UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Emulador de sistemas dinámicos para la práctica y enseñanza de control automático industrial

T E S I S

Que para obtener el título de

Ingeniero Eléctrico - Electrónico

P R E S E N T A

Daniel Roberto Mendoza Barrera

DIRECTOR DE TESIS

Dr. Hoover Mujica Ortega

Ciudad Universitaria, Cd. Mx., noviembre 2017

Jurado asignado

Presidente:	M.I. Yukihiro Minami Koyama
Secretario:	Dr. Paul Rolando Maya Ortiz
Vocal:	Dr. Hoover Mujica Ortega
1^{er} suplente:	Dr. Gerardo René Espinosa Pérez
2^{do} suplente:	Mtra. Gloria Correa Palacios

Ciudad Universitaria, Laboratorio de Control, Laboratorio de Automatización.

Ciudad de México

Asesor de tesis

Dr. Hoover Mujica Ortega

Agradecimientos

A mi familia por su apoyo, su cariño y su confianza.

A mi Universidad por haberme brindado varias de las más grandes satisfacciones de mi vida y ser mi segunda casa por todos estos años.

A mi Facultad, por haberme recibido y haber sacado lo mejor de mí, pero sobre todo por su invaluable formación.

A los integrantes del Laboratorio de Control y al Dr. Gerardo Espinosa por su apoyo durante el desarrollo de esta tesis desde su inicio.

En especial a mi asesor de tesis y amigo, Hoover Mujica Ortega por brindarme la confianza, los medios y la paciencia para desarrollar este proyecto.

Dedicatoria

Siento que es una obligación para mí tratar de dedicar esta tesis con algo más que las típicas palabras de agradecimiento, pero creo que las palabras menos elegantes siempre son las más sinceras, y no me queda más que dedicar esta tesis con todo mi cariño a mis padres, por ser el pilar sobre el que reposa lo que es y será Daniel Mendoza Barrera y que sin ellos nada de esto sería posible, a mi hermano David por su apoyo incondicional, a mi "abue"por su admiración y quien en todo momento me ha demostrado su orgullo, a mis tías Norma, Carmen, Ivonne, Sandy y mi tío Memo, quienes han estado ahí desde el primer día que entre a la escuela hasta hoy, pero muy en especial a todos mis primos Alejandro, Emiliano, Erick, Santiago, Sebastián a Memo y Ernesto quienes son una parte muy importante en mi vida. Dedico también esta tesis, a mi amigo Hoover Mujica por el apoyo incondicional, la paciencia y sobre todo por su invaluable vocación a la enseñanza que me ha guiado dentro y fuera de los salones de clase, a mis compañeros y amigos de carrera con quienes sufrí y compartí todos estos años y a quienes me haya faltado nombrar, una disculpa, pero es poco el espacio y la gratitud es mucha. A mi abuelo Coco, y a Israel, esto también es para ustedes. Por último, esto es para tí Daniel que te tardaste pero al fin quedó.

Resumen

En el presente trabajo, se describe el desarrollo de una herramienta computacional capaz de emular procesos industriales en tiempo real, a partir de la solución numérica de un conjunto de ecuaciones diferenciales ordinarias que describen a un determinado sistema. Este desarrollo pretende dotar al Laboratorio de Automatización de una plataforma efectiva de capacitación y entrenamiento para profesionales en ingeniería bajo un ambiente controlado, lo que permitirá alcanzar objetivos educacionales más elevados en el proceso de enseñanza-aprendizaje del control automático.

Es ampliamente reconocido en la comunidad de automatización y control, que el hecho de no disponer de la infraestructura necesaria para la formación y capacitación de profesionales puede complicar el proceso enseñanza-aprendizaje, evitando que el estudiante adquiera las competencias necesarias para el desarrollo e implementación de sistemas de control en la industria.

En relación con lo anterior, se puede considerar que idealmente la enseñanza de automatización se debería realizar haciendo uso de procesos industriales físicos, funcionales e instalados para la práctica de los estudiantes. Sin embargo, esto puede resultar complicado en vista de que son equipos altamente costosos de adquirir y mantener, además no estarían disponibles para la cantidad de estudiantes que pueden requerir el equipo durante la sesión de práctica, además existen diferentes cuestiones que complican aún más la instalación de un proceso industrial en una institución académica, como el factor económico que es una buena parte del impedimento ya que no solo se realiza un gasto inicial, también requiere gastos periódicos por diferentes conceptos. Por otro lado, podrían generarse riesgos a la salud, emisiones contaminantes y el uso de recursos extraordinarios, por lo que la instalación física de un sistema puede volverse más un problema que una solución.

Por lo tanto, la motivación principal de este trabajo es incrementar el nivel formativo del estudiante por medio del uso de una herramienta de software que sirva para evaluar las estrategias de control propuestas, evitando de esta

forma obligar al estudiante a tener un periodo de adaptación abrupto en el campo laboral, que no es lo ideal debido a los tiempos, recursos y riesgos que se presentan en ese ambiente.

Con el fin de lograr esta meta, en este trabajo se desarrolló una herramienta de software que emula un proceso industrial y sirve como apoyo en la sesión de clase, cubriendo la necesidad de requerir de un proceso físico y acercando a los estudiantes los equipos y sistemas que se encuentran en la industria.

El desarrollo del proyecto se llevó a cabo en diferentes etapas, se partió utilizando la técnica de *Bechmark* para determinar, en primer lugar si existen herramientas similares a esta, después observar que ventajas y desventajas tienen en función de los objetivos principales propuestos, para después atacar las debilidades e implementar las oportunidades de mejora identificadas. Este proceso se fundamentó en las necesidades reales que el estudiante presenta durante el proceso de aprendizaje.

Como primer proceso industrial a emular, se eligió un tanque atmosférico alimentado por una bomba centrífuga que a la descarga tiene una válvula tipo cuchilla con pilotaje electro-neumático. El desarrollo del software se enfocó en garantizar la correcta emulación del proceso, la cual no podía ser una secuencia de pasos pre-establecida, como los procesos que se encuentran simulados en algunas herramientas conocidas, en cambio se proveyó al software con un motor de resolución de sistemas de ecuaciones diferenciales ordinarias con capacidad de replicar cualquier sistema dinámico, con lo cual se obtiene una emulación de procesos más complejos y de comportamientos más naturales, lo cual ofrece un amplio abanico de soluciones tal como los hay en la industria, utilizando métodos numéricos para la solución de ecuaciones diferenciales ordinarias en tiempo real.

Al proveer el software con la capacidad de intercambiar datos con cualquier controlador industrial mediante un protocolo de comunicación estándar, se garantiza la inter-operatividad de este producto. Con el fin de verificar la efectividad de la transferencia de información, se realizaron pruebas de funcionamiento enlazando el proceso emulado en la aplicación de software y diversos controladores de automatismos programables ampliamente utilizados en el ambiente profesional. Los resultados de las pruebas mostraron un comportamiento satisfactorio, lo cual dio pie al desarrollo de otras funcionalidades importantes como la generación de gráficos de tendencia y la sincronización del comportamiento dinámico con la animación, las cuales están íntimamente relacionadas con la obtención de las trayectorias de la dinámica del sistema obtenida por el motor de solución.

Como última etapa del desarrollo, se realizó la implementación de herramientas adicionales para la generación de reportes relacionados con la emulación del proceso industrial; entre ellas se puede mencionar la edición y exportación de gráficos de tendencias, alertas acústicas, edición de parámetros de comunicación y la posibilidad de interacción entre el estudiante y el profesor a través de la conexión de red local.

La herramienta desarrollada representa un apoyo significativo al proceso de enseñanza y se busca integrarlo dentro del plan de estudios donde se involucren asignaturas relacionadas con la formación en ingeniería de control automático. Este desarrollo representa la plataforma base de un sistema de emulación de múltiples procesos industriales, quedando como trabajo futuro la inclusión de nuevos sistemas dinámicos y otras funcionalidades con el fin de mejorar sus aportaciones didácticas.

Contenido

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Para la formación de un profesional en ingeniería de control, se requiere no solo el estudio de los conceptos y la descripción analítica de los elementos que componen a los sistemas de control, sino también es conveniente que este conocimiento se complemente con la parte experimental, independientemente de cual sea el perfil que el estudiante desee seguir en su carrera profesional; ya sea la investigación, integración de sistemas, aplicación industrial o la docencia.

En tal sentido, es importante disponer de ciertas metodologías de enseñanza con el fin de que el estudiante desarrolle las habilidades necesarias, para llegar a niveles superiores de entendimiento que, según la taxonomía de Bloom, se obtienen de manera sistemática y secuencial; ya que un nivel cognitivo nuevo solo se puede conseguir si se han adquirido los anteriores [Goodhew, 2010].

Con base en esto, las técnicas de enseñanza en la ingeniería pueden seguir el mismo esquema pues la adquisición de conocimientos aplicables a esta deben seguir el mismo patrón, primero en el nivel de reconocimiento de un concepto nuevo, donde el estudiante sea capaz de describir, identificar, listar y relacionar conceptos de ingeniería; después llevando a cabo la comprensión, la aplicación, el análisis, la síntesis y por último la evaluación; donde ya se tiene la preparación suficiente como para argumentar, escoger, decidir, categorizar y seleccionar, no solo los conceptos aplicables durante el estudio, sino también pueda extrapolar esta categorización de conocimientos a la planeación de un proyecto de ingeniería.

También es importante identificar cómo se obtienen los niveles cognitivos, donde el primer acercamiento puede ser denominado como el umbral de conocimiento pues es el primer contacto con el nuevo concepto, después de esto se identifican los resultados de dicho acercamiento [Goodhew, 2010], es decir no solo se puede evaluar la obtención de este, también es posible probar hasta la forma en la que es capaz de aplicarse.

Existen diferentes métodos que son comúnmente empleados en la enseñanza de la ingeniería, que van desde la explicación cara a cara de un procedimiento establecido, hasta las metodologías más relevantes para las cuales se desarrolla en gran medida este trabajo, la primera de ellas es la conocida como aprendizaje basado en problemas (PBL Problem Based Learning por sus siglas en inglés), la cual otorga al estudiante las herramientas para resolver un problema y no definiendo una serie de pasos específicos para su resolución, esto lo obliga a desarrollar habilidades prácticas y analíticas, las cuales se evalúan mediante esta metodología con el fin de fortalecer en el estudiante los niveles cognitivos más altos por medio de la práctica y el enfoque desde diversos escenarios de un problema, alejándolo de las soluciones secuenciales y pre-establecidas, lo cual se convierte en una constante en la aplicación de soluciones de ingeniería.

Otro tipo de método es el denominado como el aprendizaje basado en proyectos (PjBL por sus siglas en inglés Project Based Learning), la cual tiene el propósito de aumentar los niveles cognitivos por medio de la realización de un proyecto, según lo reportado en [Prince y Felder, 2006]; son actividades que incluyen distintas tareas con el fin de entregar un producto terminado, siendo este proceso una práctica común dentro de la industria, pues esto lleva al estudiante a realizar no solo el estudio formal de un concepto, también lo conduce a ir más allá del objetivo previamente definido, como realizar investigaciones y reportar los resultados.

También existe una tendencia que está basada en las nuevas tecnologías, denominado E-Learning, el cual implica el uso de una computadora para acceder al material de aprendizaje, esto no solo se limita a la digitalización de material bibliográfico, también incluye el uso de herramientas virtuales como apoyo a la educación [Laurillard, 2002], esto se incluye dentro de la metodología denominada aprendizaje mejorado por la tecnología TEL (Technology Enhanced Learning), la cual contempla actividades no solo en papel, también considera las actividades virtualizadas como las simulaciones como principal fuente de conceptualización.

Basado en las anteriores metodologías, se presenta el siguiente desarrollo, el cual no solo aumenta la posibilidad de obtener los niveles de conocimiento superiores descritos por la taxonomía de Bloom, sino que está fundamentado en las metodologías del aprendizaje, enfocadas directamente a la educación de la ingeniería, en particular del control automático industrial, y puede considerarse una herramienta de apoyo, la cual abre una serie de oportunidades para desarrollar actividades bajo cualquiera de los procedimientos mencionados hasta ahora.

La aplicación está destinada a la formación, en temas de automatización, actividad que por lo general requiere equipos y recursos difíciles de mantener, así pues esta puede ser

utilizada como un medio de aprendizaje que ayude al estudiante a llegar a niveles cognitivos más elevados que los que tendría sin tener a su disposición una planta industrial para desarrollar sus habilidades, por lo cual se presenta el siguiente desarrollo que tiene como fin, la implementación de un emulador de procesos industriales para la práctica y enseñanza del control automático aplicado a este ambiente.

1.1. Motivación

En la actualidad, es muy común el uso de Controladores de Automatismos Programables (Programmable Automation Controller o PAC por sus siglas en inglés) para la operación de procesos industriales. Por esta razón, es muy importante que los profesionales en ingeniería cuenten con el conocimiento y destreza suficiente, para implementar soluciones adecuadas a los problemas emergentes relacionados con la automatización industrial.

Sin embargo, una formación integral en este campo se vuelve especialmente complicada de lograr, si no se cuenta con la infraestructura necesaria para replicar, lo más fielmente posible, las condiciones en las que se encuentran los procesos en la industria, lo que hace que el estudiante no comprenda completamente la importancia, el alcance y el funcionamiento de estos sistemas.

Por lo tanto, este trabajo está motivado por el deseo de incrementar significativamente el nivel formativo y cognitivo de los profesionales en ingeniería que interactúan con sistemas de control distribuido, controladores industriales programables y supervisión e integración SCADA (*Supervisory control and data acquisition* cuya traducción sería supervisión, control y adquisición de datos). En tal sentido, se busca que los estudiantes no solo sean capaces de asociar e identificar los problemas de automatización industrial, asimilar los fenómenos físicos asociados a los elementos que integran los procesos industriales y describir las especificaciones de diseño requerido, sino también, que tengan las competencias y habilidades para formular y desarrollar estrategias de control industrial, resolver problemas de automatización tomando en cuenta la instrumentación disponible actualmente, evaluar y analizar el comportamiento del sistema apreciando los efectos de la propuesta de control implementada en tiempo real y lograr así mejorar el rendimiento del proceso.

Considerando lo anterior, es fundamental que los estudiantes alcancen estos atributos durante su formación, de lo contrario se verán forzados a complementar algunos aspectos de su aprendizaje en el campo laboral, lo cual no es lo más recomendable sobre todo si se encuentran involucrados en el control y automatización de un proceso industrial crítico.

Adicionalmente, cabe mencionar que existe un gran interés por parte de las instituciones educativas y también por las empresas dedicadas a desarrollar equipo didáctico, en fortalecer la formación de profesionales en automatización industrial. En años recientes, se han conducido diversas investigaciones relacionadas con encontrar mecanismos que ayuden a mejorar el proceso enseñanza-aprendizaje, sin embargo, es el sector privado quien ha logrado un mayor impacto en el desarrollo de herramientas didácticas como simuladores de procesos, plataformas de experimentación y equipo para evaluar la respuesta de transductores y actuadores.

1.2. Antecedentes

En vista del renovado interés que existe, en las instituciones educativas, por asegurar la calidad formativa de los profesionales en ingeniería de control automático, es cada vez más frecuente encontrarse con el uso de diversos desarrollos didácticos fundamentados en las tecnologías de la información y comunicación (TIC); estos pueden significar una gran diferencia en las distintas etapas de la implementación de los sistemas de control, mismas que van desde la planeación, ingeniería de detalle hasta la operación.

Existen diferentes alternativas por las que se puede optar dependiendo puntualmente de la necesidad específica que se tenga, en primera instancia se busca la compra de un sistema real para la práctica y enseñanza de control automático, sin embargo los desarrollos basados en la simulación computacional para propósitos formativos, educacionales y de planeación se han convertido en una de las opciones más convenientes.

