DESARROLLO DE LOS COMPONENTES GRAFICOS CURVA CUADRATICA Y CUBICA DE BEZIER

Ing. Luis Andrés Valido Fajardo[1]

1. Universidad de Matanzas – Sede "Camilo Cienfuegos"- Departamento de Desarrollo de Recursos para Aprendizaje, Vía Blanca Km.3, Matanzas, Cuba. luis.valido@umcc.cu

Resumen

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). Unos de los módulos que componen el SCADA SAINUX es la Interfaz Hombre Máquina el cual constan de dos ambientes: el ambiente de configuración (AC) y el ambiente de visualización (AV). Tanto para el AC como para el AV, la Interfaz Hombre Máquina provee una biblioteca de componentes gráficos (CG) que permiten recrear de una manera lo más real posible el proceso de campos. Los CG constan entre sus limitantes el no poseer un componente grafico para la representación de curva o figuras curvilíneas. El presente trabajo tiene como objetivo principal el desarrollo de componentes gráficos para la representación de curvas y figuras curvilíneas en la Interfaz Hombre Máquina del SCADA SAINUX. En el trabajo se utilizaron métodos científicos como el analítico-sintético, histórico-lógico, modelación y observación. Además, se utilizó *Agile Unified Process* como metodología de desarrollo, lenguaje de modelado UML, la herramienta CASE *Visual Paradigm*, el lenguaje de programación C++ y como marco de desarrollo Qt. Como resultado se obtuvo dos componentes gráficos para la representación de curvas y figuras curvilíneas en la Interfaz Hombre Máquina del SCADA SAINUX.

***Palabras claves**: Componente gráfico, Curvas de Bézier, Interfaz Hombre Máquina, SCADA*

ISBN: 978-959-16-4235-6

Introducción

En la actualidad diversas industrias emplean para la automatización de sus procesos productivos los sistemas SCADA, acrónimo de *Supervisory Control and Data Acquisition* (Control, Supervisión y Adquisición de Datos). Se trata de una aplicación software especialmente diseñada para el control de producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos, autómatas programables, instrumentos inteligentes, etc.), controlando el proceso de forma automática desde la pantalla del ordenador. (Penin, 2012)

Una de las prestaciones de un SCADA es la visualización, mediante los gráficos del equipamiento actualizado para reflejar datos del campo, dentro de su estructura encontramos el módulo HMI, acrónimo de *Human Machine Interface* (Interfaz Hombre-Máquina), que no es más que la interfaz gráfica para la visualización del estado del proceso mediante objetos animados, gráficos, textos, listados, ventanas múltiples, entre otros. (Cobo, 2015)

La informatización de la sociedad cubana es una voluntad política del gobierno que está expresada en los Lineamientos de la Política Económica y Social, aprobados en el Sexto Congreso del Partido Comunista de Cuba (2011). En varios sectores de la industria y la economía cubana se necesita un sistema capaz de informatizar la supervisión y control de sus procesos. SAINUX (Sistema de Automatización Industrial basado en GNU/LINUX), es un sistema SCADA distribuido, orientado a componentes y en proceso de desarrollo por especialistas del Centro de Informática Industrial (CEDIN) de la Universidad de las Ciencias Informáticas (UCI).

El HMI del SAINUX constan de dos ambientes: el ambiente de configuración (AC), 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 ambiente de visualización (AV) es donde el operador puede supervisar y controlar la configuración realizada en el AC, 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 AC como para el AV el HMI provee una biblioteca de componentes gráficos (CG) que permiten recrear de una manera lo más real posible el proceso de campos. Estos gráficos pueden ser simples figuras geométricas como: línea, rectángulo, elipse, texto, polígono, polilínea; pero también muy complejas como los componentes de hardware utilizados en la industria que se desean supervisar como son: válvulas, tanques, generadores, motores de combustión.

ISBN: 978-959-16-4235-6

Los CG constan entre sus limitantes el no poseer un componente gráfico para la representación de curva o figuras curvilíneas lo cual acarrea como consecuencias negativas:

1. El pobre diseño de componentes y estructuras que participan en el proceso a monitorear.

2. La no modelación de componentes y estructuras que participan en el proceso a monitorear.

3. El empobrecimiento de la representación gráfica del proceso a monitorear.

Por todo lo anteriormente descrito se define como problema de investigación ¿Cómo contribuir al proceso de diseño, modelación y representación de los componentes gráficos en la Interfaz Hombre Máquina del sistema SCADA SAINUX? Por lo que se plantea como objetivo general el desarrollo de componentes gráficos para la representación de curvas y figuras curvilíneas.

