuario debe comprobar que el componente se puede mover sobre el despliegue.
Entradas/ Pasos de Ejecución: - Seleccionar el componente en el menú contextual. - Ubicarlo en el despliegue. - Seleccionar el componente y trasladarlo sobre cualquier punto.
Resultado esperado: El componente se traslada sobre el despliegue en cualquier punto.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #3

Caso de Prueba Aceptación	
Número: 4	**Historia de usuario: 4**
Nombre: Configurar la dimensión del componente.	
Descripción: Comprobar que se puede configurar la dimensión del componente, definido en el rango (- 2147483647; 2147483647).	
Condiciones de Ejecución: El usuario debe comprobar que el objeto puede ser configurado en ancho y alto.	
Entradas/ Pasos de Ejecución: - Seleccionar el objeto. - Con el clic izquierdo seleccionar uno de los puntos donde puede redimensionarse para agrandarlo o achicarlo.	

Resultado esperado: El componente se redimensiona sin perder calidad.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #4

Caso de Prueba Aceptación	
Número: 5	**Historia de usuario: 5**
Nombre: Asociar variables al componente.	
Descripción: Comprobar que se le asocian variable al componente para representar el estado.	
Condiciones de Ejecución: El usuario debe comprobar que las variables asociadas representan el estado en el que se encuentra.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir el inspector de propiedades. - Seleccionar la propiedad medición. - Adicionar mediciones.	
Resultado esperado: Las variables asociadas representan el estado del componente.	
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.	

Prueba de Aceptación #5

Caso de Prueba Aceptación

Número: 6	**Historia de usuario: 6**
Nombre: Asociar mediciones relacionadas con el componente a los parámetros medible de este.	
Descripción: Comprobar que las mediciones asociadas al componente se puedan asociar con parámetros individuales a los parámetro medibles de este.	
Condiciones de Ejecución: El usuario debe comprobar que las condiciones asociadas al componente fueron realizadas a los parámetros medibles de este.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar propiedad medición. - Seleccionar estado de la válvula o fluido de la válvula. - Seleccionar un parámetro para las mediciones asociadas.	
Resultado esperado: Las mediciones fueron asociadas a los parámetros medibles del componente.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #6

Caso de Prueba Aceptación	
Número: 7	**Historia de usuario: 7**
Nombre: Configurar el tipo de control del componente.	
Descripción: Configurar los dos tipos de control que pueden efectuarse sobre el componente (manual o automático), donde el primero permite que el operador desde el ambiente de visualización pueda cambiar el estado del dispositivo, mientras con el segundo el estado del dispositivo solo cambiará por una programación propia del dispositivo.	

Condiciones de Ejecución: Para asignar el tipo de control el componente debe estar seccionado se debe estar visualizando el inspector de propiedades.
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar Tipo de Control. - Seleccionar Manual o Automático.
Resultado esperado: Se logró configurar el tipo de control.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #7

Caso de Prueba Aceptación	
Número: 8	**Historia de usuario: 8**
Nombre: Configurar correspondencia entre los valores discretos.	
Descripción: Configurar el color que representa el valor del estado.	
Condiciones de Ejecución: El usuario debe comprobar que no existe un mismo color para representa dos estados más.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar la opción Color del Estado. - Seleccionar el color que representa el estado en que se encuentra el componente.	

Resultado esperado: El estado del componente es representado por un color.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #8

Caso de Prueba Aceptación	
Número: 9	**Historia de usuario: 9**
Nombre: Configurar la opacidad del componente.	
Descripción: Comprobar que al componente puede configurarse su opacidad.	
Condiciones de Ejecución: El usuario debe comprobar que se cumple la propiedad de opacidad para el componente.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir el inspector de propiedades. - Seleccionar opción de opacidad.	
Resultado esperado: La opción de opacidad para el componente es configurable.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #9

Caso de Prueba Aceptación	
Número: 10	**Historia de usuario: 10**
Nombre: Configurar color de la tubería del componente.	