En tal sentido, algunos centros de desarrollo e investigación se han dado a la tarea de desarrollar aplicaciones para cubrir estas necesidades, dichos esfuerzos han sido encaminados principalmente a acercar a los estudiantes a los procesos industriales; algunos únicamente mostrando de manera prescriptiva su comportamiento y otros desarrollos más elaborados y específicos describen en mayor detalle la dinámica del sistema con fines de enseñanza del control automático industrial.

En la literatura especializada, se encuentran reportados desarrollos como el presentado por [D'Arthenay, 2015], donde se propone una aplicación en operación *Hardware-in-the-loop* (HIL) en la cual, como el nombre de la técnica indica, requiere de electrónica auxiliar dentro del lazo de flujo de datos, con el fin de realizar la adquisición de señales por medio de una tarjeta desarrollada por la empresa National Instruments, para así poder acoplar los módulos de entrada y salida de un controlador industrial real (Siemens S7-300) con variables provenientes de una simulación de procesos continuos y discretos desarrollada en Easy Java Simulations (EJS) dentro de una computadora con LabView. Esta propuesta,

requiere recursos adicionales a los de una solución de control real pues en la práctica profesional no son necesarios los elementos intermedios que este desarrollo necesita y además, incorpora herramientas accesorias que no forman parte del estándar de lenguajes de programación para controladores industriales descrito en el IEC61131-3, que es el conjunto de normas e informes técnicos publicado por la Comisión de Electrotécnia Internacional para estandarizar autómatas programables. [John y Tiegelkamp, 2010].

Adicionalmente, se puede encontrar diversas aplicaciones didácticas, las cuales están destinadas al acercamiento del estudiante con la dinámica de los procesos industriales y al control de sistemas, muchas de ellas están destinadas a la visualización de los procesos y su entendimiento. En este sentido la aplicación descrita en [Sánchez*, et al.*, 2002] realiza una simulación de procesos industriales complejos, los cuales pueden modificarse por medio de la introducción de diversos parámetros, sin embargo, esta no está destinada a la capacitación o la enseñanza del control automático industrial.

Por otro lado, la aplicación propuesta en [Vargas*, et al.*, 2014] expone el desarrollo de un simulador de un motor de CD para el control de velocidad; éste además de buscar prescindir de un sistema físico, tiene la posibilidad de implementar el control de velocidad por medio de un motor real, lo cual, hace de esta una herramienta versátil para la enseñanza, puesto que da al estudiante la posibilidad de comprobar el conocimiento adquirido por medio de la implementación real, sin embargo en dicho trabajo no propone sistemas industriales.

Ahora bien, en cuanto a la aplicación de software para la educación específica en control automático industrial, en [Sánchez Del Pozo Fernández, 2012] se presenta el desarrollo de un entorno 3D configurable, el cual usa una conexión por medio de OPC (*OLE for Process Control*, por sus siglas en inglés), para comunicarse con un controlador real para interactuar con el proceso, el cual se lleva a cabo con el desarrollo del modelado de los diferentes elementos dinámicos para lograr una animación realista, utilizando estaciones de trabajo real lo cual, como se ha mencionado, es una parte esencial de la formación de profesionales en ingeniería.

En vista de esto, los beneficios económicos y técnicos que puede generar el hecho de contar con una aplicación que pueda representar un sistema o un proceso industrial para evaluar soluciones de control, existen diferentes empresas que se han dedicado a la creación de software enfocado a la planeación de sistemas de ingeniería, específicamente con el fin de verificar las condiciones de operación durante las primeras etapas de diseño de un proceso.

Por ejemplo, existen aplicaciones como Flexsim y Simul8 las cuales son sistemas de simulación destinados al análisis de productividad, diseño, optimización y control de procesos

descritos en [Beaverstock*, et al.*, 2011] y [Concannon*, et al.*, 2007] respectivamente, que le hacen posible al usuario conocer el comportamiento de un sistema en etapa de diseño, a la vez permite una configuración completa y personalizada del proceso a evaluar, por lo que está encaminada a verificar la viabilidad económica y de operación de un sistema.

A su vez la iniciativa privada también se ha enfocado al desarrollo de aplicaciones orientadas a la educación y al entrenamiento de estudiantes y personal, los cuales tienen características específicas que permiten considerarlas para realizar un *Benchmark* y así poder definir las funcionalidades necesarias para facilitar el desarrollo de las competencias necesarias.

LogixPro, desarrollado por la empresa TheLearningPit destinado al aprendizaje de la programación de controladores Allen-Bradley de pago que utilizan el software RSLogix 500 [Kamel y Kamel, 2016], el cual cuenta con una interfaz de programación que replica el ambiente de programación por lo que no cuenta con todas las funcionalidades de un controlador real, y a pesar de que este cuenta varios sistemas para practicar la programación del controlador, el desarrollo se ha estancado eliminando la posibilidad de aumentar las posibilidades de mejorar.

VirPLC es un software de la empresa IES Palamos, el cual es fácil de utilizar y demuestra su enfoque didáctico [Ferrer Rojas, 2017] al guiar al estudiante paso a paso a montar una solución de control, además permite la creación de un sistema básico propio, no obstante el ambiente de programación que usa es totalmente integrado al software y esto no permite al estudiante a familiarizarse con una estación de programación real .

Por último, ITS PLC de la empresa Real Games, es un simulador de sistemas completo, pues no solo permite la creación y puesta en marcha de sistemas secuenciales complejos, también es el único de los anteriores que tiene conexión directa con un controlador real, específicamente Siemens [De Magalhães, 2012], y que es una herramienta de entrenamiento de grandes marcas en la industria, sin embargo también exige una computadora con un rendimiento significativo para llevar a cabo la simulación y en su caso ejecutar el programa de control programado dentro del mismo software.

1.3. Formulación del problema

El proceso de enseñanza-aprendizaje en control automático industrial puede presentar circunstancias que limitan la posibilidad del estudiante a desarrollar las competencias necesarias durante su formación. Lo anterior, debido a la falta de contacto que se tiene con los procesos industriales, pues la instalación de un proceso industrial dentro de el área de

práctica puede presentar dificultad para su instalación, haciendo que la probabilidad de contar con uno de estos se reduzca.

Antes de poder contar con la instalación, la cuestión económica juega un papel determinante, pues aún contando con un lugar equipado para ello, el costo únicamente de la instalación puede llegar a ser demasiado elevado, además derivado de esto existen distintos gastos de operación, como los consumibles y la asignación de recursos para el mantenimiento de los equipos instalados, además de los que puedan llegar a necesitarse debido a las situaciones imprevistas hace que el costo de mantener un proceso industrial requiera una inversión significativa.

Otra circunstancia poco favorable es la poca disponibilidad del equipo para la práctica de los estudiantes, pues una cantidad limitada de estos supone un problema en si mismo pues normalmente la demanda de su uso es excesiva para el número de instalaciones disponibles, asimismo las constantes de tiempo de estos procesos suelen ser muy grandes, por lo que puede tomar mucho tiempo regresar a sus condiciones iniciales.

En lo que respecta a las consideraciones previas a una instalación industrial, se encuentra, el uso de diversas fuentes de energía, materiales u otros recursos que generalmente no están disponibles en un ambiente académico, además de la generación de una considerable cantidad de emisiones al ambiente, lo que puede requerir la disposición de residuos y demás desechos con los que la institución se vería obligado a lidiar.

Finalmente existe un factor de riesgo considerable por la propia naturaleza de los procesos industriales, pues durante su operación se está en contacto directo con las fuentes de energía y partes móviles que lo componen, donde se compromete la integridad del estudiante debido a un error en el uso del proceso y el equipo involucrado, además debido a esto es posible que se requieran permisos especiales para poder utilizar cierto tipo de energías o materiales.

Por lo anterior, se requiere una herramienta de apoyo que ayude a los estudiantes a incrementar las competencias y habilidades deseadas y a su vez que se prescinda en las medida de lo posible de recursos que puedan causar mayores complicaciones que beneficios para todos los involucrados en el proceso enseñanza-aprendizaje.

1.4. Objetivo

El objetivo del siguiente trabajo es desarrollar una herramienta para la práctica y enseñanza del control automático industrial, por medio de la emulación de procesos industriales basado en sus comportamientos dinámicos, los cuales serán el resultado de la solución en tiempo real de un sistema de ecuaciones diferenciales por medio de métodos numéricos y representando sus efectos gráficamente por medio de una animación representativa del proceso; el cual será controlado mediante un PAC a través del protocolo de comunicación OPC, obteniendo una base funcional para la integración de cualquier modelo matemático representado en ecuaciones diferenciales ordinarias.

Lo anterior y soportado en la motivación antes planteada, tiene el fin de incrementar los niveles cognitivos en base a la taxonomía de Bloom [Goodhew, 2010], con lo cual un estudiante de control automático será capaz no solo de conocer, describir o comprender los componentes y equipos que conforman un proceso industrial, si no también tendrá la capacidad de análisis, aplicación y experimentación, que un profesional en ingeniería requiere para el diseño y puesta en marcha de un sistema de control industrial.

1.5. Contribuciones

Debido a la extensa cantidad de funcionalidades con las que debe contar el software, se desarrollaron diferentes componentes las cuales interactúan en conjunto para poder llevar a cabo todas las tareas que se requieren y así brindar una experiencia de operación real al estudiante en el momento de emular un sistema.

En primer lugar se cuenta con una interfaz de usuario sencilla y fácil de comprender para el usuario, que fue organizada, para acceder rápidamente a las opciones básicas de la configuración del sistema, así como las de comunicaciones entre el software y el controlador, la instalación se lleva a cabo de manera convencional por medio de un ejecutable que contiene todos los archivos necesarios para utilizar la aplicación.

La comunicación por medio del protocolo *OPC* se hace de manera transparente al usuario, debido a que se cuenta con un cliente capaz de conectarse con cualquier servidor de este tipo, mismo que es estándar en la industria con fines de intercambio de información entre aplicaciones y que es explorado mediante una función recursiva para ubicar y ordenar todos los elementos presentes en el servidor y así acceder con mayor facilidad a los datos que el estudiante requiere para interactuar con el controlador.

Se cuenta con la posibilidad de integrar casi cualquier ambiente de animación para recrear visualmente al proceso que se esta emulando, por ejemplo está versión cuenta con una

animación creada en Adobe Flash CS6 Profesional, la cual permite compartir elementos desde de la aplicación base hacia el servidor *OPC* y viceversa, teniendo como alternativa ejecutar un protocolo de comunicación secundario el cual sirve para establecer y direccionar los elementos dentro del sistema, como los transmisores y actuadores, es decir se desarrolló un protocolo alterno capaz de distinguir entre una comunicación de datos proveniente del controlador o una de menor nivel encargada del intercambio de información entre la animación y la aplicación principal.

Con el fin de poder evaluar el sistema de ecuaciones diferenciales en tiempo real se utilizó un temporizador de alta precisión, el cual genera un hilo de alta prioridad dentro del procesador y así poder llevar a cabo el cálculo de los estados según el método numérico seleccionado, teniendo como paso mínimo de ejecución 1 *[ms]*.

Se integró una pantalla personalizada para el estudiante, la cual recopila información sobre el usuario que utiliza el software, con el fin de crear un enlace de comunicación entre el instructor y el estudiante con el fin de a futuro poder intercambiar información entre computadoras, logrando el monitoreo en tiempo real de la solución y variables generadas a través de una red local, lo cual estaría encaminado a que la labor de evaluación se lleve a cabo de manera más rápida e incluso puedan documentarse los resultados a la vez que son generados.

La contribución más significativa, es el desarrollo de un motor de solución de sistemas de EDO (Ecuaciones diferenciales ordinarias), el cual está preparado para ingresar un sistema de ecuaciones de orden arbitrario y resolverlo en tiempo real con una adición mínima al código, es decir la integración de nuevos sistemas está simplificada a ingresar el modelo en forma de ecuaciones diferenciales con una sintaxis simple, la cual está documentada en el Apéndice A.

1.6. Organización de la tesis

Debido a que el presente trabajo pretende describir diferentes procesos del desarrollo de la aplicación, es conveniente que para cada uno de estos exista la documentación completa y detallada para su mayor comprensión, por tal motivo este documento está dividido como sigue.

En el Capítulo 2, se da breve explicación sobre control automático industrial, describiendo las caracteríticas de los controladores y los protocolos de supervisión industriales, así como la implementación de la comunicación por medio de estándares industriales, su modo de operación y su elección, por último se realiza una comparación con proyectos similares

por medio de la técnica del *Benchmark*, para definir que características son en las que se pueden mejorar en cuanto a las capacidades de cada uno de ellos, pues estos aportan en diferente medida utilidades que se pueden aprovechar por cualquier metodología.

En el Capítulo 3, se realiza el análisis sobre la teoría de los sistemas dinámicos, pues para poder tener un software de comportamiento realista se necesita del estudio de los sistemas dinámicos que intervienen en los procesos, con lo cual se puede lograr un modelo que represente lo más fielmente posible a un proceso, el cual recurrirá al uso de un motor de solución de sistemas de ecuaciones diferenciales ordinarias basado en métodos numéricos.

En el Capítulo 4, se documenta el desarrollo del proyecto, describiendo cada etapa del desarrollo del software, desde la integración de la interfaz, la conexión con el controlador industrial mediante el protocolo OPC, hasta la programación e implementación del árbol de procesos que incluye la ejecución del motor de solución de sistemas de ecuaciones diferenciales.

En el Capítulo 5, se reportan las pruebas de funcionamiento, y se evalúa el desempeño del software interactuando junto con el controlador, analizando el consumo de recursos y las herramientas de reporte con las que cuenta.

Para finalizar se exhibirán las conclusiones sobre las pruebas y tambien las propuestas sobre el trabajo a futuro partiendo de la base que se ha desarrollado con este trabajo.

Capítulo 2

Procesos industriales y estándares de control automático

El control automático industrial es una rama de la ingeniería que se basa en el uso de elementos específicos para conocer el estado de un proceso por ejemplo sensores, actuadores, controladores industriales o incluso computadores con el fin de llevar a cabo tareas pre-establecidas de manera autónoma con la menor intervención humana posible y así mantener los parámetros del sistema dentro de los límites deseados. Normalmente la automatización de un procesos está compuesta por dos partes, las partes mecánicas que ayudan a minimizar las exigencias musculares de los operadores y la parte automatizada que a su vez limita la exigencia sensorial y mental de los operadores.

Debido a la alta demanda de automatización que se ha generado, es habitual que un solo operador tenga a su cargo la operación de varios procesos, por lo cual surgió la necesidad de reunir la información con el fin enfocar la atención del operador en un solo punto que contenga toda la información relevante para él, así pues se desarrolla la automatización descentralizada, es decir fue necesario instaurar la comunicación entre los componentes de un sistema de control con el fin de tener acceso a las variables del proceso, estandarizando los medios y los formatos que se utilizan, los cuales se pueden dividir en dos grupos principales, los protocolos de campo que reúnen a los medios físicos y la manera de enviar la información y los protocolos de control y supervisión que permiten el procesamiento y la visualización obtenida por los protocolos de campo, de la manera que al operador más le convenga.

2.1. Control de procesos en la industria

En muchas ocasiones los procesos industriales pueden parecen ser sencillos o con un principio de funcionamiento simple, sin embargo existe una complicación para el estudiante al

presentar soluciones a los problemas de automatización que estos pueden presentar, como la de estar consciente de las consecuencias que no son obvias para él a primera vista, es decir un profesional en ingeniería puede tener una solución de control automático, sin errores de sintaxis o de ejecución, sin embargo puede tener problemas en su lógica lo cual puede derivar en una reacción no esperada por parte del sistema y esto en la industria normalmente tiene consecuencias de mucha gravedad como paros de línea, perdidas de producto, paros de producción, etc. por ello la herramienta busca sensibilizar al alumno en estos aspectos y así ayudarlo a considerarlos desde la etapa del aprendizaje [Badiru, 2005].

Además de las capacidades cognitivas que debe desarrollar el estudiante, debe también aprender a relacionar los elementos con los que cuenta para poder proponer una solución de control. Como se ha descrito anteriormente uno de los principales problemas que existen en la actualidad es que los estudiantes de estas especialidades no están familiarizados con el tipo de procesos que están presentes en la industria, tan solo tienen una idea de ellos o están muy acostumbrados a ejemplos sencillos y sin mayores consecuencias, de manera que la simulación de procesos ha tomado relevancia durante la etapa de la formación.