Materiales y métodos o Metodología computacional

Conceptos asociados a la investigación

Curva de Bézier: Es un sistema que se desarrolló hacia los años 1960 para el trazado de dibujos técnicos, en el diseño aeronáutico y en el de automóviles. Su denominación es en honor a Pierre Bézier, quien ideó un método de descripción matemática de las curvas que se comenzó a utilizar con éxito en los programas de CAD[1]

La idea de definir geométricamente las formas no es demasiado compleja: un punto del plano puede definirse por coordenadas. Por ejemplo, un punto A tiene unas coordenadas (x1, y1) y a un punto B le corresponde (x2,y2). Para trazar una recta entre ambos basta con conocer su posición. Si en lugar de unir dos puntos con una recta se unen con una curva, surgen los elementos esenciales de una curva Bézier; los puntos se denominan puntos de anclaje o nodos. La forma de la curva se define por unos puntos invisibles en el dibujo, denominados puntos de control, manejadores o manecillas.

Las curvas de Bézier han sido ampliamente usadas en los gráficos generados por ordenador para modelado de curvas suaves[2]. Las transformaciones afines tales como traslaciones y

[1] El diseño asistido por computadoras (diseño asistido por ordenador en español), más conocido por sus siglas inglesas CAD (*computer-aided design*), es el uso de un amplio rango de herramientas computacionales que asisten a ingenieros, arquitectos y diseñadores.

[2] Se le llama curva suave a la curva que no posee puntos angulosos. Un ejemplo puede ser el círculo, la elipse, la parábola,

ISBN: 978-959-16-4235-6

rotaciones pueden ser aplicadas, con gran facilidad, a las curvas, aplicando las transformaciones respectivas sobre los puntos de control. (Grau, 2012)

HMI Human Machine Interface (HMI) ó Interfaz Hombre-Máquina: Es uno de los módulos de un SCADA, y es el encargado de presentar la información en forma de sinópticos gráficos. Como su nombre lo indica, es la interfaz que permite al operador interactuar con el sistema. Existen diferentes tipos de HMI, de los cuales se pueden reconocer claramente tres de ellos: Interfaces Gráficas de Usuario (GUI), Interfaces de Usuario basadas en la Web y Interfaces de Usuario para dispositivos móviles.

Existen varios tipos de interfaces que actualmente son utilizadas en menor escala, entre ellas están las basadas en líneas de comandos, las táctiles basadas en gestos, las multipantallas y las basadas en texto. (Shneiderman, 2010)

En el SCADA el módulo HMI estará dividido en dos sub-módulos, el ambiente de configuración y el ambiente de ejecución.

- **Ambiente de configuración:** El ambiente de configuración o editor gráfico, permite configurar los procesos, definir y gestionar las variables, editar los controladores, los comandos, las alarmas y variadas opciones adicionales. Detalladamente, el editor de configuración es el que permite a un mantenedor 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 sinópticos, almacenados en el ordenador en el SCADA o bien importados desde otras aplicaciones durante la configuración del paquete de software.

- **Ambiente de ejecución:** El ambiente de ejecución, es otro de los sub-módulos del módulo de HMI, también conocido como RunTime o Visualizador, se encarga de visualizar las animaciones y los objetos definidos en el editor, muestra lo que está ocurriendo en el campo en tiempo real, es el que envía los comandos a las estaciones remotas, quién recibe los valores de las variables, interactúa con la mayoría de los operadores pues se emplea para supervisar el proceso de manera directa.

SCADA SAINUX: El sistema SCADA SAINUX es un sistema distribuido, compuesto por diferentes subsistemas o módulos que trabajan de manera conjunta para llevar a cabo las tareas de supervisión, control y adquisición de datos. Cada uno de estos módulos es encargado de realizar una tarea específica:

Comunicaciones: Se encarga de la comunicación entre los diferentes módulos que forman parte del sistema. Implementa dos formas de comunicación, el modelo de comunicación asincrónica (Moore, 1992), el cual tiene como base el Servicio de Notificación de TAO,

ISBN: 978-959-16-4235-6

fundamentado en el paradigma publicación-subscripción y el modelo sincrónico (Hines, 2008) inherente al modelo cliente/servidor de la especificación utilizada (CORBA) (Gelbmann, 2002).