Descripción: Comprobar que el color de la tubería puede ser configurable para diferenciarlo de los demás atributos del objeto.
Condiciones de Ejecución: El usuario debe comprobar que el color de la tubería pueda ser cambiado por otro color**.**
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color de la tubería. - Seleccionar color del panel de colores.
Resultado esperado: El color de la tubería puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #10

Caso de Prueba Aceptación	
Número: 11	**Historia de usuario: 11**
Nombre: Configurar color de la brida del componente.	
Descripción: Comprobar que el color de la brida pueda ser configurable para diferenciarlo de los demás atributos del componente.	
Condiciones de Ejecución: El usuario debe comprobar que el color de la brida pueda ser cambiado por otro color.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades.	

- Seleccionar opción Color de la brida. - Seleccionar color del panel de colores.
Resultado esperado: El color de la brida puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #11

Caso de Prueba Aceptación	
Número: 12	**Historia de usuario: 12**
Nombre: Mostrar u ocultar bridas del componente.	
Descripción: Verificar que la opción de mostrar/ ocultar bridas es efectiva.	
Condiciones de Ejecución: El usuario debe comprobar que las bridas pueden mostrarse y ocultarse según desee.	
Entradas/ Pasos de Ejecución: - Seleccionar componente. - Abrir inspector de propiedades. - Marcar o desmarcar opción de mostrar/ocultar bridas.	
Resultado esperado: Las bridas pueden marcarse o desmarcarse.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #12

Caso de Prueba Aceptación	
Número: 13	**Historia de usuario: 13**
Nombre: Configurar el color que representa el estado indefinido del componente.	

Descripción: Comprobar que el color que representa el estado indefinido es distinto a los colores que se utilizan para representar algún estado del componente.
Condiciones de Ejecución: El usuario debe comprobar que el color del estado indefinido es negro.
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Clic izquierdo con el ratón. - Visualizar la paleta de colores. - Seleccionar color negro.
Resultado esperado: El color del estado indefinido es el negro.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #13

Caso de Prueba Aceptación	
Número: 14	**Historia de usuario: 14**
Nombre: Mostrar/ocultar por ciento.	
Descripción: Verificar que la opción de mostrar/ ocultar por ciento es efectiva.	
Condiciones de Ejecución: El usuario debe comprobar que el por ciento de llenado de la válvula de porciento pueden mostrarse y ocultarse según desee.	
Entradas/ Pasos de Ejecución: - Seleccionar componente. - Abrir inspector de propiedades. - Marcar o desmarcar opción de mostrar/ocultar por ciento.	

Resultado esperado: El por ciento puede marcarse o desmarcarse.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #14

Caso de Prueba Aceptación	
Número: 15	**Historia de usuario: 15**
Nombre: Configurar dimensiones del Meter Emergente.	
Descripción: El usuario debe comprobar que las dimensiones pueden ser configurables.	
Condiciones de Ejecución: El usuario debe comprobar que el Meter Emergente puede ser configurado en ancho y alto.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Con el clic izquierdo seleccionar uno de los puntos donde puede redimensionarse para agrandarlo o achicarlo.	
Resultado esperado: El Meter Emergente puede redimensionarse sin perder calidad.	
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.	

Prueba de Aceptación #15

Caso de Prueba Aceptación	
Número: 16	**Historia de usuario: 16**
Nombre: Configurar color del anillo.	

Descripción: Comprobar que el color del anillo pueda ser configurable para diferenciarlo de los demás atributos del componente.
Condiciones de Ejecución: El usuario debe comprobar que el color del anillo pueda ser cambiado por otro color.
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color del anillo. - Seleccionar color del panel de colores.
Resultado esperado: El color del anillo puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #16

Caso de Prueba Aceptación	
Número: 17	**Historia de usuario: 17**
Nombre: Configurar color del plato.	
Descripción: Comprobar que el color del plato pueda ser configurable para diferenciarlo de los demás atributos del componente.	
Condiciones de Ejecución: El usuario debe comprobar que el color del plato pueda ser cambiado por otro color.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color del plato.	