Debido a la necesidad de la industria de implementar procesos automatizados existe una gran variedad de estos elementos y equipos, dentro de los que destacan los controladores industriales programables, mismos que han tomado gran importancia en estas aplicaciones, gracias a su confiabilidad, robustez y su gran dinamismo para implementar automatismos complejos, la Figura 2.1 muestra una comparación entre las propiedades de los medios de control comunes actualmente.

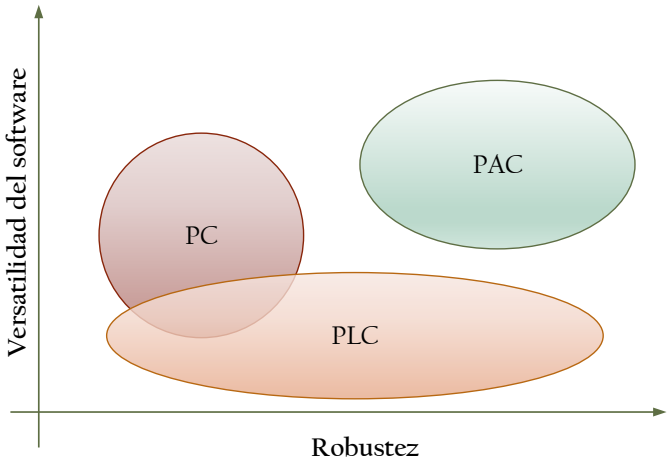

Figura 2.1. Comparación entre equipos de control automático

Así pues, la importancia para un estudiante de automatización industrial de conocer el funcionamiento y dominar la programación de estos equipos se vuelve una materia prioritaria para este, pues debido a su relevancia muchos de los procesos en diseño o existentes están basados en los controladores de automatismos programables, por lo que el entrenamiento que debe tener el estudiante debe ser lo más apegado a la realidad.

2.2. Controladores de automatismos programables

Los controladores programables han sustituido en gran medida a las técnicas de control existentes, debido a que combinan de manera equilibrada y eficiente las características de confiabilidad y robustez de un controlador programable con la versatilidad del software de una computadora. Estos, han ganando terreno en la industria, por ser capaces, de controlar de manera efectiva procesos en los que intervienen cualquier tipo de señales, sin importar que estos sean críticos, o que anteriormente necesitaran un operador de tiempo completo, sustituyéndolo por un controlador .

Con las actualizaciones en la tecnología de automatización industrial, se han desarrollado las capacidades de los controladores industriales, mismos que han tenido que adaptarse a los procesos, no solo como medios de control de procesos repetitivos o peligrosos, pues las nuevas necesidades de la industria le han brindado a los controladores nuevas tareas que se han vuelto más complejas, por lo que de acuerdo a esta evolución se pueden dividir los controladores programables como los controladores lógicos programables o PLC que son una substitución directa de la lógica cableada y los controladores de automatismos programables o PAC, los cuales adquieren nuevas características, como control de movimiento, procesamiento avanzado de señales analógicas y funciones de comunicación avanzada, las propiedades de cada uno son comparadas en la Tabla 2.1 .

Tabla 2.1. Comparación PLC y PAC

PAC	**PLC**
Protocolos libres	Red, comunicación y protocolos propietarios
Lógica basada en excepciones	Funciones basadas en lógica cableada
Procesamiento distribuido	Fundamento en la lógica de escalera
Multitarea	Escaneo de instrucciones contínuo

En adelante y para todo el desarrollo de este documento al referirse Controladores Industriales se referirá a los PAC, teniendo en cuenta que estos son los predominantes en la industria actual y cuyos atributos los hacen factibles los objetivos del desarrollo. Pues una de las principales características del uso de este tipo de controladores es que pueden enviar, recibir y procesar información proveniente de fuentes distantes, cualidad que será aprovechada para el intercambio de la información.

Con base en lo anterior, uno de los temas relevantes durante el proceso de diseño del software fue la implementación de un canal de comunicación entre el controlador y la emulación que se llevará a cabo en una computadora, aprovechando esta capacidad de intercambiar datos de un controlador moderno con el cual es factible realizarlo por medio de dos estándares comunes en la industria.

Actualmente existen diferentes protocolos que permiten el intercambio de datos entre diferentes equipos de control, sin importar el fabricante, algunos de ellos, están pensados exclusivamente para el ámbito industrial y otros son tan robustos y confiables que se usan casi indistintamente en el ambiente doméstico, administrativo e industrial, en este tema se analizan dos de estos protocolos, para intercambiar datos entre aplicaciones.

La elección del protocolo se redujo a dos opciones, las cuales se establecen dentro del equipo que ejecuta el software y ambos intercambian información entre aplicaciones lo cual es una característica importante para los propósitos del desarrollo. Ya que uno de las motivaciones del software es la de no requerir hardware adicional al que se utiliza en un ambiente de trabajo real.

2.3. Protocolos de control y supervisión

Los sistemas de control, actuales tienen la necesidad de intercambiar datos desde y hacia todos los instrumentos y componentes que lo integran. Para llevarlo a cabo existen diferentes protocolos que son utilizados con el fin de estandarizar esta comunicación, sin importar el fabricante o la naturaleza del equipo, es decir si se trata de un transmisor, un controlador, una pantalla de supervisión, etc.

Con base en esto se identificaron dos de los protocolos utilizados por los fabricantes de controladores industriales para intercambiar datos entre aplicaciones que se ejecutan dentro del mismo equipo, en este caso de una computadora. Con el fin de seleccionar el más adecuado se tendrán en cuenta diferentes aspectos de cada uno, como la velocidad de transmisión y la estabilidad de la transferencia de datos. Estos son los protocolos OPC y DDE, OLE for Process Control y Dynamic Data Exchange por sus siglas en inglés,

respectivamente, los cuales son estándares de control y supervisión industriales.

2.3.1. Objetos de incrustación y enlazado

Antes de poder describir a estos protocolos en específico, se debe conocer que son los objetos *OLE Object Linking and Embedding*, traducido como objetos de incrustación y enlazado [Mahnke*, et al.*, 2009].

OLE es conocido normalmente como una actualización del protocolo DDE *Dynamic data exchange*, sin embargo la función de este tipo de objetos es vincular dos aplicaciones, incrustando una sobre otra, de modo el intercambio de datos entre ellas sea inmediato.

El ejemplo clásico de esta superposición de aplicaciones es el de colocar un gráfico de Excel dentro de un documento de Word, y que este a su vez permita editar los datos de manera inmediata, como si se trabajara directamente en Excel. Este protocolo hace que el manejo de información se haga fácil entre aplicaciones de Windows utilizando objetos OLE haciendo uso del protocolo DDE.

2.3.2. OLE for Process Control

Tras conocer la función de los objetos OLE, se puede entender más fácilmente el concepto de OPC, que esta diseñado para hacer uso de ellos dentro de un ambiente industrial de manera eficiente y confiable, como consecuencia de la propuesta de grupo de fabricantes junto con Microsoft que desarrollaron una especificación abierta que permitiera gestionar datos entre aplicaciones y hardware de diferentes equipos, observando la necesidad de estandarizar el flujo de datos y dejar de lado las comunicaciones restrictivas que un protocolo propietario, para lo cual se creó un modelo permitiría la comunicación universal entre fabricantes [Mahnke*, et al.*, 2009].

Al igual que en DDE se puede tener acceso a diferentes datos disponibles en los equipos, sin embargo la estructura de entrada en forma de árbol, similar a un explorador de Windows, permite agrupar etiquetas de identificación para la información en grupos con una ruta especifica y cuyos elementos mínimos de información son llamados *items*. Así cada fabricante podrá acceder a cualquier variable que necesite tener acceso de cualquier fabricante o equipo conociendo únicamente la ruta de dicho elemento [Iwanitz y Lange, 2010].

La estructura básica del protocolo de comunicación es la ilustrada en la Figura 2.2.

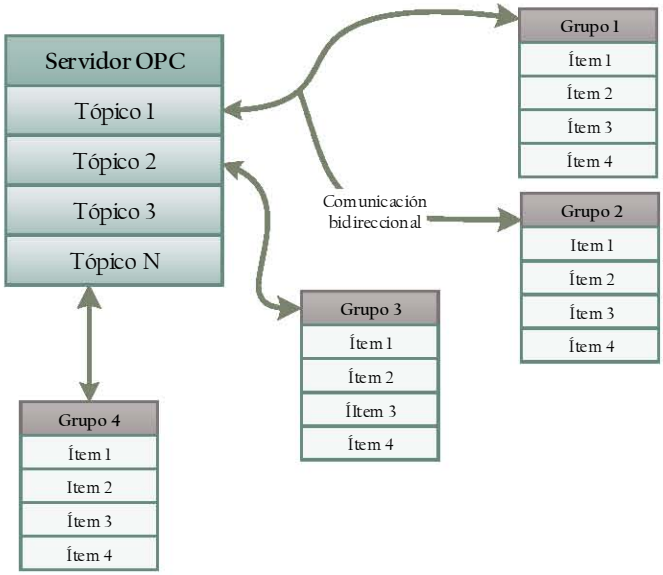

Figura 2.2. Estructura básica de servidor OPC

2.3.3. OPC en el emulador de procesos

A pesar de que ambos protocolos son estándares que permiten una comunicación libre, existen características que por la naturaleza de cada protocolo harán que se implemente o se descarte para el desarrollo del software. El protocolo OPC cumple con los requerimientos que se necesitan para alcanzar los objetivos se plantearon al inicio, las características que se destacan para su utilización son:

- Ofrece mayor fluidez en la transmisión de datos, ya que fue diseñado para obtener un mejor rendimiento, pues es importante que el flujo de datos se realice de manera constante para que el proceso de la emulación no sufra retrasos o pérdidas de información durante su ejecución.
- Ofrece una mayor flexibilidad para el desarrollo de aplicaciones.
- Soporta múltiples clientes, en comparación con DDE que acepta uno por cada consulta al servidor.
- Manejo de objetos OLE, para intercambio de datos entre aplicaciones.

Por tratarse de estándares industriales se han desarrollado componentes de programación que permiten la implementación del protocolo de manera sencilla; uno de estos es el complemento dOPC Kassl para la integración de clientes dentro de aplicaciones desarrolladas

bajo el entorno Delphi.

2.3.4. Dynamic Data Exchange

El protocolo DDE (*Dynamic Data Exchange*, por su siglas en inglés) cuya traducción es intercambio dinámico de datos, es un protocolo de comunicación, basado en la interacción entre dos aplicaciones que se ejecutan bajo el sistema operativo Windows. Este permite que una aplicación se comunique con otra para poder manipular los datos de origen en la aplicación de destino.

DDE trabaja de la siguiente forma, si se tiene un controlador industrial conectado a una red y nuestra computadora tiene instalado un servidor DDE, este tiene el trabajo de comunicar la computadora con la red industrial, se puede comunicarnos con el controlador llamando a las etiquetas establecidas para referirse a las variables enviadas o recibidas por él, obedeciendo a las peticiones del servidor DDE, quien a su vez entrega los datos que recibe a las aplicaciones Cliente de la computadora, como puede ser una hoja de Excel.

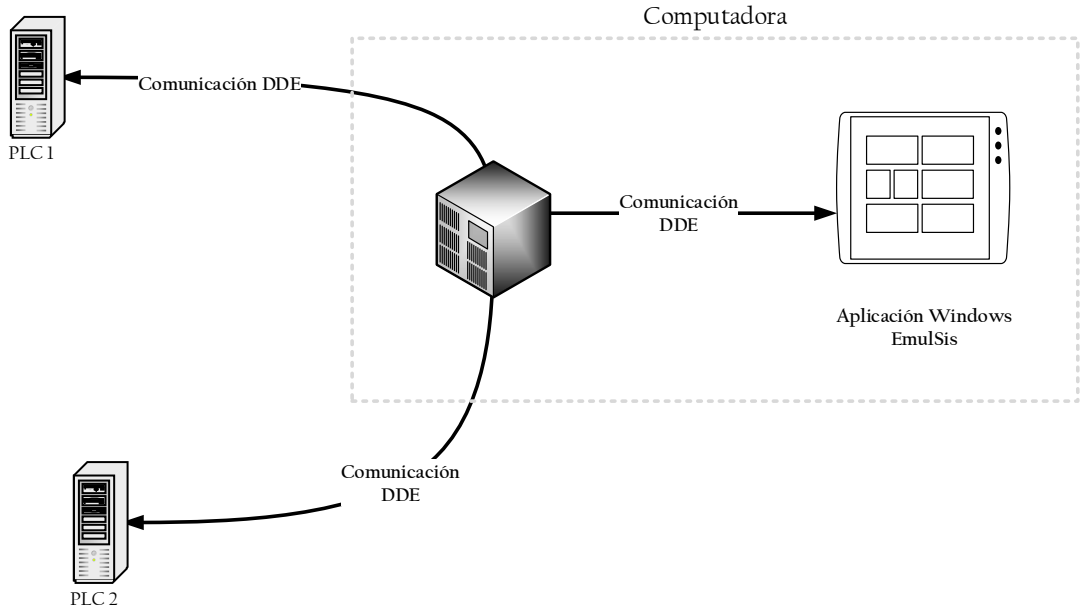

Figura 2.3. Diagrama del protocolo DDE

Capítulo 3

Sistemas dinámicos en procesos industriales

Una de las cualidades con las que debe contar el software, es la de tener la capacidad de reproducir los comportamientos dinámicos presentes en procesos industriales, sin importar la complejidad o el número de variables que intervengan en éstos, con el fin de producir una emulación realista de cualquier tipo de sistema, para lo cual es necesario contar con un modelo matemático en la forma de sistemas de ecuaciones diferenciales ordinarias.

Los sistemas dinámicos pueden poseer características con las que pueden clasificarse en diferentes tipos. En algunas ocasiones puede ocurrir que un sistema puede pertenecer a una categoría, cuando éste sea modelado para cierto tipo de análisis y también pueda clasificarse de manera distinta cuando el objetivo tenga otros propósitos.

Estas clasificaciones pueden darse por diferentes factores, como en sistemas lineales o no lineales; también si se trata de un sistema continuo o un sistema discreto, o bien pueden dividirse en sistemas de parámetros concentrados donde las variables del análisis se concentran en un punto del sistema, y de parámetros no concentrados o de parámetros distribuidos, que analiza la concentración o la variación de una u otra variable de análisis en un punto especifico del sistema [Moctezuma, 2015].

El objetivo de conocer estas características de los sistemas, es principalmente la de identificar el tipo de comportamientos dinámicos que son requeridos para que el software realice la emulación, y que en este caso será necesario un modelado de espacio de estados representado en un sistema de ecuaciones diferenciales, sin importar si se trata de ecuaciones lineales o no lineales.

Con este fin se tiene como propósito la implementación del denominado motor de solución de sistemas de EDO, por lo que no es necesario que se llegue a una solución analítica de

un sistema en especifico, ya que el software esta destinado únicamente a la emulación y la presentación de resultados de manera gráfica, y que utilizará una solución numérica para obtener los estados en cada instante de tiempo.

3.1. Métodos de solución numérica

Para que el software cumpla con los objetivos establecidos es necesario obtener una solución explícita del sistema de ecuaciones, lo que quiere decir que no es necesario obtener los resultados de la solución analítica, permitiendo hacer uso de métodos numéricos para obtener las soluciones con una exactitud aceptable.

Para esto, se debe tomar en consideración lo siguiente, como el software está destinado a ofrecer una solución en tiempo real, el número de cálculos por segundo que se realizan debe tener el menor impacto posible en el rendimiento del equipo, además de procurar la incfrementando exactitud de la solución posible, ya que está determinada por el diferencial de tiempo con el que se realizan los cálculos, y que tiene como límite 1 [ms] que corresponde al periodo de resolución máximo ofrecida por el software para llevar a cabo la solución del sistema.