Configuración: El módulo de configuración está formado por varios componentes o elementos de la configuración del proyecto relacionado con seguridad, comunicación, con los dispositivos de campo, variables entre otros. Es el encargado de almacenar, persistir y suministrar la información base para el funcionamiento de los demás módulos del SCADA.

Adquisición o núcleo de procesamiento de datos: El núcleo de procesamiento de datos es el encargado del tratamiento y análisis en tiempo real de la información adquirida desde el campo.

Seguridad: Este módulo proporciona las funcionalidades necesarias para garantizar el trabajo autorizado a usuarios y módulos. Tiene implementada herramientas que protegen al sistema de ataques maliciosos o involuntarios por parte de personas o recursos, tales como fallas de energía y problemas de red.

Base Datos Histórico (BDH): Se encarga del almacenamiento y procesado ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos, para realizar posteriores análisis como diagnósticos o reportes.

Interfaz Hombre Máquina: Este módulo se encarga de representar en un ordenador, los procesos que ocurren en el campo en tiempo real. Muestra de forma gráfica los procesos operacionales, alarmas, variables, reportes y despliegues que permiten al operador el contacto directo con el sistema.

Soluciones de software analizadas

***Adobe Illustrator* (AI):** Es un editor de gráficos vectoriales en forma de taller de arte que trabaja sobre un tablero de dibujo, conocido como mesa de trabajo y está destinado a la creación artística de dibujo y pintura para ilustración (ilustración como rama del arte digital aplicado a la ilustración técnica o el diseño gráfico, entre otros). Es desarrollado y comercializado por *Adobe Systems*.(Johnson, 2012)

Macromedia *FreeHand* (FH): Es un programa informático de creación de imágenes mediante la técnica de gráficos vectoriales. Gracias a ella, el tamaño de las imágenes resultantes es escalable sin pérdida de calidad, lo que tiene aplicaciones en casi todos los ámbitos del diseño gráfico. (Gonzalez, 2003)

ISBN: 978-959-16-4235-6

CorelDRAW: Es una aplicación informática de diseño gráfico vectorial, es decir, que usa fórmulas matemáticas en su contenido. Ésta, a su vez, es la principal aplicación de la suite de programas CorelDRAW *Graphics Suite* ofrecida por la corporación Corel y que está diseñada para suplir múltiples necesidades, como el dibujo, la maquetación de páginas para impresión y/o la publicación web, todas incluidas en un mismo programa. (2001)

Inkscape: Es un editor de gráficos vectoriales en formato SVG, gratuito, libre y multiplataforma. Las características de SVG soportadas incluyen formas básicas, trayectorias, texto, canal alfa, transformaciones, gradientes, edición de nodos, exportación de SVG a PNG, agrupación de elementos, etc. (Cedric Gemy, 2007)

GIMP (*GNU Image Manipulation Program*): }Es un programa de edición de imágenes digitales en forma de mapa de bits, tanto dibujos como fotografías.Es un programa libre y gratuito. Forma parte del proyecto GNU y está disponible bajo la Licencia pública general de GNU y GNU Lesser General Public License[3] o Licencia Pública General para Bibliotecas de GNU), o simplemente por su acrónimo del inglés GNU LGPL, es una licencia de software creada por la *Free Software Foundation* que pretende garantizar la libertad de compartir y modificar el software cubierto por ella, asegurando que el software es libre para todos sus usuarios.} Es el programa de manipulación de gráficos disponible en más sistemas operativos (Unix, GNU/Linux, FreeBSD, Solaris, Microsoft Windows y Mac OS X, entre otros). (van Gumster, 2011)

Dia: Dia es una aplicación informática de propósito general para la creación de diagramas, desarrollada como parte del proyecto GNOME. Está concebido de forma modular, con diferentes paquetes de formas para diferentes necesidades. Dia está diseñado como un sustituto de la aplicación comercial Visio de Microsoft. Se puede utilizar para dibujar diferentes tipos de diagramas.

Métodos teóricos

Analítico-Sintético: Se utiliza para analizar teorías y elementos bibliográficos relacionados con la representación de curvas y figuras curvilíneas en ordenadores, permitiendo la extracción de los elementos más importantes que hacen viable el comienzo de la investigación.

Histórico-Lógico: Se utiliza para el estudio del fenómeno en cuestión y su evolución histórica, permitiendo así, profundizar los conocimientos sobre el tema.