- Seleccionar color del panel de colores.
Resultado esperado: El color del anillo puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #17

Caso de Prueba Aceptación	
Número: 18	**Historia de usuario: 18**
Nombre: Configurar color de la aguja.	
Descripción: Comprobar que el color de la aguja pueda ser configurable para denotarlo con un color deseado	
Condiciones de Ejecución: El usuario debe comprobar que el color de la aguja pueda ser cambiado por otro color.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color de la aguja. - Seleccionar color del panel de colores.	
Resultado esperado: El color de la aguja puede ser cambiado por otro cualquiera.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #18

Caso de Prueba Aceptación	
Número: 19	**Historia de usuario: 19**
Nombre: Configurar color de la posición del valor.	

Descripción: Comprobar que el color de la posición del valor pueda ser configurable para denotarlo con un color deseado.
Condiciones de Ejecución: El usuario debe comprobar que el color de la posición del valor pueda ser cambiado por otro color.
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color de la posición del valor. - Seleccionar color del panel de colores.
Resultado esperado: El color de la posición del valor puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #19

Caso de Prueba Aceptación	
Número: 20	**Historia de usuario: 20**
Nombre: Configurar color del valor.	
Descripción: Comprobar que el color de la posición del valor pueda ser configurable para diferenciarlo del color de la posición del valor.	
Condiciones de Ejecución: El usuario debe comprobar que el color del valor pueda ser cambiado por otro color.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color del valor.	

- Seleccionar color del panel de colores.
Resultado esperado: El color del valor puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #20

Caso de Prueba Aceptación	
Número: 21	**Historia de usuario: 21**
Nombre: Configurar tamaño de la fuente del valor.	
Descripción: Comprobar que se pueda configurar el tamaño de la fuente del valor para mostrar en diferentes dimensiones la fuente.	
Condiciones de Ejecución: El usuario debe comprobar que el tamaño de la fuente puede ser vista en distintos tamaños.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar Tamaño de fuente	
Resultado esperado: El tamaño de la fuente del valor puede ser cambiado.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #21

Caso de Prueba Aceptación	
Número: 22	**Historia de usuario: 22**
Nombre: Mostrar valor.	

Descripción: Verificar que el valor se muestre.
Condiciones de Ejecución: El usuario debe comprobar que el valor muestre el estado de llenado o vacío del componente.
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Mostrar valor.
Resultado esperado: El valor se muestra correctamente.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #22

Caso de Prueba Aceptación	
Número: 23	**Historia de usuario: 23**
Nombre: Configurar color de la escala.	
Descripción: Comprobar que el color de la escala pueda ser configurable para denotarlo con otro color.	
Condiciones de Ejecución: El usuario debe comprobar que el color de la escala puede ser modificado.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar opción Color de la escala. - Seleccionar color del panel de colores	

Resultado esperado: El color de la escala puede ser cambiado por otro cualquiera.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #23

Caso de Prueba Aceptación	
Número: 24	**Historia de usuario: 24**
Nombre: Configurar tamaño de la fuente de la escala.	
Descripción: Comprobar que se pueda configurar el tamaño de la fuente de la escala para mostrar en diferentes dimensiones la fuente.	
Condiciones de Ejecución: El usuario debe comprobar si el tamaño de la escala puede redimensionarse.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir inspector de propiedades. - Seleccionar la opción Tamaño de la escala. - Cambiar tamaño.	
Resultado esperado: El tamaño de la escala puede cambiarse.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #24

Caso de Prueba Aceptación	
Número: 25	**Historia de usuario: 25**
Nombre: Mostrar/ocultar escala numérica.	