Pensando en estas consideraciones, se utilizan los siguientes métodos para llevar a cabo el algoritmo de solución, los cuales son: el Método de Euler, método de Euler mejorado y el método de Runge-Kutta de 4to orden; los cuales ofrecen menor cantidad de cálculos por segundo para mejorar el rendimiento o mayor exactitud de resolución según el método seleccionado, mismos que se describen brevemente a continuación.

3.1.1. Método de Euler

El método presentado por Leonhard Euler, fue el primero en ser publicado para la solución de sistemas de ecuaciones diferenciales de primer orden de la forma (3.1):

$$\frac{dx}{dt} = f(x,t) \tag{3.1}$$

Utilizando una condición inicial $x(t_0) = x_0$, el método de Euler está basado en la aproximación de un diferencial Δt finita, de una derivada continua $\frac{dx}{dt}$, derivado de la ecuación

de Taylor mostrada en 3.2, [Kulakowski, *et al.*, 2007].

$$\frac{dx}{dt} \approx \frac{[x(t_0 + \Delta t) - x(t_0)]}{\Delta t} \tag{3.2}$$

De la cual se obtiene la solución aproximada dada por la expresión 3.3:

$$x(t_0 + \Delta t) \approx x(t_0) + f(x_0, t_0)\Delta t \tag{3.3}$$

De la forma anterior se puede aplicar el algoritmo para cualquier sistema de la forma:

$$\frac{dq_i}{dt} = f_i(\mathbf{q}, t), i = 1, 2, ..., n$$

3.1.2. Método de Euler mejorado

Este método puede ser denominado también método de Heun o método de Runge-Kutta de 2do orden. La diferencia entre este método y el método anterior, es que este busca un refinamiento, encontrando un promedio entre dos valores obtenidos, lo cual hace que requiera dos pasos para encontrar una solución, primer se utiliza la ecuación del método de Euler, para lo cual se re escribe la ecuación 3.3, de la forma:

$$\dot{x}(t_0 + \Delta t) = x(t_0) + k_1\Delta t \tag{3.4}$$

donde

$$k_1 = f(x_0, t_0)$$

Ahora, el valor de $\dot{x}$ se usa para aproximar la pendiente utilizando la expresión:

$$k_2 = f[\dot{x}(t_0 + \Delta t), t_0 + \Delta t]$$

Así la aproximación utiliza las dos pendientes resultantes para obtener la solución de acuerdo a la expresión 3.5:

$$x(t_0 + \Delta t) \cong x(t_0) + \frac{k_1 + k_2}{2}\Delta t \tag{3.5}$$

El método requiere un paso extra de cálculo pero se obtiene una mejor aproximación durante el cálculo del estado en el tiempo t para todos los estados.

3.1.3. Método de Runge-Kutta

Este método es la conjunción de diferentes niveles de aproximación, de los cuales los primeros dos niveles son idénticos a los de los métodos de Euler y de Euler mejorado respectivamente, donde se calculan dos aproximaciones más para implementar el método de 4to orden. En las maneras más comunes de la implementación se tiene la expresión siguiente 3.6, [Kulakowski*, et al.*, 2007]:

$$x(t_0) \tag{3.6}$$

Donde:

$$k_1 = f[x(t_0), t_0] \tag{3.7}$$

$$k_2 = f[(x(t_0) + \Delta t\frac{k_1}{2}), (t_0 + \frac{\Delta t}{2})] \tag{3.8}$$

$$k_3 = f[(x(t_0) + \Delta t\frac{k_2}{2}), (t_0 + \frac{\Delta t}{2})] \tag{3.9}$$

$$k_4 = f[(x(t_0) + \Delta t k_3), (t_0 + \Delta t))] \tag{3.10}$$

Entonces el método está dado por la siguiente ecuación:

$$y_{i+1} = y_i + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4) \tag{3.11}$$

De los tres métodos utilizados este es el que presenta el menor error con respecto a la solución analítica, cabe mencionar que el error se disminuye en mayor medida si se utiliza un valor de Δt pequeño, el software tiene por defecto un valor de $0{,}005s$, pues es él tiempo de muestreo mínimo para un controlador industrial de alta gama, sin embargo se puede modificar aunque no se recomienda disminuir demasiado este parámetro ya que el rendimiento del equipo se puede ver afectado.

Se puede escoger cualquiera de los tres métodos al ejecutar el software, dependiendo de las características del equipo donde este se ejecuta, con el fin de adecuarse con su rendimiento, a pesar de esto los valores de los estados se calculan en tiempo real. Con base en lo anterior se eligieron los métodos antes descritos.

3.2. Proceso tanque de agua atmosférico

Con la posibilidad de poder llevar a cabo la solución de los sistemas con las características de desempeño y velocidad necesarias, es posible hacer implementar la representación gráfica del proceso, la cual estará ligada a los valores obtenidos por el software, por lo cual basta con obtener un modelo e integrarlo a la batería de procesos disponibles dentro del software, para lo cual se hizo la primera elección de estos, pensando en poner a prueba la capacidad del software para resolver sistemas complejos en tiempo real.

La elección del proceso se llevo a cabo teniendo en cuenta la necesidad de observar procesos industriales reales, para lo cual se eligió el proceso de llenado con agua de un tanque atmosférico, siendo este un proceso recurrente en la industria y que puede servir como un ejemplo didáctico con diversas maneras de implementación y variables de comportamiento, lo cual lo hace un proceso ideal para la enseñanza.

El tanque de agua atmosférico implementado cuenta con diferentes equipos que presentan dinámicas que pueden ser modeladas, algunas de las cuales pueden ser manipuladas para obtener diferentes efectos con respecto a la solución de su sistema de EDO, en el cual la variable de control es la de la altura de la columna de agua, como parte del sistema se tiene: una bomba, dos switches de nivel, una válvula neumática y el propio tanque.

El sistema representado por un diagrama de instrumentos y tuberías o P&ID (Pipes and Instrumentation Diagram por sus siglas en inglés), es el siguiente según especifica en las normas ISA (*International society of automation*), por su traducción la sociedad internacional para la automatización [Creus Solé, 2012]:

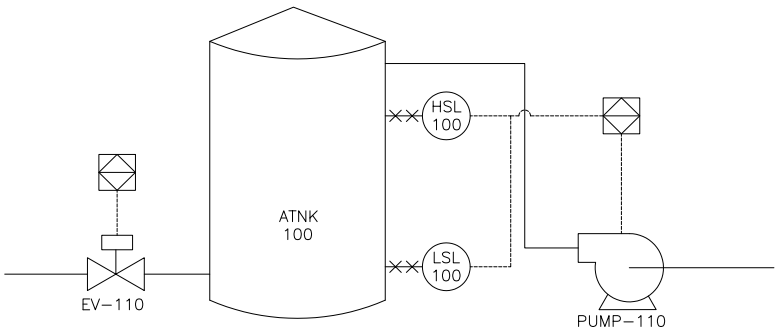

Figura 3.1. P&ID de proceso de tanque atmosférico

3.3. Modelo matemático

El sistema que se utilizó para desarrollar y probar el programa de emulación fue el de llenado de un tanque atmosférico, el cual está constituido por un a bomba de llenado, una válvula para el vaciado y dos switches de nivel, el cual posee diferentes parámetros que se pueden modificar para poder poner en practica distintos tipos de solución a diferentes problemas de control, aunque parece un sistema simple, resulta ser muy recurrente dentro de la industria, además se llevó el modelado de este sistema a un nivel de detalle lo suficientemente descriptivo como para ver los detalles que arroja una solución propuesta por parte del estudiante, con el fin de que este pueda observar el resultado de su propuesta.

Para poder describir la integración del modelo dinámico se divide la generación del modelo en base a los elementos que lo conforman, lo cuales se detalla a continuación.

3.3.1. Tanque y válvula

La variable de proceso que se puede controlar para este sistema es la altura del líquido en el tanque, por tanto el modelo tiene como salida a este estado, para esto se necesita utilizar las dimensiones realistas de un tanque [Janevska, 2013].

Algunos de los parámetros a continuación se pueden modificar para poder variar el comportamiento del tanque y así obtener más opciones de soluciones de control. Para la parte en especifico de la válvula de vaciado y el tanque, en si mismo se tienen las siguiente ecuaciones:

$$\dot{H} = \frac{1}{A_t}(q_i - q_0 - q_r) \tag{3.12}$$

$$q_0 = (V_c + V_c z_1)C_c A_o + T_s C_c A_o \sqrt{2gH \underbrace{\left(\frac{1}{2}sgn\,(H) + \frac{1}{2}\,|sgn\,(H)|\right)}_{CondicionA}} \tag{3.13}$$

$$q_r = |q_i - q_0 + T_r T_s| \underbrace{C_R \left(\frac{1}{2}sgn\,(q_i) + \frac{1}{2}\,|sgn\,(q_i)|\right)}_{CondicionB} \tag{3.14}$$

$$C_R = \left(\frac{1}{2}sgn\,(H - H_{max}) + \frac{1}{2}\,|sgn\,(H - H_{max})|\right) \tag{3.15}$$

$$T_s = \left|a_1 sin(8t) + a_2 sin(3t) + a_3 sin(t)(\tfrac{1}{H+1})(\tfrac{100q_i}{q_i+1})\right| \tag{3.16}$$

$$\dot{V_c} = z_1 \tag{3.17}$$

$$\dot{z_1} = -\lambda_1^2 V_c - 2\lambda_1 z_1 + \lambda_1^2 K_V \tag{3.18}$$

Los parámetros que aparecen en estas expresiones representan las siguientes magnitudes, las cuales han sido determinadas para mantener un comportamiento realista durante la emulación:

Tabla 3.1. Parámetros para el modelo de llenado

A_t	Representa el área de la base del tanque.
H	Nivel del tanque, que en este caso es la variable de proceso, es decir la variable a controlar.
H_{max}	Nivel máximo del agua o la altura del tanque , que esta medido en metros.
q_i	Flujo de entrada al tanque, debido a la bomba de llenado
q_0	Flujo de salida del tanque
q_r	Flujo de rebose, este parámetro toma efecto cuando el tanque llega a su máxima capacidad, lo que significa que el líquido debe derramarse y puede tomarse como un flujo extra de salida del tanque, pero que no esta relacionado con el diámetro de la tubería de la válvula.
V_c	Estado de apertura de la válvula de vaciado, el cual únicamente toma un valor de 0 o 1 para indicar el estado de la misma.
T_d	Coeficiente de turbulencia de descarga.
T_r	Coeficiente de turbulencia de rebose.
T_S	Turbulencia superficial del agua por el efecto de q_i.
K_V	Señal de posición de la valvula, pero esta vez en como la señal de control de la misma.
a_1	Amplitud de efecto de la ola en la superficie.
A_0	Sección transversal de la línea de salida del tanque.

Para lo cual se utilizan los siguientes valores como pre-establecidos para la realización de las pruebas.

Tabla 3.2. Parámetros de tanque y válvula

Coeficiente de turbulencia de descarga T_d	0.15
Coeficiente de turbulencia de rebose T_r	0.025
Amplitud de ondas de turbulencia en a_1	0.0015
Sección transversal de la línea de salida A_0	0.0025

Los parámetros de la Tabla 3.2 pueden tomar valores arbitrarios, pero están restringidos a valores pequeños ya estos representan turbulencias o ruido dentro del sistema, igualmente pueden tener un valor de 0 si es que se quiere mejorar el rendimiento, sin embargo el último parámetro A_0 que es el diámetro de la tubería debe tener un valor razonable, ya que si ingresa un valor muy grande afectaría al flujo de salida, lo cual haría que aunque se obtuvieran resultados matemáticamente correctos visualmente la animación tendría un comportamiento sin sentido e irreal, lo cual trata de evitarse a toda costa pues el objetivo es tener la sensación más realista posible.

3.3.2. Motor de inducción trifásico

Aquí se considera el modelo de un motor de inducción trifásico de multiples pares de polos tipo jaula de ardilla, representado en un plano bifásico ortogonal equivalente por medio de de la transformación de Blondel [Blondel, *et al.*, 1913]. En este marco de referencia y por consideración de que las fases son simétricas y además distribuidas sinusoidalmente, se evita la dependencia de la posición del rotor, lo que simplifica en gran medida el análisis [Liu, *et al.*, 1989].

Para la obtención del modelo matemático del motor de inducción se supone la existencia de una relación lineal entre flujo magnéticos y corrientes eléctricas. También se considera la permeabilidad magnética en los núcleos laminados como infinita despreciando los efectos en las ranuras, las pérdidas en el hierro y en los devanados. Dado lo anterior y a la aplicación de la Ley de Gauss y la Ley de Ampere [Meisel, 1984] se tiene que:

$$\begin{bmatrix}\psi_s \\ \psi_r\end{bmatrix} = \begin{bmatrix}L_s I_2 & L_{sr} I_2 \\ L_{sr} I_2 & L_r I_2\end{bmatrix} \begin{bmatrix}I_s \\ I_r\end{bmatrix} \tag{3.19}$$

Donde $\psi \triangleq [\psi_s^T, \psi_r^T]^T \in \Re^4$ es el vector de encadenamiento de flujos, $I \triangleq [I_s^T\ , I_r^T]^T \in \Re^4$ es el vector de corrientes, L_s, L_r, $L_{sr} > 0$ son las inductancias en estator, rotor y mutuas respectivamente, e $I_2 \in \Re^{2\times 2}$ es una matriz de identidad. Los subindices $(.)_s$ y $(.)_r$ son usados para denotar variables de estator y rotor respectivamente.

A partir de la expresión anterior es posible obtener una representación no lineal de la dinámica del motor empleando las corrientes de estator y los flujos de rotor como variables de estado eléctricas del sistema [Espinosa y Ortega, 1994]. Este modelo, conocido en la literatura como el *modelo ab* [Seely, 1962], [Meisel, 1984], modelo de Stanley [Krishnan, 2001] o modelo en el marco de referencia fijo al estator [Krause, *et al.*, 2002], esta dado por:

$$\dot{I}_s = \gamma I_s + \left(\frac{L_{sr} R_r}{\sigma L_r^2}\right)\psi_r - \left(\frac{n_p L_{sr}}{\sigma L_r}\right)\omega J \psi_r + \frac{U_s}{\sigma} \tag{3.20}$$

$$\dot{\psi}_r = -\left(\frac{R_r}{L_r}\right)\psi_r + (n_p \omega J)\psi_r + \left(\frac{R_r L_{sr}}{L_r}\right) I_s \tag{3.21}$$

$$\dot{\omega} = \left(\frac{1}{J}\right)\frac{n_p L_{sr}}{L_r} I_s^T J \psi_r - \left(\frac{B}{J}\right)\omega - \frac{T_L}{J} \tag{3.22}$$

Donde τ_e es el par electromagnético, ω la velocidad en el eje del motor, $R_s, R_r > 0$ las resistencias en estator y rotor respectivamente, n_p es número de par de polo, $J > 0$ la inercia del rotor, $B \geq 0$ el coeficiente de amortiguamiento mecánico o fricción viscosa, τ_L el par de carga externo aplicado al eje del rotor, $U_s \in \Re^2$ los voltajes de estator mientras que:

$$\gamma = \left(\frac{L_{sr}^2 R_r}{\sigma L_r^2} + \frac{R_s}{\sigma}\right) \tag{3.23}$$

$$\jmath = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{3.24}$$

Con el fin de mantener el realismo en la emulación del sistema, se tomaron en consideración parámetros de una bomba Grundfos CR15-1 conectada a una tensión de 230$[V_{AC}]$ y cuyos parámetros completos son los siguientes:

Tabla 3.3. Parámetros de motor de inducción

Potencia	0.746 kW
Amplitud de tensión del estator	$U_S = 230V$
Resistencia de estator	$R_s = 2{,}516\Omega$
Resistencia de rotor	$R_r = 1{,}9461\Omega$
Inductancia de estator	$L_s = 0{,}2340mH$
Inductancia de rotor	$L_r = 0{,}2302mH$
Inductancia mutua	$L_{sr} = 0{,}2226mH$
Fricción viscosa	$B = 1{,}1 \times 10^{-4} N \cdot m \cdot s/rad$
Coeficiente de momento de inercia	$J_m = 6{,}04675 \times 10^{-3} kg \cdot m^2$

3.3.3. Bomba centrífuga

Esta parte del sistema, describe el comportamiento del agua dentro de la bomba, lo que genera el flujo de entrada al tanque, como función de las ecuaciones anteriores con el fin de que este responda con un flujo acorde al comportamiento de la bomba.

$$\dot{\tau_{Lb}} = z_2 \tag{3.25}$$

$$\dot{z_2} = -\lambda_2^2 \tau_{Lb} - \lambda_2 z_2 + \lambda_2^2 \left\{ (k_p q_i + K_z \alpha_2 N) \frac{\omega}{2} \right\} \tag{3.26}$$

$$q_i = \left| \frac{k_v Y}{\sqrt{1 + k_c k_v^2 Y^2}} \sqrt{AC_I} z_2 \alpha_1 N \right| \tag{3.27}$$

$$C_I = \underbrace{\left(\frac{1}{2} sgn\,(A) + \frac{1}{2} \left| sgn\,(A) \right| \right)}_{CondicionC} \tag{3.28}$$

$$A = k_w \omega^2 - \rho g h_{gv} + p_b - p_2 \tag{3.29}$$

$$\tau_L = \tau_{Lb} + z_2 \alpha_3 N \tag{3.30}$$

Tabla 3.4. Parámetros de bomba centrífuga

ω	Velocidad de rotor
q_i	Flujo a través de la bomba
C_1	Condición de flujo de entrada al tanque o rebose
A	Medición del flujo de entrada mediante la altura del tanque

Los modelos anteriores sirvieron para construir uno que describa al sistema en su totalidad, y tal como se mencionó en los objetivos iniciales, el análisis de este a detalle no es uno de ellos; únicamente se pretende obtener un sistema lo suficientemente descriptivo que nos permita la emulación del tanque y sus componentes, con los comportamientos dinámicos adecuados para una representación realista como base de una evaluación confiable del motor de solución de sistemas implementado, el cual es ingresado al software de la siguiente manera.