Modelación: Permitió la elaboración de múltiples diagramas para un mejor entendimiento del problema y solución.

[3] La Licencia Pública General Reducida de GNU, o más conocida por su nombre en inglés GNU *Lesser General Public License* (antes GNU *Library General Public License*)

ISBN: 978-959-16-4235-6

Métodos empíricos

Observación: Se emplea para estudiar las características y comportamientos de las soluciones similares, permitiendo obtener información relevante sobre el funcionamiento de dichas soluciones.

Metodología de desarrollo de software: *Agile Unified Process* (AUP)

Con una buena metodología se pretende reducir costos y retrasos de proyectos, así como mejorar la calidad del software. *Agile Unified Process* (AUP) ó Proceso Unificado Ágil es una versión simplificada del Proceso Unificado de Rational (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP (Ambler, 2005)

Lenguaje de Modelado: UML

UML (*Unified Modeling Language*) es desde finales de 1997 un lenguaje de modelado visual que se utiliza para especificar, visualizar, construir y documentar artefactos de un sistema de software (Rumbaugh, 2007).

Es válido destacar que UML es un lenguaje de modelado, no un método o un proceso. UML constituye un lenguaje más expresivo, claro y uniforme que los anteriores definidos para el diseño Orientado a Objetos, no brinda el total éxito de los proyectos pero si mejora sustancialmente el desarrollo de los mismos al posibilitar una nueva y fuerte integración entre las herramientas, los procesos y los dominios.

Herramienta CASE: *Visual Paradigm*

Herramienta CASE (del inglés, *Computer Aided Software Engineering*) con licencia gratuita, que propicia un conjunto de ayudas para el desarrollo de programas informáticos dando soporte al modelado visual con UML (del inglés, *Unified Modeling Language*), desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación (Oscar, 2013).

Lenguaje de programación: C++

C++ es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al lenguaje de programación C mecanismos que permiten la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

Existen gran cantidad de bibliotecas implementadas que permiten la reutilización del código fuente, disminuyen el tiempo de desarrollo y aumenta la productividad. La

ISBN: 978-959-16-4235-6

utilización de plantillas también permite la reutilización del código y la programación genérica (Design, 2005).

Marco de trabajo: Qt

El marco de trabajo de desarrollo Qt es un conjunto de herramientas multiplataforma utilizada para construir aplicaciones en C++, Python con interfaces gráficas o no en dependencia de las necesidades del desarrollador. Está patentado bajo la licencia GPL 2 y LGPL. Entre las utilidades que brinda esta biblioteca se encuentran las de diseño de interfaces de usuario, dibujar gráficas 2D y 3D de alta calidad utilizadas en diversos ambientes como en la gestión de negocios, herramientas de monitorización entre otras (Blanchette, 2006).

Sistema Operativo

La distribución seleccionada es Debian, específicamente la versión 7.0 o Wheezy. Debian o Proyecto Debian es una comunidad conformada por desarrolladores y usuarios, que mantiene un sistema operativo GNU basado en software libre. El sistema se encuentra precompilado, empaquetado y en un formato deb para múltiples arquitecturas de computadoras y para varios núcleos (Hunger, 2001).

Requisitos funcionales

Los requisitos funcionales son funciones del sistema de software o sus componentes. Cada función se describe como un conjunto de entradas, comportamientos y salidas. Los requisitos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone, un sistema debe cumplir. Son complementados por los requisitos no funcionales[4], que se enfocan en cambio en el diseño o la implementación. Establecen los comportamientos del sistema (Sommerville, 2015).

Por lo anteriormente explicado se definen como requisitos funcionales para ambos componentes gráficos:

1. Diseñar y representar figuras con el uso de la curva de Bezier en sus variantes cuadrática y cúbica.

2. Visualizar de los puntos de anclaje y control del componente en el ambiente de configuración cuando este seleccionado.

3. Manipular de forma independiente de cada punto de anclaje y de control que conforma la curva.

[4] Un requisito no funcional o atributo de calidad es, en la ingeniería de sistemas y la ingeniería de software, un requisito que específica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requisitos funcionales.

ISBN: 978-959-16-4235-6

4. Modificar el ancho del trazo de la curva.
5. Modificar el color del trazo de la curva
6. Adicionar un nuevo punto de anclaje a la curva por uno de sus extremos
7. Eliminar uno de los puntos de anclaje extremos de la curva.
8. Unir los puntos extremos de la curva con una línea.
9. Pintar el área encerrada por la curva en caso de que los dos puntos extremos estén conectados.