Descripción: Verificar que la opción de mostrar/ ocultar escala numérica es efectiva.
Condiciones de Ejecución: El usuario debe comprobar que la escala numérica pueden mostrarse y ocultarse según desee.
Entradas/ Pasos de Ejecución: - Seleccionar componente. - Abrir inspector de propiedades. - Marcar o desmarcar opción de mostrar/ocultar escala numérica.
Resultado esperado: La escala numérica pueden marcarse o desmarcarse
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #25

Caso de Prueba Aceptación	
Número: 26	**Historia de usuario: 26**
Nombre: Configurar posición de la unidad de medida.	
Descripción: Comprobar que la unidad de medida que representa el estado del componente puede ubicarse en cualquier punto del despliegue**.**	
Condiciones de Ejecución: El usuario debe comprobar que la posición de la unidad de medida puede trasladarse.	
Entradas/ Pasos de Ejecución: - Seleccionar la unidad de medida. - Ubicar con el ratón en cualquier punto del despliegue.	
Resultado esperado: La posición de la unidad de medida es configurable.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #26

Caso de Prueba Aceptación	
Número: 27	**Historia de usuario: 27**
Nombre: Configurar tamaño de fuente de la unidad.	
Descripción: Comprobar que se pueda configurar el tamaño de la fuente del valor para mostrar en diferentes dimensiones la fuente.	
Condiciones de Ejecución: El usuario debe comprobar que el tamaño de la fuente puede redimensionarse.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Abrir el inspector de propiedades. - Seleccionar la Opción “Tamaño de fuente”. - Cambiar el tamaño de fuente.	
Resultado esperado: El tamaño de la fuente es configurable.	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #27

Caso de Prueba Aceptación	
Número: 28	**Historia de usuario: 28**
Nombre: Mostrar unidad de medida.	
Descripción: Verificar que la opción de mostrar la unidad de medida es efectiva.	
Condiciones de Ejecución: El usuario debe comprobar que la unidad de medida puede mostrarse según desee.	

Entradas/ Pasos de Ejecución: - Seleccionar componente. - Abrir inspector de propiedades. - Marcar o desmarcar opción de mostrar unidad de medida.
Resultado esperado: La unidad de medida puede marcarse o desmarcarse.
Evaluación de la prueba: Prueba satisfactoria.

Prueba de Aceptación #28

Caso de Prueba Aceptación	
Número: 29	**Historia de usuario: 29**
Nombre: Configurar dimensión del flujómetro Emergente.	
Descripción: Comprobar que se puede configurar la dimensión del flujómetro Emergente.	
Condiciones de Ejecución: El usuario debe comprobar que el flujómetro puede ser configurado en ancho y alto.	
Entradas/ Pasos de Ejecución: - Seleccionar el componente. - Con el clic izquierdo seleccionar uno de los puntos donde puede redimensionarse para agrandarlo o achicarlo.	
Resultado esperado: El componente se redimensiona sin perder calidad	
Evaluación de la prueba: Prueba satisfactoria.	

Prueba de Aceptación #29

Caso de Prueba Aceptación

Número: 30	Historia de usuario: 30
Nombre: Ejecución de comando sobre las mediciones asociadas al componente.	
Descripción: Comprobar que en el componente se ejecuta el comando de escritura sobre la medición que representa el estado.	
Condiciones de Ejecución: El usuario debe permitir ejecutar comandos que se reciben a través de la medición analógica para representar el estado del componente	
Entradas/ Pasos de Ejecución: - Visualizar el componente en el ambiente de ejecución. - Clic derecho sobre el componente. - El componente despliega un menú contextual con los posibles estados a que puede ser cambiados el estado actual de la válvula. - El usuario es coge un ítem del menú y da clic, enviando un comando de escritura con el punto y el valor de escribir - Como resultado se cambia el color de la válvula o sale una ventana emergente indicando el error que se ha producido.	
Resultado esperado: Se cambió de color de la válvula con el color del estado que se mandó ejecutar.	
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.	

Prueba de Aceptación #30

Caso de Prueba Aceptación	
Número: 31	**Historia de usuario: 31**
Nombre: Representar en el componente el estado en el que se encuentra las mediciones asociadas.	