3.3.4. Modelo de ingreso al motor de solución de sistemas

A continuación se muestra el modelo en espacio de estados completo tal cual lo recibe en software para su resolución: .

$$
\begin{aligned}
\dot{x_0} &= \frac{1}{A}(q_i - q_0 - q_r) \\
\dot{x_1} &= x_2 \\
\dot{x_2} &= -\lambda_1^2 x_1 - 2 - \lambda_1 x_2 + \lambda_1^2 A_v \\
\dot{x_3} &= -\gamma x_3 + \frac{L_s r R_r}{\sigma L_r^2} x_5 + \frac{N_p L_s r}{\sigma L r} x_7 x_6 + \frac{u_1}{\sigma} \\
\dot{x_4} &= -\gamma x_4 + \frac{L_s r R_r}{\sigma L_r^2} x_6 + \frac{N_p L_s r}{\sigma L r} x_7 x_5 + \frac{u_2}{\sigma} \\
\dot{x_5} &= -\frac{R_r}{L_r} x_5 - n_p x_7 x_6 + \frac{R_r L_s r}{L r} x_3 \\
\dot{x_6} &= -\frac{R_r}{L_r} x_6 + -n_p x_7 x_5 + \frac{R_r L_s r}{L r} x_4 \\
\dot{x_7} &= \frac{N_p L_s r}{J m L r}(x_4 x_5 - x_3 x_6) - \frac{B}{J_m} x_7 - \frac{T_l}{J_m} \\
\dot{x_8} &= x_9 \\
\dot{x_9} &= -\lambda_2^2 x_8 - \lambda_2 x_9 + \lambda_2^2 M_p
\end{aligned}
$$

Es importante observar en el modelo anterior que las operaciones matriciales que se pueden encontrar ya están desarrolladas, los valores de las variables que no son estados únicamente están indicados, pues estos pueden tener valores diferentes dependiendo de los parámetros de los equipos que se elijan, esta es la representación tal cual la recibe el motor de soluciones, por lo cual estos valores son calculados cada vez.

Capítulo 4

Diseño y desarrollo del emulador de procesos industriales

Durante la planeación y el diseño de la aplicación se deben consideran distintos aspectos al pretender que el programa cumpla con todas las características que se plantearon en los objetivos de este trabajo. Por ejemplo, además de buscar la forma en la que el software se comunicará de manera eficiente o la manera en la que se resolverán los sistemas, también se debe tomar en cuenta la interfaz de usuario para hacer al software más fácil de usar para el estudiante.

Por tanto se tendrá especial atención en la funcionalidad del software, la facilidad para navegar por las opciones y hacer lo más sencilla posible la configuración de parámetros de proceso. Por ejemplo, se sintetizó la elección de opciones de comunicación entre un servidor OPC y el cliente con el que cuenta software, por lo que no es necesario que el estudiante comprenda en su totalidad el funcionamiento del protocolo, únicamente está presente por ser un estándar industrial.

Lo anterior creará conjunto de funcionalidades para integrar un software completo y funcional, además de estar diseñado para continuar y complementar el desarrollo, por consiguiente se le ha nombrado *EmulSis* a la primer versión, por la unión de palabras Emulador de Sistemas, pues una emulación por definición es la imitación de las características de algo procurando igualarlas o mejorarlas, cuyo aspecto se muestra en la Figura 4.1.

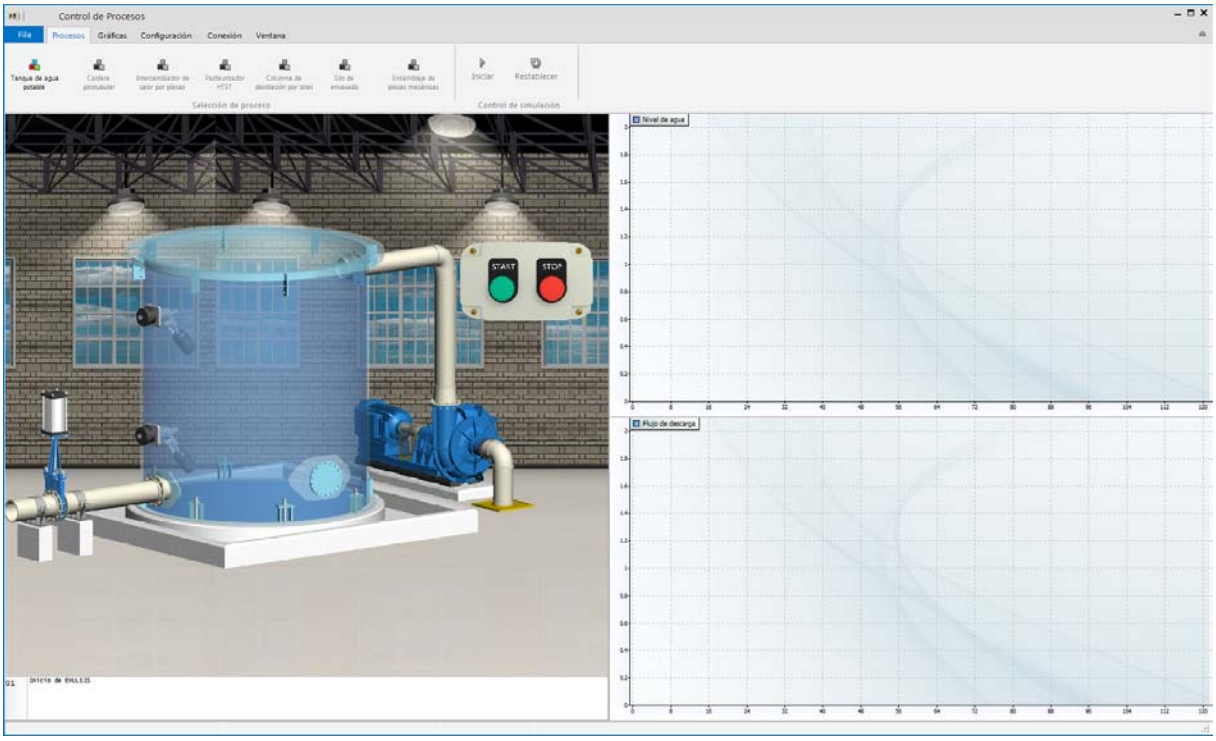

Figura 4.1. Ventana de Software EmulSis

4.1. Técnica de *Benchmark* aplicada a herramientas similares

La metodología como conocida como *Benchmark* se emplea con el fin de poder comparar las características de un producto determinado, dicha metodología es utilizada comúnmente en el campo de la mercadotecnia para planear estrategias de mercado y en el ambiente de la informática para comparar y observar las ventajas y desventajas de un determinado software frente a un grupo de aplicaciones similares.

La técnica de *Benchmark* actúa como un catalizador para la mejora continua al identificar cualquier deficit existente en el producto, relacionado con el rendimiento, en comparación con el de los demás dentro del grupo de estudio; también proporciona un camino efectivo para alcanzar las mejores prácticas en el desarrollo de productos o sistemas [Wardhaugh, 2003].

Por otro lado, en [Clemente y Laburu, 2005] se define al método del Benchmark como una comparación sistemática de procesos y/o resultados; sin embargo, su objetivo es el de aprender de los procesos o prácticas que permitan lograr los mejores resultados, y adaptarlos al producto en diseño. Por tanto se puede decir que es una herramienta dirigida

a la acción o cambio, que implica aprendizaje, gestión del conocimiento y adaptación de prácticas en busca de la mejora continua.
Sin embargo [Badia*, et al.*, 1999], menciona que el proceso de *Benchmark* no consiste en copiar las mejores características, sino en aprenderlas y aplicarlas mediante la adaptación, creación y rediseño del producto.

En este proyecto, como parte del diseño y desarrollo de la herramienta de software, se llevó a cabo este análisis con el fin de incrementar el potencial del software. Para este proceso se seleccionaron las siguientes aplicaciones:

- LogixPro v1.87
- VirPLC v3.12
- Mitsubishi FX-Training v1.0
- ITS PLC Pro v1.3.6
- LogixSim

Se encontraron diversas aplicaciones como las mencionadas en la sección 1.2 que están orientadas hacia la formación en control industrial, donde algunas se enfocaban principalmente en mostrar su función como herramientas didácticas, otros estaban encaminados a usuarios que ya tienen una base de conocimientos en la materia y dos de estos ya son aplicaciones funcionales que se encuentran en etapa de distribución y están siendo utilizadas en distintas instituciones con fines formativos.

Se han descartado algunas de las aplicaciones que a pesar de compartir objetivos con este proyecto, se encuentran en una fase de desarrollo similar a este o se fueron abandonados por sus desarrolladores, por lo cual el análisis de sus funcionalidades sería de poco interés.

Tras utilizar las aplicaciones seleccionadas fue posible mejorar el diseño y funcionalidades de EmulSis. En la Figura4.2 se puede ver una síntesis de la aplicación de la técnica de Benchmark, donde se exponen las características en las que el software muestra fortalezas en comparación con cada aplicación, y también las áreas de oportunidad en las que se puede mejorar desde la planeación inmediata o a futuro, por tal motivo se puede decir que la aplicación de Benchmark es importante para el desarrollo del proyecto, pues esta inicia el proceso de diseño formal de la composición operativa de la aplicación, tomando como premisa principal el apoyo a la formación de profesionales en ingeniería.

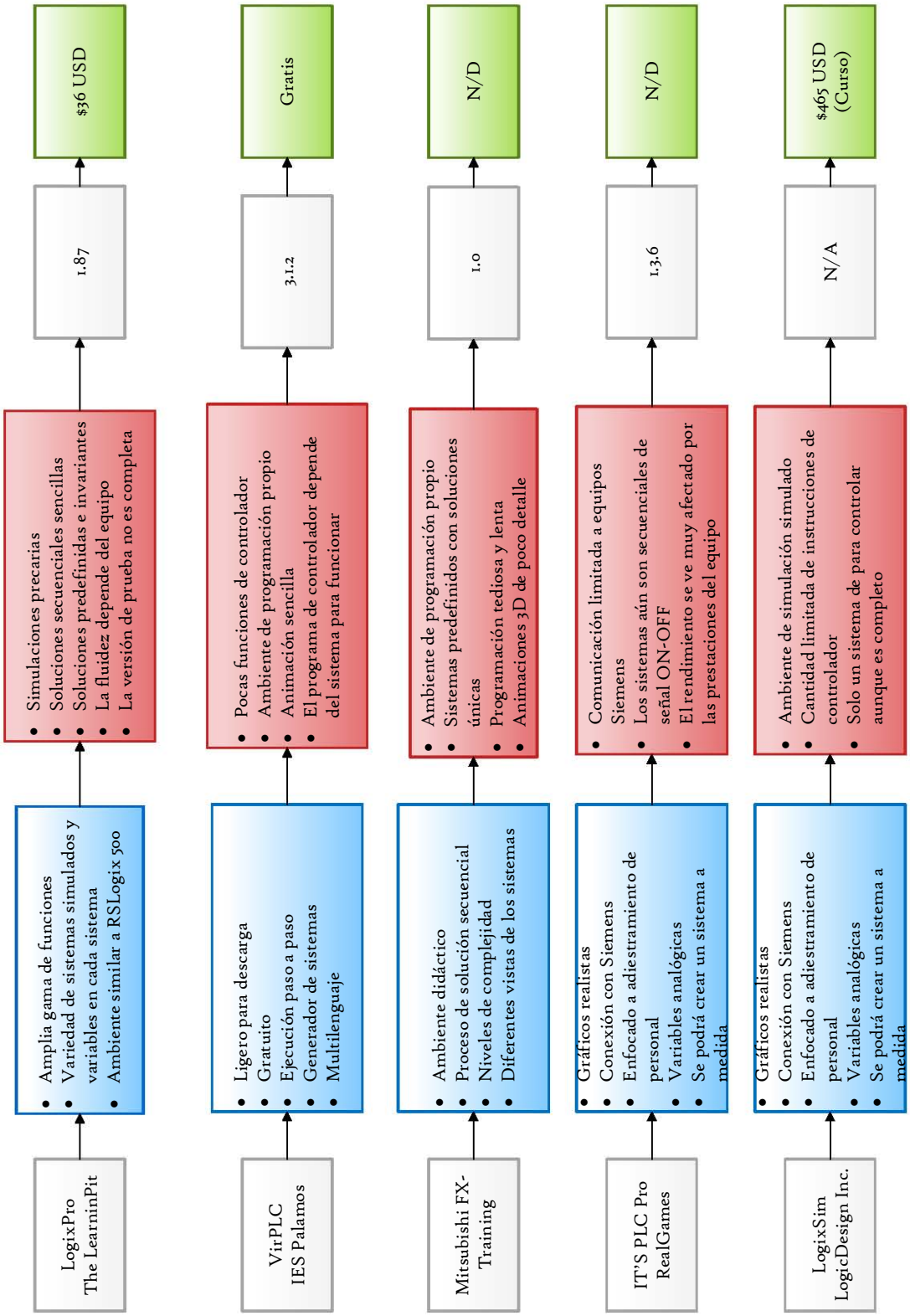

Figura 4.2. Resultado de la aplicación del estudio de Benchmark

Se pueden observar los puntos principales del análisis entre las aplicaciones, los cuales se relacionan directamente con la formación, la calidad de la representación gráfica del proceso y el seguimiento a las actualizaciones del software en cada caso.

Considerando la información mostrada en la Figura 4.2 se puede observar que existen algunas áreas de oportunidad como la representación visual y la tecnología utilizada para la animación o la interfaz gráfica y algunas configuraciones, sin embargo el motor de solución de sistemas de EDO ofrece una gran ventaja con respecto a los demás, pues brinda la posibilidad de emular cualquier proceso y con esto aumentar el número de procesos a emular de manera ilimitada, dado que en comparación a los demás, EmulSis no se limita a un proceso en particular o la implementación de una solución de control especifica, ya que está enfocado a la representación dinámica de cualquier sistema, por tal motivo se pueden manipular, añadir o reducir el número de variables en cada proceso.