Patrones de diseño

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. La utilización de patrones de diseño, permite ahorrar grandes cantidades de tiempo en la construcción de software. El software construido es más fácil de comprender, mantener y extender. Estos se pueden clasificar en: (Gamma, 2003)

Comportamiento: Los patrones de comportamiento están relacionados con los algoritmos y la asignación de responsabilidades entre los objetos. Son utilizados para organizar, manejar y combinar comportamientos.

Estructurales: Los patrones estructurales se ocupan de cómo las clases y objetos se combinan para formar grandes estructuras y proporcionar nuevas funcionalidades.

Creacionales: Los patrones creacionales abstraen el proceso de creación de instancias y ocultan los detalles de cómo los objetos son creados o inicializados.

Para la implementación de la solución expuesta en este trabajo se hará uso del patrón de diseño de tipo comportamiento Método Plantilla el cual define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura (Books, 2010).

Resultados y discusión

A continuación se muestra el diagrama de clases que refleja las entidades que componen la solución de este trabajo y una breve descripción de cada una de ellas.

ISBN: 978-959-16-4235-6

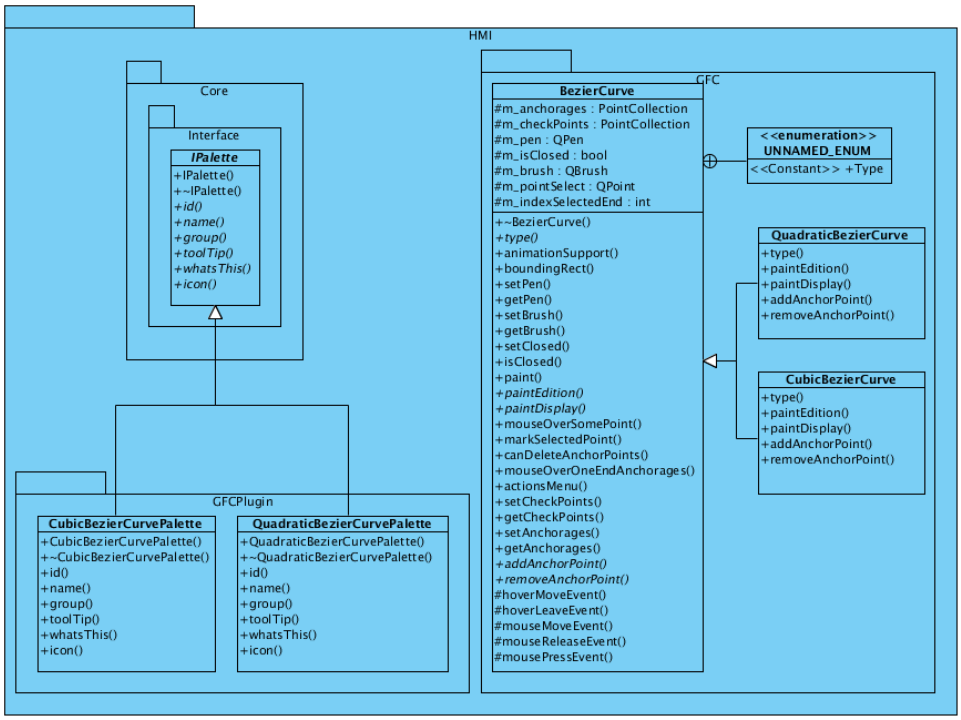

Figura 1. Diagrama de clases de la solución.

La solución propuesta en este trabajo se compone de dos partes, la primera las paletas de cada unos de los componentes las cuales representan a los componentes dentro del sistema y le permiten al usuario escoger dichos para ser utilizados dentro de los despliegues. Para el desarrollo de las paletas fueron implementadas dos entidades:

QuadraticBezierCurvePalette: Entidad encargada de representar el componente gráfico curva cuadrática de Bézier dentro de la paleta de componentes permitiendo visualmente que el usuario seleccione dicho componente con el mouse y lo arrastre hacia el despliegue donde desea utilizarlo.

CubicBezierCurvePalette: Entidad encargada de representar el componente gráfico curva cúbica de Bézier dentro de la paleta de componentes permitiendo visualmente que el usuario seleccione dicho componente con el mouse y lo arrastre hacia el despliegue donde desea utilizarlo.