Descripción: Comprobar que el usuario vea a través del componente si se representa el estado de una de las mediciones asociadas a través de él.
Condiciones de Ejecución: El usuario debe comprobar que a través del componente se represente el estado de las mediciones asociadas a él.
Entradas/ Pasos de Ejecución: - Llega al visualizador una actualización de la medición. - La medición se actualiza. - Notifica a todos los asociados a ella. - El componente recibe la notificación. - El componente representa el nuevo estado/valor de la medición.
Resultado esperado: Representa correctamente el estado de la medición.
Evaluación de la prueba: Una vez realizada la primera iteración detectaron varias no conformidades, a las cuales se les da solución y se realiza otra iteración que arrojó una evaluación satisfactoria.

Prueba de Aceptación #31

ANEXO 3 DISEÑOS DE CLASES

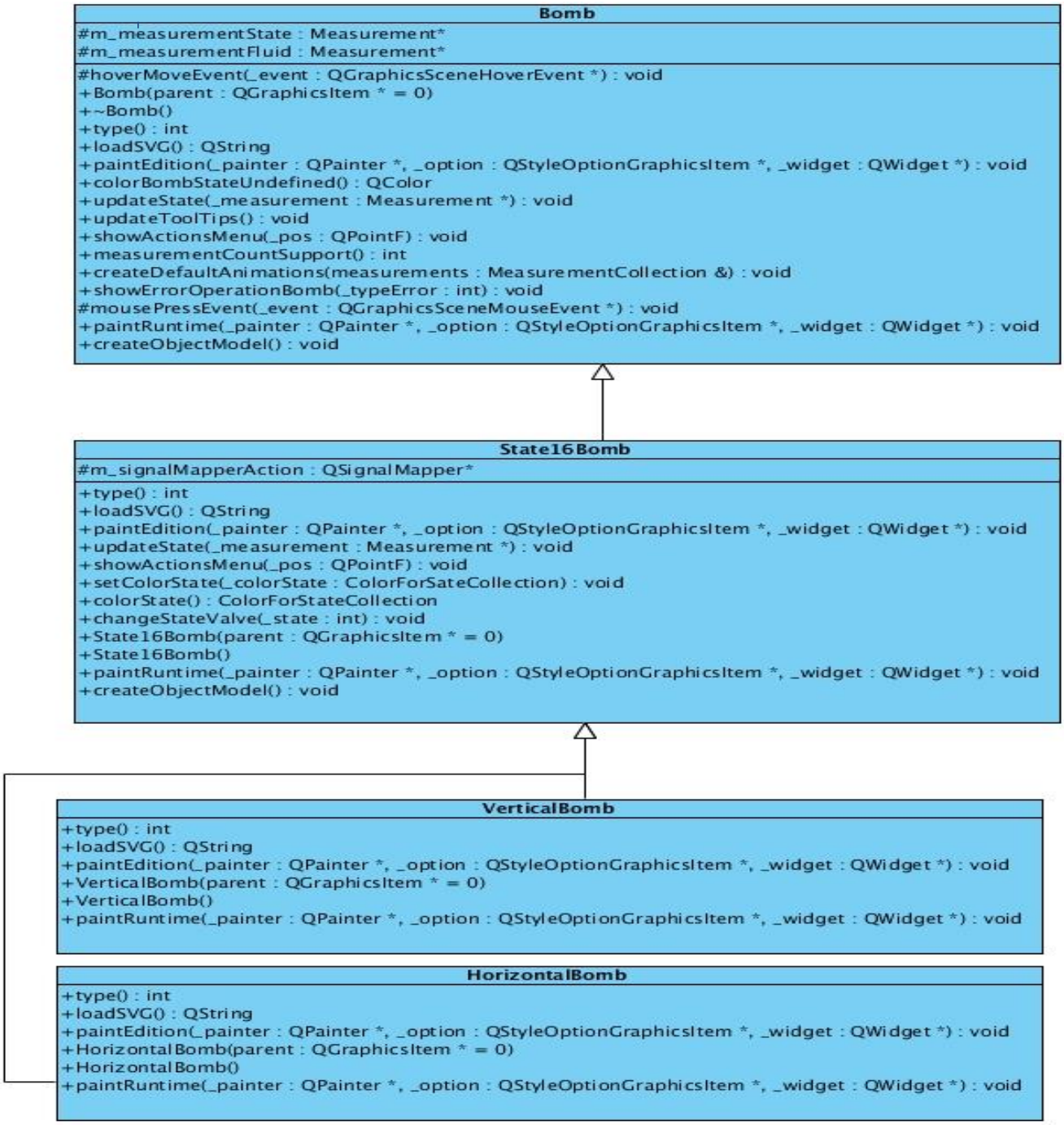


Fig16. Diseño de clase para la entidad Válvula y las clases que heredan de ella.