Esto último se considera como una de las características más importantes que se dan muestran en el resultado del estudio, pues le da posibilidad de implementar cualquier sistema y emularlo por medio del software manteniendo las mismas propiedades de desempeño y representación en tiempo real, la cual no esta presente en ningún software documentado.

4.2. Análisis de funcionalidad

Estas cualidades se pueden separar en dos categorías principales, la primera es la manera en que el software lleva a cabo la emulación del sistema, dejando de lado el sistema dinámico, podemos observar que la manera en que se lleva a cabo este proceso pues el sistema tiene la capacidad de recibir y utilizar las variables que el motor de solución arroja en tiempo real de la misma manera que se envían o reciben estímulos de un proceso instrumentado real, es decir es posible observar todos los comportamientos del sistema en todo momento.

Para esto se puede emular la respuesta de cualquier elementos de control, por ejemplo, gracias a que se conoce la altura de la columna de agua dentro del tanque podemos enviar al controlador una señal de valor numérico que represente la señal de un transmisor de presión diferencial para medir la altura de la columna de agua como una variable analógica, de la misma manera en que lo hace el transmisor real hacia un módulo de entradas analógicas, lo cual permitiría implementar un control PID si se requiriera, o bien se podrían enviar variables de control analógicas como la frecuencia en la bomba o la posición de la válvula para este sistema en particular, lo cual es la posible por la presencia de la frecuencia dentro del modelo.

Una característica importante de la programación del software se basa en no requerir de una capacitación intensiva para aprender a configurarlo, por ello esto se incorporó una interfaz simplificada para este propósito, además de la posibilidad de hacerlo mediante de un asistente de configuración; con el fin de no requerir un conocimiento avanzado del uso de protocolos industriales pero manteniendo los estándares reales de comunicación entre software EmulSis con cualquier controlador o aplicación que utilice un servidor OPC, por ejemplo RSLinx de Allen-Bradley, SNMP de Siemens o COMMGR para Delta Electronics.

4.3. Protocolo de comunicación estudiante-aplicaciones-controlador

EmulSis está conformado por diversas aplicaciones que comparten datos entre sí, donde la aplicación principal es el software EmulSis en sí mismo, por lo cual ha sido necesario encontrar un equilibrio justo entre todas las interacciones que lleva a cabo el software para que el estudiante no detecte de ninguna forma algún tipo de comportamiento anormal al emular un proceso.

En primer lugar, la interfaz de EmulSis como se muestra en la Figura 4.3, cuenta con una apariencia familiar con una presentación tipo *Ribbon*, que se distribuye como sigue:

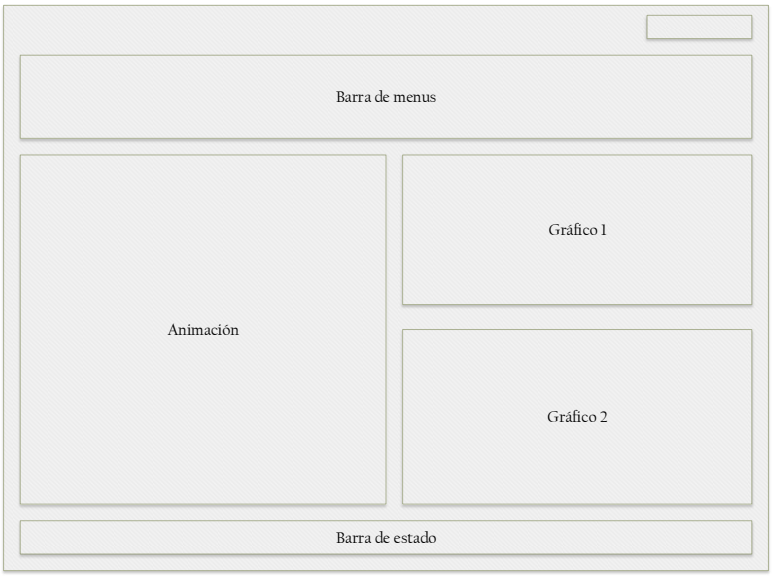

Figura 4.3. Distribución de ventana de EmulSis

Las secciones mostradas son recurrentes en casi cualquier software sin importar cual sea su utilidad, con el fin de no tener una interfaz complicada o difícil de interpretar, por el contrario se busca simplicidad pero al mismo tiempo contener los elementos necesarios de una manera accesible, aprovechando al máximo el espacio en pantalla:

Barra de menus En esta sección se encuentran todas las funciones del software, la configuración y elección del sistema, la comunicación con la herramienta de control a utilizar, la configuración de los gráficos y la comunicación en red con el profesor para mejorar la comunicación con el alumno.

Animación Aquí se observa la representación gráfica del sistema seleccionado para controlar, en este caso se presenta el sistema del tanque atmosférico, una ventaja de esta es que se puede mostrar en pantalla completa con el fin de observar con mayor detalle los resultados de la solución de control aplicada utilizando el menú *Ventana > Agrandar animación*

Gráficos 1 y 2 Muestra la evolución de los estados del modelo o cualquier otra variables que se encuentre en el proceso, en este caso de manera predeterminada la altura de la columna de agua y el flujo de salida del sistema, mismas que pueden ser editadas y exportadas si se requiere para elaborar un reporte por medio de la opción *Gráficas > Tamaño de imagen > Exportar*.

En esta parte de la ventana se muestra otro panel que contiene opciones adicionales según la acción seleccionada en la barra de menus, como las tablas de los resultados de los cálculos, la información de la configuración de *Tags* y direcciones de controlador, la comunicación con el servidor de OPC, y las configuraciones de la comunicación entre el estudiante y el profesor.

Barra de estado Se muestran diversos estados, del programa así como el tiempo de emulación actual.

Así mismo, la sección de animación puede considerarse como un aplicación huésped dentro del programa principal, pues ésta es un desarrollo importante para el funcionamiento del programa principal que en si misma tiene un cierto nivel de complejidad que tuvo que mantenerse a la par de todo el diseño, no solo porque genera las respuestas de los actuadores y transmisores del sistema, también lleva a cabo un pequeño protocolo de comunicación que sirve para intercambiar datos con la aplicación principal al compartir datos con el componente de Flash Player.

Este protocolo se utiliza con dos propósitos, el primero de ellos es el de configurar las señales de los componentes del sistema, los transmisores y actuadores con el fin de asignarles una dirección la cual se utilizará para poder controlar el sistema tal como se hace

en realidad, el alumno no puede obtener datos tal como lo hace un transmisor real y escribir datos como se hace con un actuador, como puede ser una válvula modulante o un variador de frecuencia, por lo que esto debe ser implementado dentro de la aplicación huésped de la animación:

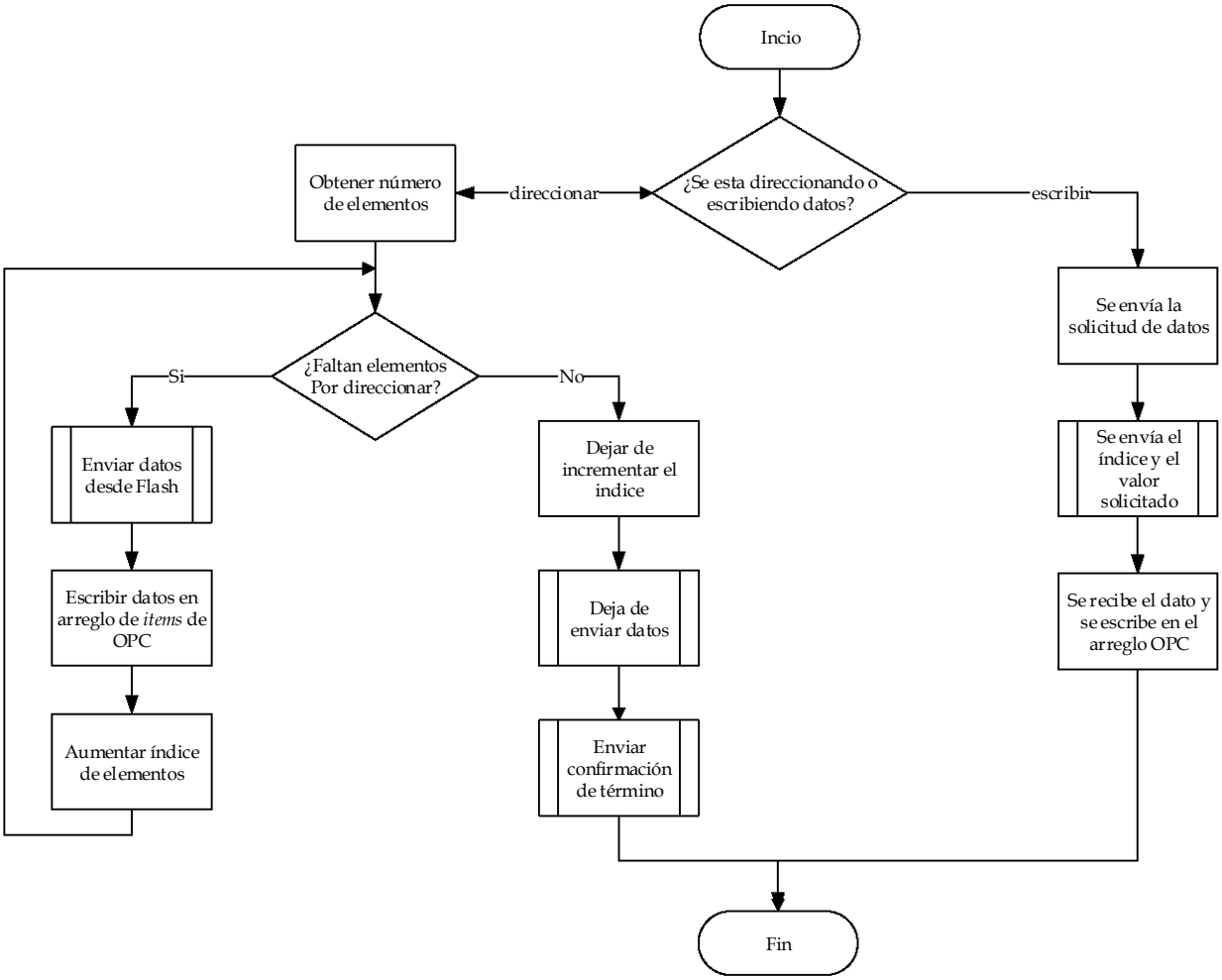

Figura 4.4. Flujo de datos Delphi - Flash

Después la comunicación entre estudiante y el profesor, lo cual se logra por medio de una red local, por ello al iniciar el software, se busca la dirección IP del equipo que ejecuta el programa, con el fin de establecer comunicación con uno o mas usuario, para esto el único requerimiento es que todos los equipos se encuentren en la misma red con el dominio IP.

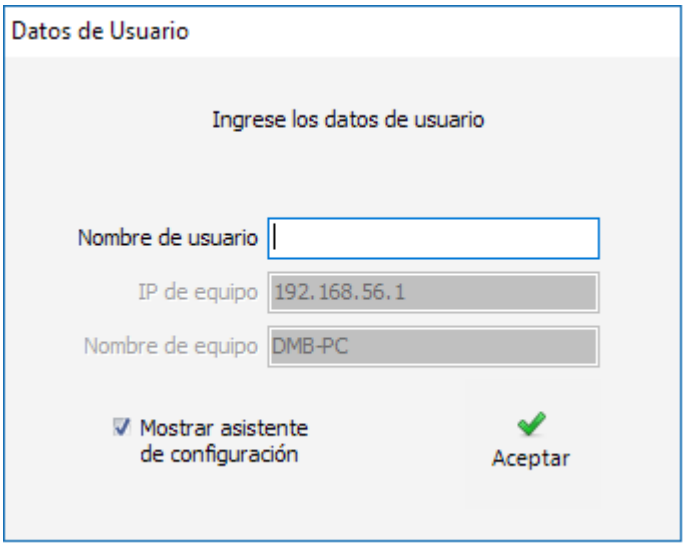

Figura 4.5. Ventana de identificación de equipo en red

Como parte de la implementación para las actualizaciones futuras, únicamente se incluyó la base de la comunicación, como se ilustra en la Figura 4.5, es decir la comunicación se implementó, no así la variedad de utilidades que se pueden derivar de esta, solo se puede transmitir texto cuando existe una aplicación que ejecute un servidor para recibir los mensajes.

Esta parte tiene un gran potencial en su desarrollo como parte de un trabajo futuro, ya que esta comunicación inicial que conecta al profesor y al estudiante fue integrada con la cualidad de comprobar y reducir los efectos de esta transmisión de datos durante la emulación del sistema, lo que quiere decir que se pueden enviar mensajes de texto y datos de la ejecución de control en tiempo real, con un mínimo de afectaciones a la velocidad de resolución de sistema dinámico, ya que no es posible eliminar el retraso en la tarea principal por tratarse de un proceso central, sin embargo para el propósito del programa y su naturaleza didáctica es útil y posee en si misma una gran área de oportunidad para continuar con este desarrollo.

4.4. Implementación comunicación OPC

Como ya se ha dicho, la comunicación entre el software y cualquier medio de control industrial se realiza por medio del protocolo OPC, que se encuentre operando dentro del equipo que ejecute las aplicaciones EmulSis y servidor.

El funcionamiento del protocolo dentro del sistema fue programado, para que se generen las configuraciones que necesita para intercambiar la información se envía y se recibe.

El software, al iniciar su ejecución, busca dentro del equipo anfitrión los servidores OPC disponibles, si encuentra uno o más de ellos se enlistan y los coloca como elegibles tal como muestra la Figura 4.6, posteriormente, el usuario debe conectarse al servidor elegido para comunicar al controlador industrial con el software EmulSis, una vez que se hace una conexión exitosa se muestran los detalles sobre la conexión en el panel lateral.

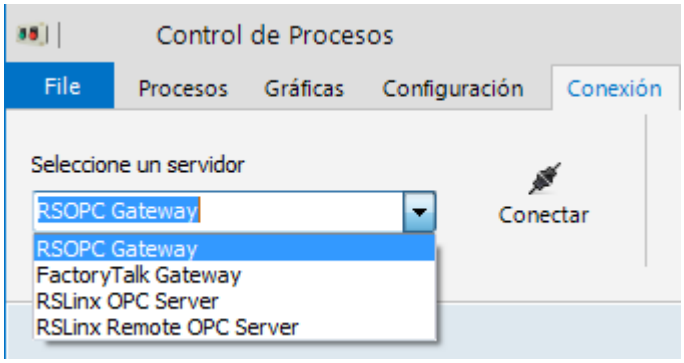

Figura 4.6. Selección de Servidor OPC

En este momento inicia la auto-configuración de parámetros dentro de EmulSis, el cual se identifica como un cliente del servidor OPC y solicita la lista de los Tópicos registrados en el servidor, y a su vez se crea un Grupo donde se almacenan los datos, esto, es suficiente para que el software solicite datos y pueda escriba las variables especificas ofrecidas por el servidor en el Tópico seleccionado. Realizar todo esto se lleva a acabo de manera transparente al usuario, pues al seleccionar el sistema el software crea objetos llamados *Grupos* y *Subgrupos*, que funcionan como arreglos de dimensiones multiples para gestionar los datos dentro del software. Para que este proceso se realice con rapidez ya que el software solo crea un grupo y un subgrupo al guardar los elementos.

4.5. Implementación del motor de solución de sistemas dinámicos

La implementación del motor de solución de sistemas de EDO es la parte medular del proyecto, pues es la característica más importante de EmulSis la cual le permite emular los procesos. Con el fin de mejorar la eficiencia del programa al momento de su ejecución se buscaron varios aspectos orientados a la optimización de este proceso durante su desarrollo.

Esta sección del proyecto tiene gran relevancia pues no solo nos da oportunidad de continuar aumentando el número de procesos que el software emula, también lo hace de manera automática para el desarrollador, quien únicamente necesita conocer el modelo

representado en EDO para poder integrarlo a la batería de sistemas, aclarando que esto únicamente incluye la parte analítica y no la parte gráfica que tiene que ser desarrollada por otros medios.