Ambas entidades heredan de la interfaz abstracta *IPalette* de la cual deben heredar todas aquellas entidades que su función es representar algún componente gráfico dentro de la paleta de componentes obligando que dichas entidades reimplementen seis funcionalidades las cuales se encargan de devolver los datos del componente gráfico como son: su identificador, nombre, grupo al que pertenece, el ícono que debe visualizarse en la paleta, pista de identificación y ayuda.

ISBN: 978-959-16-4235-6

En el caso de los componentes desarrollados en la solución estarán ubicados en grupo de Básicos de la paleta de componentes del editor gráfico de SAINUX.

La otra parte de la solución la completan las entidades:

BezierCurve: Es la entidad base donde se aglomeran todas la funciones y atributos que son comunes para cualquier entidad encargada de representar una curva de Bezier independiente del grado de la curva que representa. Esta entidad presenta cinco funcionalidades virtuales[5] obligando que las clases que hereden de ella reimplemente estas funcionalidades producto que su implementación y lógica varía de acuerdo al grado de la curva.

QuadraticBezierCurve: Entidad que representa la curva de Bézier de grado 2. Hereda las funciones y atributos de la clase ***BezierCurve*** reimplementado las cuatro funciones virtuales pertenecientes a la clase base de las Curvas de Bézier.

CubicBezierCurve: Su función u objetivo es similar a la entidad ***QuadraticBezierCurve*** pero en su caso específico es la representación de la curva de Bézier de grado 3.

A continuación se muestra gráficamente dentro del ambiente de configuración el lugar donde se muestra las paletas de componentes desde donde el usuario o mantenedor podrá seleccionar las curvas de Bézier para su utilización dentro del despliegue.

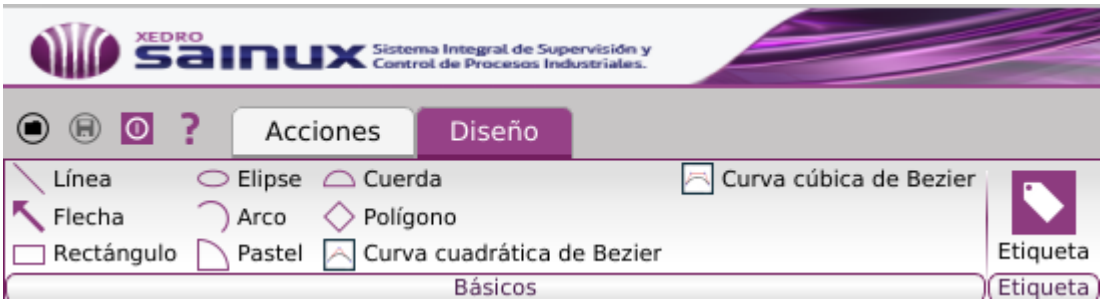

Figura 2. Acceso a los componentes desde la paleta de diseño.

En la siguiente imagen se muestra ambos componentes dentro de un despliegue en el ambiente de configuración. Mostrándose en la izquierda de la imagen el componente curva cúbica de Bézier mientras a la derecha se observa la curva cuadrática de Bézier.

[5] En programación orientada a objetos (POO), una función virtual o método virtual es una función cuyo comportamiento, al ser declarado virtual, es determinado por la definición de una función con la misma cabecera en alguna de sus subclases. Este concepto es una parte muy importante del polimorfismo en la POO.