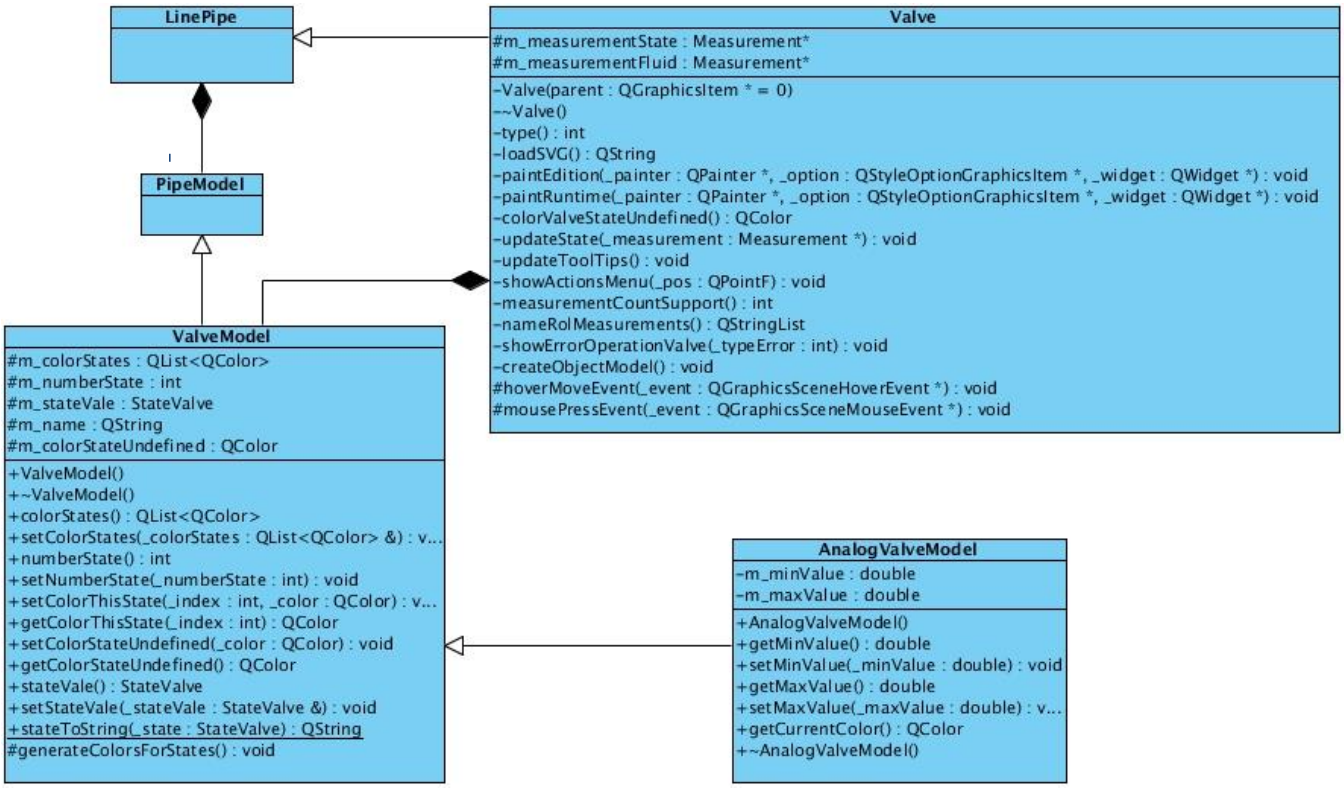


Fig17. Diseño de Clases para los modelos de la entidad válvula