La implementación de esta plataforma se basa en una programación basada en tres tipos de objetos, que son tipos de datos predefinidos para que el software ejecute la solución del sistema sin necesidad de agregar líneas extra en un desarrollo futuro, los tipos de datos son los siguientes:

TState Es el tipo de dato que representa a una ecuación de estado del sistema, el cual a su vez esta definido como una *función* de dos parámetros, el primero es un arreglo de valores de dimensión igual al número de estado que contiene a los valores actuales de los estados, el segundo es un arreglo de dimensión igual al número de entradas al sistema, que devuelve el valor del estado instantáneo.

```
Type
TState = function(s: array of Double; inp: Double): Double;
```

TSis Es un arreglo de N TState que representa al sistema de ecuaciones de estado, y esta definido como:

```
Type
TSis = Array of TState;
```

Se notará que se hace énfasis en las palabras función y procedimiento, esto es porque se hace una diferencia en la ejecución de cada una de estas acciones, mientras que en una función se necesita devolver un valor del tipo especificado al final de su ejecución, en el procedimiento no es necesario y solo lleva a cabo las acciones que se definen dentro de su estructura.

Entonces como se puede ver la implementación de un nuevo sistema dentro del software es similar a como se hace una solución analítica, primero se obtienen las ecuaciones diferenciales, después se genera el sistema de ecuaciones y se resuelve, en este cadso numericamente.

Una vez que se ha completado la declaración de las ecuaciones de estado y agregadas a un sistema, se puede elegir como es que se requiere que se soluciona el sistema, por lo que el método de solución se convierte para nuestro caso en otro tipo de dato llamado **TSolver**, el cual puede ser llamado *eSolver*, *ieSolver* y *rkSolver*, y está definido como sigue:

```
Type
TSolver = procedure(sis: TSis; u: array of Double);
```

Que como puede observarse es un *procedimiento*, el cual tiene como parámetros a un sistema del tipo **TSis**, y un arreglo de entradas que representan el vector de entradas **u**.

El proceso de incluir un nuevo sistema puede parecer complicado por el tipo de declaraciones que se llevan a cabo, se puede entender mejor si se comprende que el tipo de dato únicamente requiere que se declare una función como en cualquier lenguaje de programación, en este caso de la siguiente manera, por ejemplo para un sistema de un solo estado:

```
//Después de definir los tipos de datos
interface
function dx0(x: array of Double; inp: Double): Double;

//Defición de la ecuación de estado
implementation

function dx0(x: array of Double; inp: Double): Double;
  begin
  // Aquí la definición de la ecuación de estado
  end;
```

Hasta ahora se ha definido únicamente la función de estado, la cual está representada por la función $dx0$ se hará la declaración del sistema completo y su solución como sigue:

```
//Decalaración de variables
var
dxh0 : TState;            //Definimos a dxh0 como un TState
sistema : Tsis;           //Definimos el sistema
u : Array of double       // Definir un vector de entradas
solver : TSolver;         //Definir variable para llamar al método de solución

  // Al seleccionar el tipo de sistema se arma el modelo según las variables y las funciones antes dadas

  procedure TTMSForm1.TanqueButtonClick(Sender: TObject);
  begin
  dxh0 := dx0;            //Recordando que TState es una función con las características de dx0

  setlenght(sistema,1);   //Establecer el tamaño o número de estados

  setlenght(u,1);         //Establecer tamaño de vector de entradas

  sistema[0] := dxh0;     //Se indica que la ecuacion de estado pertenece a "sistema"

  //Definir el método de solución del sistema
  solver := rkSolver;

  end
```

Lo anterior nos indica que al hacer click en el botón de nombre *"TanqueButton"* se alista el sistema para ser resuelto, ahora solo basta llamar la siguiente secuencia :

```
//Llamada a resolver el sistema en el tiempo $t$
rkSolver(sistema,u);
```

Estas últimas líneas pueden ejecutarse bajo cualquier evento que se desee controle la resolución del sistema, y para los objetivos de este proyecto es necesario que esta solución se haga en tiempo real por lo que es necesario contar con una base de tiempo fiable y constante para que los cálculos se realicen de esta manera, este comportamiento se describe en la sección siguiente.

4.6. Emulación del proceso físico

Uno de los objetivos del proyecto es que la emulación fuera realista bajo cualquier cambio de condiciones y más importante aún a la solución de control propuesta e implementada por el estudiante, y así proveerlo de un ambiente seguro y didáctico, por estas características se le ha llamado emulación al proceso de representar el sistema dinámico, ya que no solo se representa el estado físico del sistema utilizando la representación de estados y se presenta gráficamente, también es posible detener la emulación y reiniciar el sistema a sus condiciones iniciales de manera instantánea.

Obviamente el comportamiento del un sistema físico real es imposible de lograr de esta manera y la manera de realizar la emulación puede resultar confusa, por lo que la diferencia del flujo de datos de control entre la aplicación en comparación de un sistema de control implementado en la industria, se muestra en la Figura 4.7.

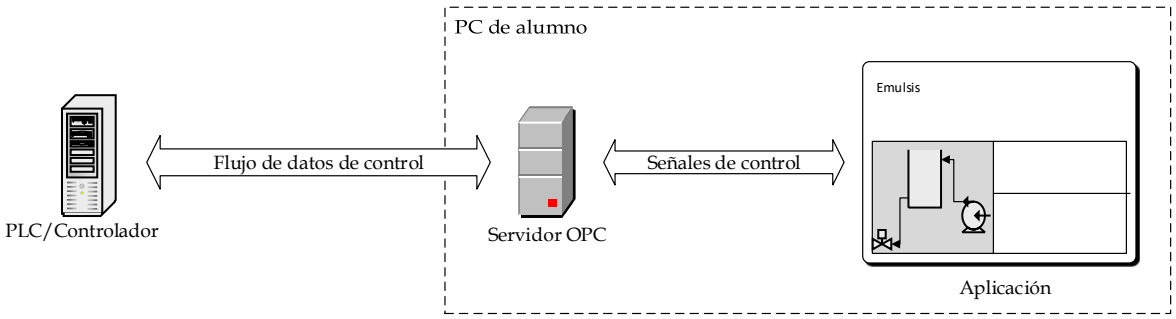

Figura 4.7. Flujo de datos de control de EmulSis

En comparación con un proceso de control real, del sistema:

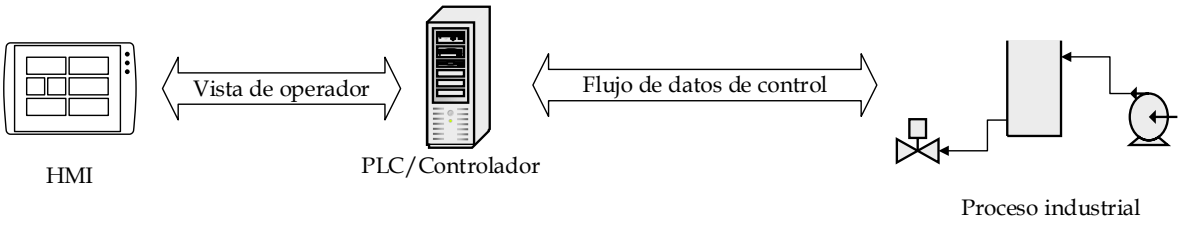

Figura 4.8. Proceso de control real

Como se observa en la Figura 4.8 esta presenta un flujo de datos más ligero, que es lo que normalmente se utiliza en un lazo de control industrial, también se muestra una Interfaz Humano-Máquina (HMI por las siglas en inglés), esto no aparece en la Figura 4.7 ya que como el ambiente del software es visual se representa el sistema, en la realidad puede que el operador no tenga a la vista al proceso y necesite de esta interfaz para saber que es lo que ocurre en él.

Muchas veces por el lado práctico, algunas otras por el riesgo o la lejanía entre áreas de monitoreo es necesario utilizar este tipo de equipos, los cuales no entran en ningún apartado de este desarrollo como objetivo ya que la representación visual es suficiente, sin embargo como estas interfaces están normalmente conectadas al controlador mediante una red Ethernet o un protocolo serial, también puede hacerse el ejercicio de conectar y programar una de estas interfaces al arreglo y monitorizar el proceso emulado. Lo cual resulta práctico, ya que puede considerarse a EMULSIS como un proceso virtual independiente de cualquier relación con algún controlador especifico, y con respuesta natural ante cualquiera de estos con los que se haga la práctica o capacitación. Ya que el controlador solo esta encargado de llevar a cabo la solución del sistema no interviene de ninguna manera en el proceso.

4.7. Ejecución en tiempo real del proceso

Después de tener la capacidad de resolver cualquier sistema, sin importar la naturaleza o la complejidad de este es necesario hacerlo en tiempo real, para que el alumno tenga la sensación de estar observando el sistema en vivo, y poder ver los resultados de su solución de control de manera fluida y realista.

En este punto es necesario que el tiempo que tarda el software en dejar pasar determinado periodo de tiempo no se retrase con los procesos alternos que se llevan acabo en el equipo que ejecuta EmulSis, o con los propios de él, ya que no solo se están resolviendo los modelos matemáticos, también esta guardando los datos, se actualiza la animación y se grafícan los resultados. Para ello se las acciones que se llevan a cabo tienen prioridades diferentes que se respetan cada vez que se completa un ciclo dentro del temporizador.

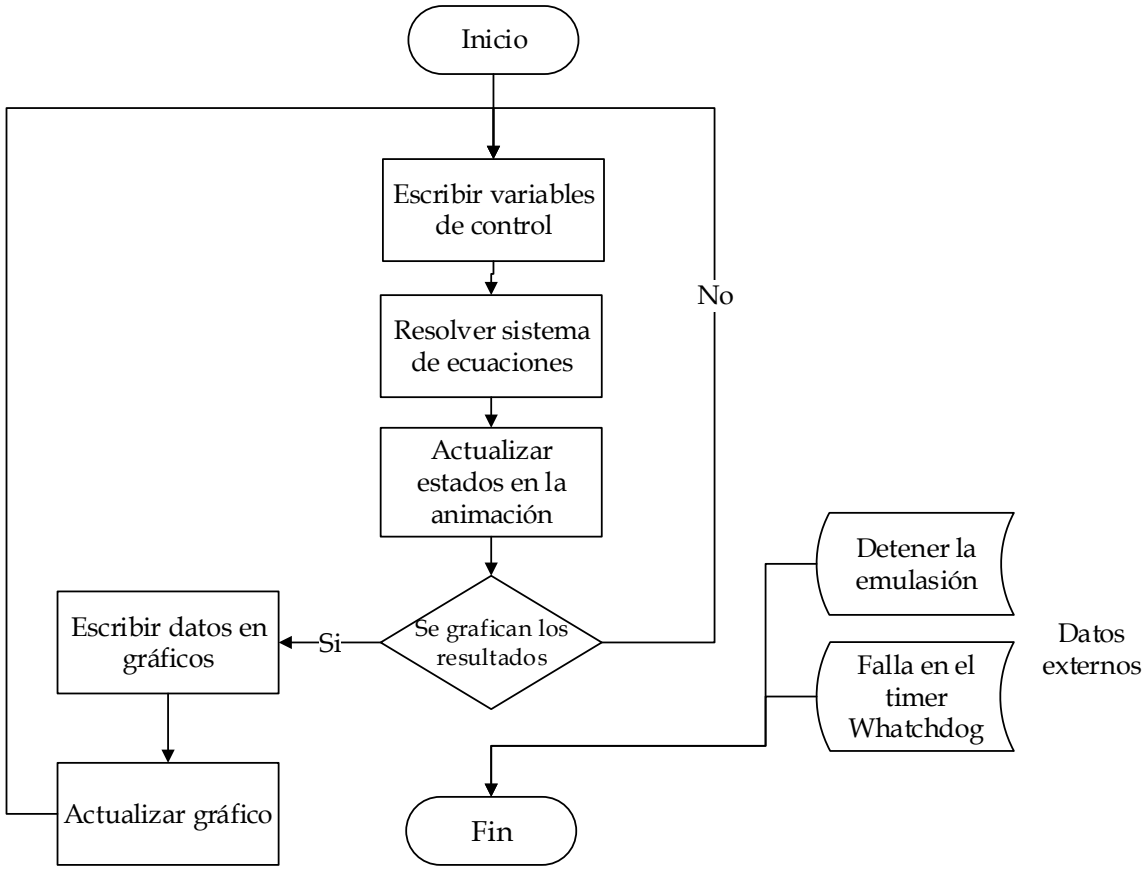

Figura 4.9. Procesos de temporizador en tiempo real

Como se observa en la Figura 4.9, los procesos que se llevan a cabo son únicamente lecturas y escrituras de datos, sin embargo la complejidad de este proceso está en que todos estas instrucciones se llevan a cabo cada vez que el temporizador termina una cuenta. Como se mencionó el hecho de utilizar la plataforma de desarrollo de Delphi es la de poder gestionar los recursos del sistema casi sin restricciones, y en este caso se hace algo a este nivel, ya que el timer utilizado nombrado *HighPrecisionTimer*, el cual calcula la velocidad del procesador del equipo y ejecuta una interrupción al mismo para poder determinar que el periodo de tiempo que ha transcurrido desde que se habilito el timer, el cual recibe el tiempo en orden de la frecuencia de repetición del evento, es decir, que si damos una frecuencia de $1kHz$ los procesos del diagrama de la Figura 4.9 se repetirá 1000 veces en un segundo, idealmente, esto ya que el tiempo que se toma en realizar todas estas acciones también debe ser considerado y el hecho de que las acciones del timer sean limitadas responden a esta situación.

También se puede observar una rutina de error en el timer, esto es porque el componente tiene la bondad de revisar si es que se ha llegado al final del procedimiento expresado en el evento del mismo y verificar si terminó correctamente, si no es así este sale del ciclo del temporizador y reinicia el conteo del periodo.

El hecho de que este proceso se realice de manera rápida, puede acarrear errores en el momento de ejecución, ya que a pesar de que se emplean acciones sencillas dentro del timer, se utiliza mucho acceso a memoria, que puede causar un efecto botella en algunos equipos, para condensar esta situación se tienen diferentes opciones para que el usuario manipule el comportamiento de la emulación con el fin de reducir los comportamientos no deseados, algunas de estas soluciones son: la posibilidad de escoger la frecuencia de operación de estos procesos, elegir el número de muestras en los gráficos ya que estos consumen gran cantidad de recursos al ser mostrados en pantalla, o así mismos suprimir completamente la representación en el tiempo de los estados.

Capítulo 5

Evaluación del sistema

Antes de incluir el software EMULSIS en un plan de estudio como herramienta de apoyo, fue necesario realizar diferentes pruebas con el fin de descartar problemas o comportamientos erróneos durante su funcionamiento por ejemplo, ejecutándolo en equipos de poco rendimiento, realizando diversas acciones durante la emulación e identificando cualquier situación que usuario pudiera experimentar durante su uso.

Lo anterior se llevó a cabo realizando el planteamiento de una solución de control y realizando diferentes ciclos de emulación tal como el usuario final lo haría, por tanto el proceso de evaluación comienza al iniciar el programa y configurandolo para la emulación, como se describirá más adelante.

Es fácil notar que el programa está enfocado en fomentar en el estudiante buenas prácticas de diseño e implementación de un sistema de control, pues como se observa en la Figura 5.1, el usuario debe completar una tabla de asignación de variables como parte de la configuración del sistema, lo cual es un proceso documental requerido en la industria.