ISBN: 978-959-16-4235-6

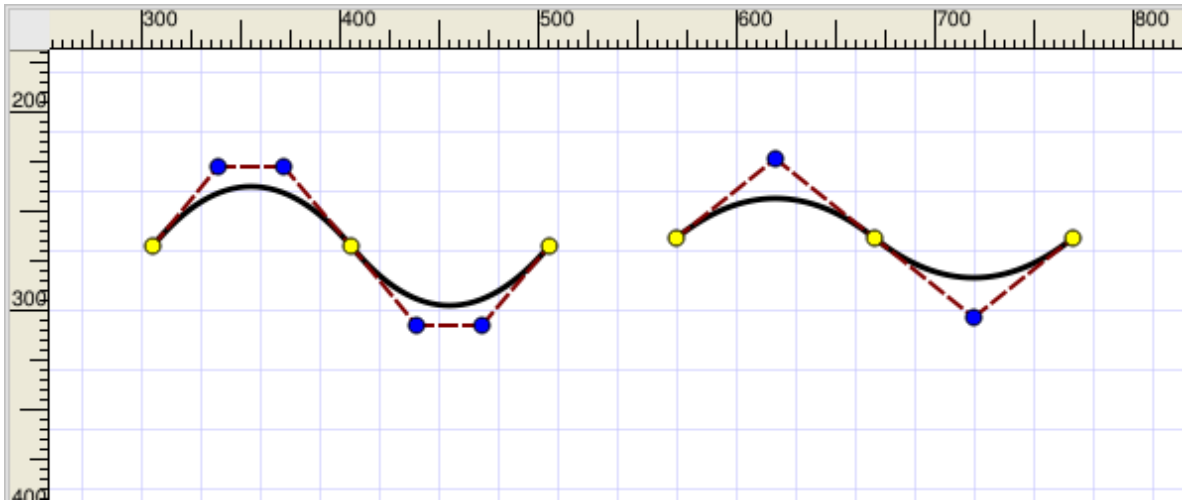

Figura 3. Visualización de los componentes dentro de un despliegue en ambiente de configuración.

Para la configuración de los componentes se puede hacer con el ratón sobre el componente cuando se desea modificar la posición de un punto de anclaje o de control o sencillamente eliminar o adicionar un nuevo punto de anclaje a la curva. El resto de la propiedades configurables de ambos componentes se realiza a través del inspector de propiedades del Editor desde donde el operador o usuario podrá definir los valores para el resto de las propiedades que definen al componente. A continuación se muestra el inspector de propiedades visualizando las propiedades de los componentes.

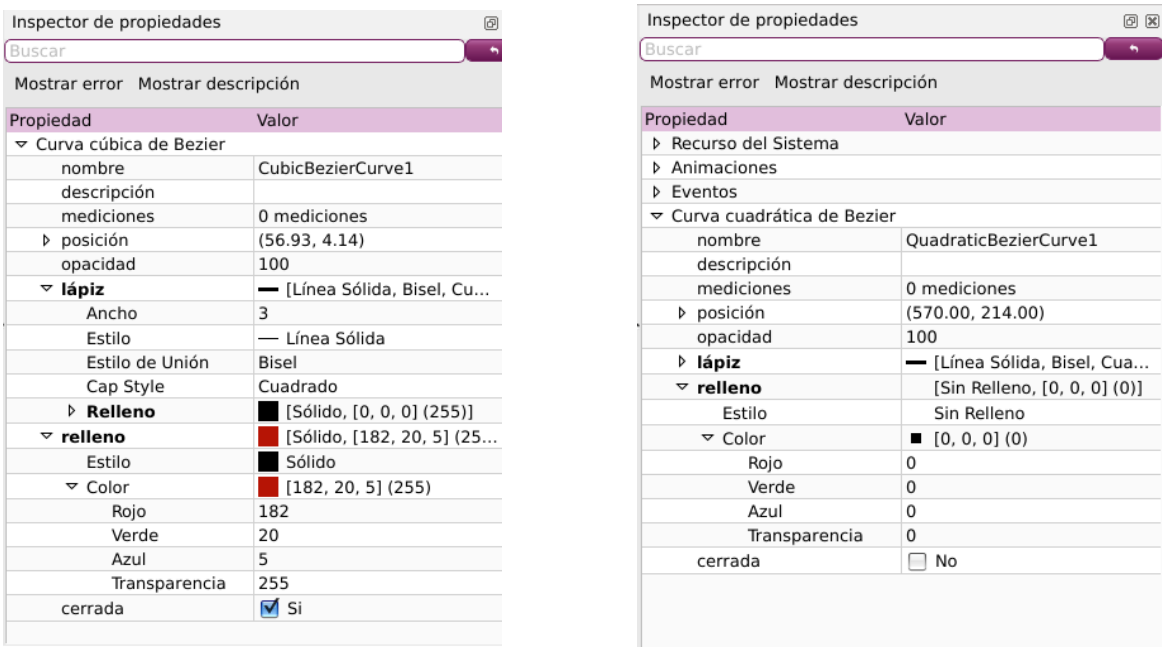

Figura 4 . Inspector de propiedades visualizando las propiedades configurables del componente curva cúbica y cuadrática de Bézier.