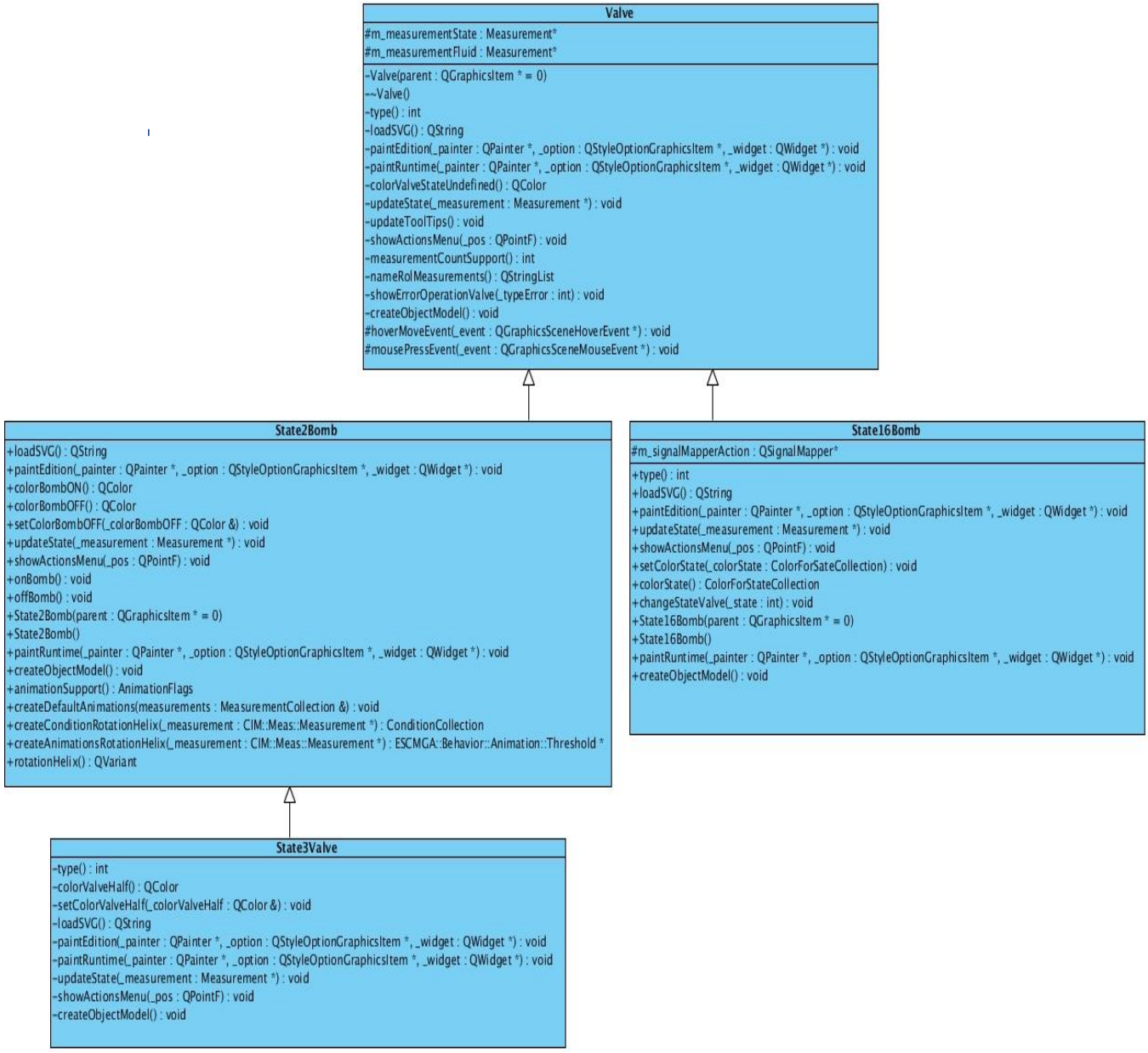


Fig18.Diseño de clases para las entidades de tipo Válvula

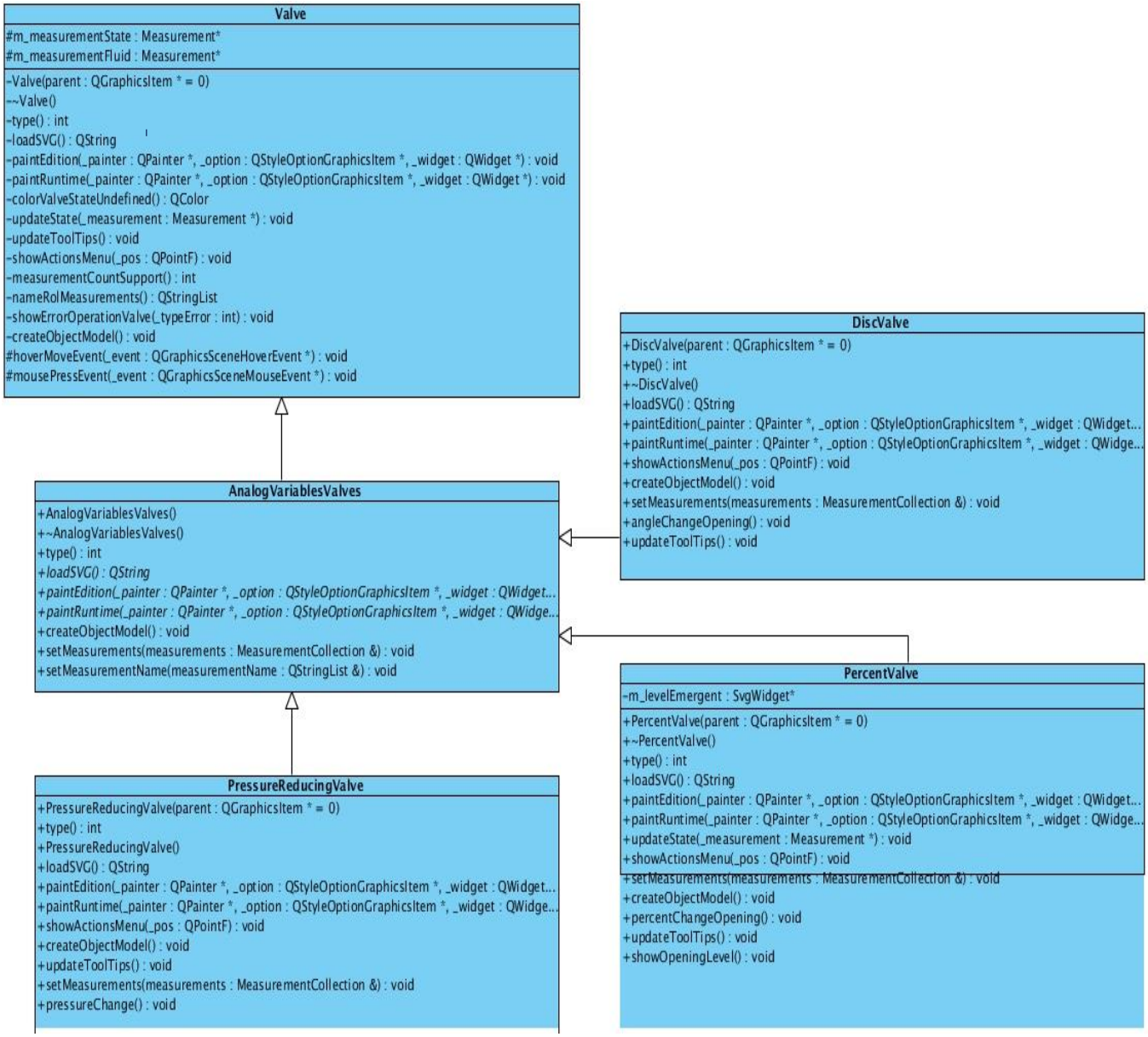


Fig19. Diseño de clases para las entidades que heredan de la entidad Válvula