ISBN: 978-959-16-4235-6

Como se ha evidenciado los componentes gráficos desarrollados con la investigación realizada son capaces de solventar el problema que inicialmente se planteo en este trabajo, además los componentes cumple con todos los requisitos funcionales que le fueron definidos. Pero es el resultado positivo de la investigación motiva o da pie a dos ideas o sugerencias para posteriores investigaciones teniendo como inicio o base los resultados obtenidos en este trabajo. Dichas ideas o sugerencias son:

Inserción de puntos de anclaje: Como parte de las funcionalidades u operaciones que se pudiera realizar sobre los componentes desarrollados en el ambiente de configuración se pudiera incluir la de insertar nuevos puntos de anclaje con sus respectivos puntos de control asociados en dependencia del grado de la curva. Esto podría contribuir en la configuración o edición de las figuras que se realizan con el componente. La idea tiene su fundamento en un subconjunto del grupo de las soluciones analizadas en la investigación las cuales ofrecen esta funcionalidad.

Parametrización de la curva: Se parte de la idea de representar las curvas en un plano a partir del valor que tenga las coordenadas tanto en el eje de las abscisas como en el de las ordenadas los puntos que integran la curva. Dichos valores podrían asociarse a constantes o algunas de las variables analógicas del proceso que es supervisado y controlado desde el SCADA SAINUX.

Conclusiones

La realización del presente trabajo contribuyó al proceso de diseño, modelación y representación de los componentes gráficos en la Interfaz Hombre Máquina del sistema SCADA SAINUX con el desarrollo de los componentes de curvas cuadrática y cúbica de Bézier permitiendo la representación de curvas y figuras curvilíneas dentro de los despliegues. La solución descrita permite el diseño y modelación de componentes y estructuras que participan en el proceso a monitorear lográndose un mayor realismo en las representaciones gráficas creadas en los despliegues. El análisis y discusión de la solución obtenida propone seguir profundizando en los resultados obtenidos trabajando sobre dos aristas o ideas, la primera en la ampliación de las funcionalidades que brindan los componentes para su edición y la segunda en la parametrización de las curvas que representan los componentes. Destacar que en el desarrollo del trabajo fueron utilizadas tecnologías y herramientas libres.

ISBN: 978-959-16-4235-6

Bibliografía

Ambler, Scott. 2005. The Agile Unified Process (AUP). 2005.

Blanchette, J. and Summerfield, M. 2006. *C++ GUI Programming with Qt 4.* 2006.

Books, LLC. 2010. *{Method: Factory Method Pattern, Template Method Pattern, Constructor,Multiple Dispatch, Copy Constructor, Method.* 2010.

Cedric Gemy, K.W. 2007. *Inkscape User Manual - Aug-2007.* s.l. : Martin Iturbide, 2007.

Cobo, Raul. 2015. *El ABC de la Automatizacion HMI.* 2015.

2001. *Curso de Corel Draw!: manual del alumno.* 2001.

Design, D. and Stroustrup, B. 2005. *The C++ Programmig Language (4th Edition) By Bjarne Stroustrup: C++.* 2005.

Gamma, E. and Helm, R. and Johnson, R. and Vlissides, J. 2003. *Design Patterns: Elements of Reusable Object-Oriented Software with Applying Uml and Patterns:An Introduction to Object-Oriented Analysis and Design and the Unified Process.* 2003.

Gelbmann, R. 2002. *Evaluation and Comparison of CORBA (Object Request Broker) Implementations.* 2002.

Gonzalez, F.P. 2003. *Guia de campo de Macromedia: Freehand MX.* 2003.

Hines, J. and University of Pittsburgh. 2008. *Highly Synchronous Communication - Characterization, Modeling and.* s.l. : University of Pittsburgh, 2008.

Hunger, S. 2001. *Debian GNU/Linux Bible.* 2001.

Johnson, S. 2012. *Adobe Illustrator CS6 on Demand.* s.l. : Pearson Education, 2012.

2011. Lineamientos de la política económica y social del partido y la revolución. 2011.

Moore, J.S. 1992. *A formal model of asynchronous communication and its use in mechanically.* s.l. : National Aeronautics and Space Administration, Office of Management,, 1992.

Oscar, S. 2013. *Visual Paradigm for Uml.* 2013.

Penin, A.R. 2012. *Sistemas SCADA.* s.l. : Marcombo, 2012.

ISBN: 978-959-16-4235-6

Rumbaugh, J. and Booch, G. and Jacobson, I. 2007. *El Lenguaje unificado de modelado: manual de referencia.* 2007.

Sommerville, I. 2015. *Software Engineering.* 2015.

van Gumster, J. and Shimonski, R. 2011. *GIMP Bible.* s.l. : Wiley, 2011.

ISBN: 978-959-16-4235-6