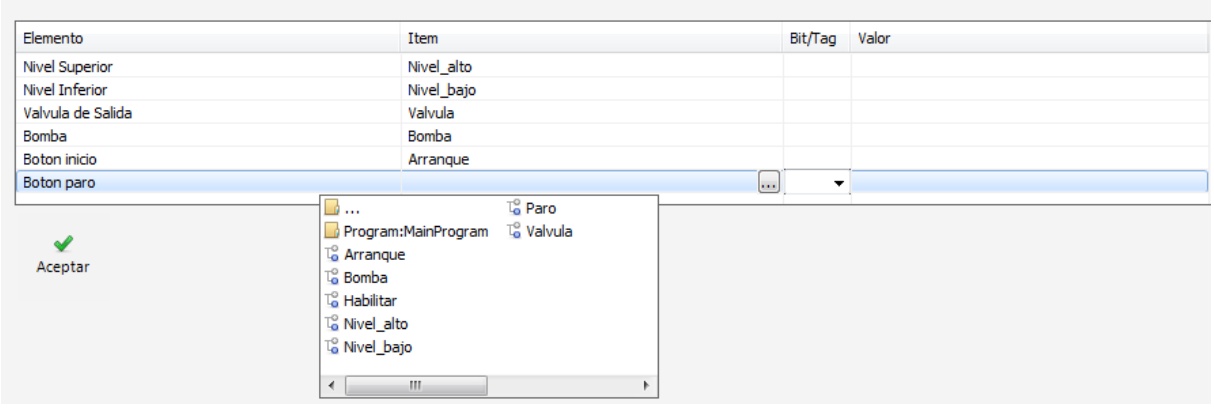

Figura 5.1. Tabla de asignación de variables del software EMULSIS

Además de ejecutar el programa con diferentes equipos de computo para determinar su comportamiento con distintos niveles de *hardware*, los procesos de emulación y control con distintos controladores industriales, incluyendo controladores virtuales con el fin de poder exponer al software a la mayor exigencia posible.

5.1. Condiciones de evaluación

Los distintos ciclos de emulación y control del proceso del tanque atmosférico y las pruebas iniciales de la comunicación por medio de OPC se llevaron a cabo con un PAC marca Allen-Bradley de la serie ControlLogix 5571 (Figura 5.2), pues es un equipo de alto rendimiento que permitió evaluar a EMULSIS de una manera concluyente, pues ofrece las características de un controlador de última generación.

Figura 5.2. ControlLogix 5571

También se tuvo acceso a distintos equipos, con el fin de evaluar el software con diferentes controladores y así poder comprobar que la solución de control se puede llevar a cabo sin

importar el controlador utilizado, la lista de estos es la siguiente:

- Allen-Bradley SLC 500
- Allen-Bradley Micrologix 1400
- Allen-Bradley CompactLogix L24ER
- Delta-Electronics DVP-SS2
- Allen-Bradley ControlLogix 5571
- Allen-Bradley EMU-500 [1]
- Softlogix Virtual Chassis 5800 [1]

Cabe señalar que en la lista anterior es predominante la marca de Allen-Bradley, lo cual responde al hecho de que es uno de los fabricantes más utilizados en la industria sin importar del tamaño del controlador o la aplicación que se le de, por ello fue más recurrente tener acceso a estos al llevar a cabo las pruebas de control.

Lo anterior no significa que el uso del software se limita al uso de controladores Allen-Bradley, pues según lo mencionado en el Capítulo 2, OPC es un protocolo diseñado y estandarizado para intercambiar datos entre aplicaciones industriales, por tal motivo no existe ningún problema en comunicar EMULSIS con servidores de otros fabricantes como OMRON, Siemens, ABB, etc.

En relación con el rendimiento de EMULSIS durante la ejecución de los procesos que se llevan a cabo durante la emulación era necesario comprobar el impacto que tienen en la computadora, pues los métodos de solución numérica exigen la realización de grandes cantidades de cálculos por segundo, además de llevar a cabo otro tipo de acciones como actualizar la animación, generar los gráficos y obtener constantemente los estados de las señales para enviarlos al controlador.

5.2. Resultados

Los resultados obtenidos después de las evaluaciones mencionadas, no solo consideran si el controlador fue capaz de llevar a cabo la automatización del proceso, pues también se analizó cada aspecto que EMULSIS genera como resultado de su uso, estos pueden ser de cualquier tipo como la respuesta en la animación, la generación de gráficos para reporte, las distintas opciones para optimizar recursos e incluso saber si la interfaz es

[1] Procesadores emulados

adecuada para la operación del programa, de manera más crítica se observaron los las tareas más importantes como la comunicación con el controlador y la ejecución del motor de soluciones en tiempo real.

Al comenzar con la ejecución se muestra la pantalla de inicio del software EmulSis expuesta en la Figura 5.3 demostrando que es un producto terminado y completamente operativo donde hasta el mínimo detallo ha sido considerado para mejorar la experiencia del usuario, además tiene el propósito de crear una identidad para el software con el fin de continuar con su desarrollo siguiendo las mismas motivaciones y expandiendo sus objetivos haciendo uso de las bases sentadas en esta primera versión.

Figura 5.3. Pantalla de presentación Emulsis

Después de iniciar EmulSis, se observa la pantalla principal del programa, misma que se ha descrito en la Figura 4.3, la cual tiene diferentes opciones para poder manipular la vista y el comportamiento del software según la necesidad del usuario, además está diseñada para tener un aspecto familiar e intuitivo con el objetivo de mejorar la navegación entre los diferentes menus disponibles.

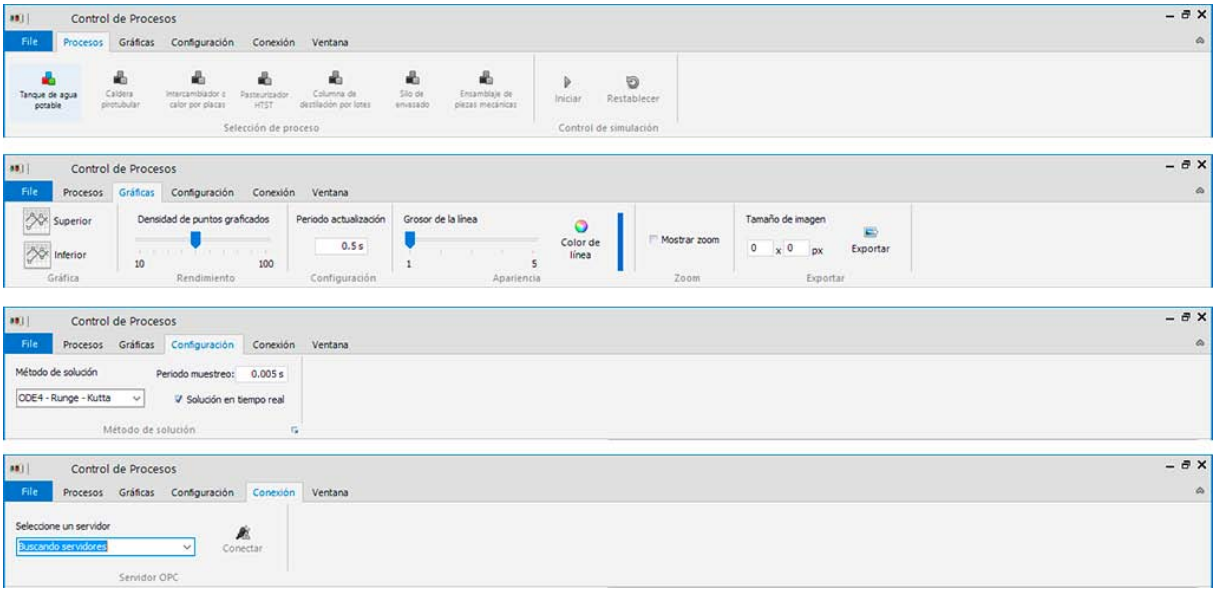

Figura 5.4. Menús de usuario disponibles por medio de la interfaz tipo *Ribbon*

En este punto el programa debe ser configurado con los parámetros mínimos para realizar un proceso de emulación como, la selección del proceso y del servidor OPC, el direccionamiento de los componentes del sistema de control y si es necesario las opciones gráficas para mejorar el rendimiento. Lo cual, gracias a la configuración de la interfaz de usuario, se realiza de manera sencilla y rápida por lo que no es mayor problema aprender a configurar el software en poco tiempo.

Para la evaluación de EmulSis se utilizó un problema común en la industria, el cual puede parecer trivial, sin embargo aún con los elementos presentes pueden aplicarse numerosas soluciones de control al proceso. El problema planteado es el siguiente:

Un tanque atmosférico con dos switches de nivel instalados uno que se considera el nivel alto y otro el nivel más bajo de líquido permitido, el proceso requiere que se descargue la misma cantidad de agua cada vez, por lo que al llenar el tanque hasta el nivel alto con ayuda de la bomba, se debe accionar la válvula de vaciado únicamente hasta el nivel bajo para posteriormente repetir el ciclo. Al inicio el tanque esta completamente vacío y el operador debe iniciar la secuencia manualmente, de igual manera debe ser capaz de detenerla en el momento que lo requiera y volver a activarla de la misma forma.

En base a lo anterior, se planea la estrategia de control, para lo cual se genera un programa en *RSLogix 5000* desarrollado en lenguaje de escalera o *ladder*, para llevar a cabo la secuencia deseada, el cual se ilustra en la Figura 5.5

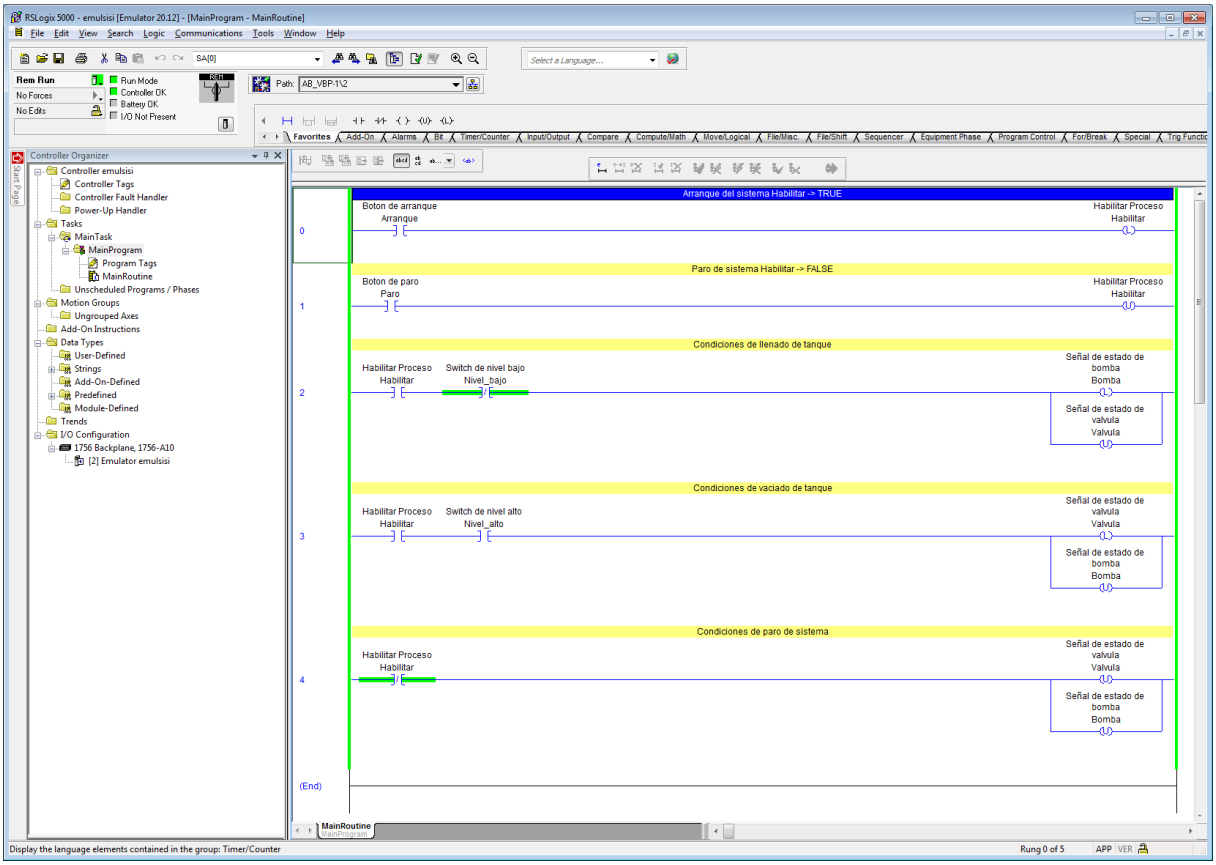

Figura 5.5. Programa de control visto en la estación de trabajo en línea con el controlador

Al configurar EmulSis para comunicarse con el servidor OPC de RSLinx ya no es necesario realizar ninguna acción más que presionar el botón de arranque, tal como lo haría un operador en el proceso real, para que el sistema inicie su funcionamiento.

Un punto importante es que en la industria normalmente no es posible observar a simple vista el resultado de la solución de control y por ello tienen que interpretarse los datos arrojados por los transmisores de campo, en este caso podemos observar ambas cosas, ya que la función del software es la de familiarizar al estudiante con el proceso y su control automatizado, pero también es una herramienta de apoyo a la enseñanza por lo cual es conveniente que el estudiante vea de primera mano los efectos de su solución.

Por lo que en este caso se puede observar la columna de agua ascendiendo por el tanque y además se genera un gráfico que representa el mismo resultado (Figura 5.6), lo cual es lo que observaría un operador en la pantalla de control.

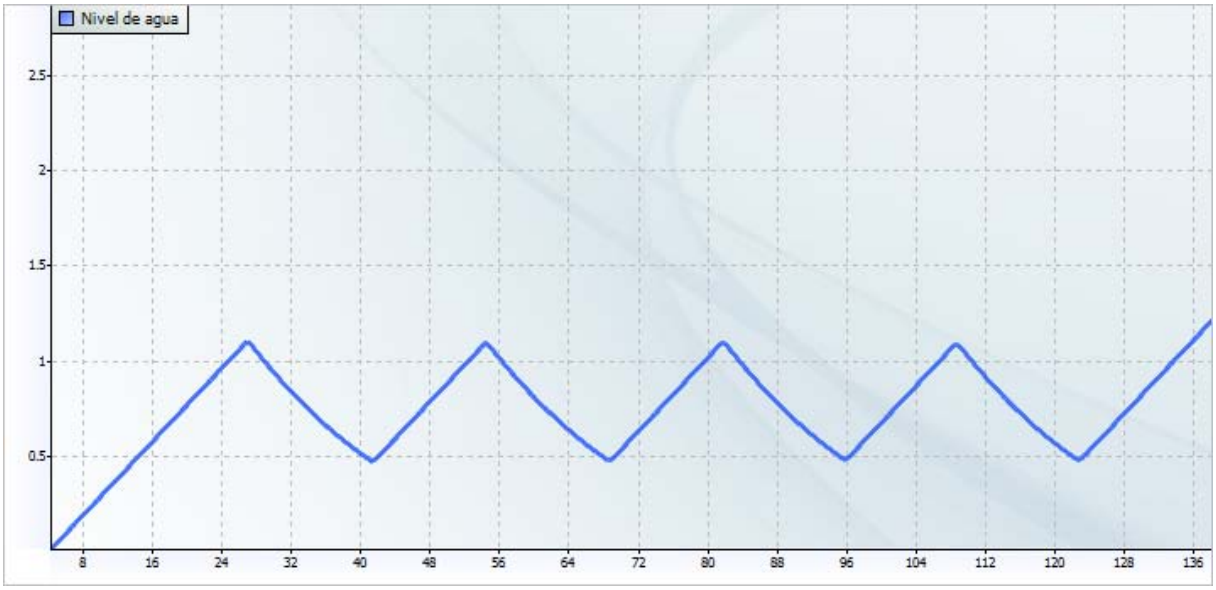

Figura 5.6. Tendencia representativa el nivel de la columna de agua

Adicionalmente se puede generar el gráfico de una variable adicional durante el mismo proceso de emulación, de manera predeterminada se visualiza la tendencia del flujo de salida de agua mostrado por la Figura 5.7, donde indistintamente se puede seleccionar y mostrar los valores de cualquier estado gracias al motor de solución de sistema de EDO el cual resuelve y guarda todos los datos resultantes de la emulación.

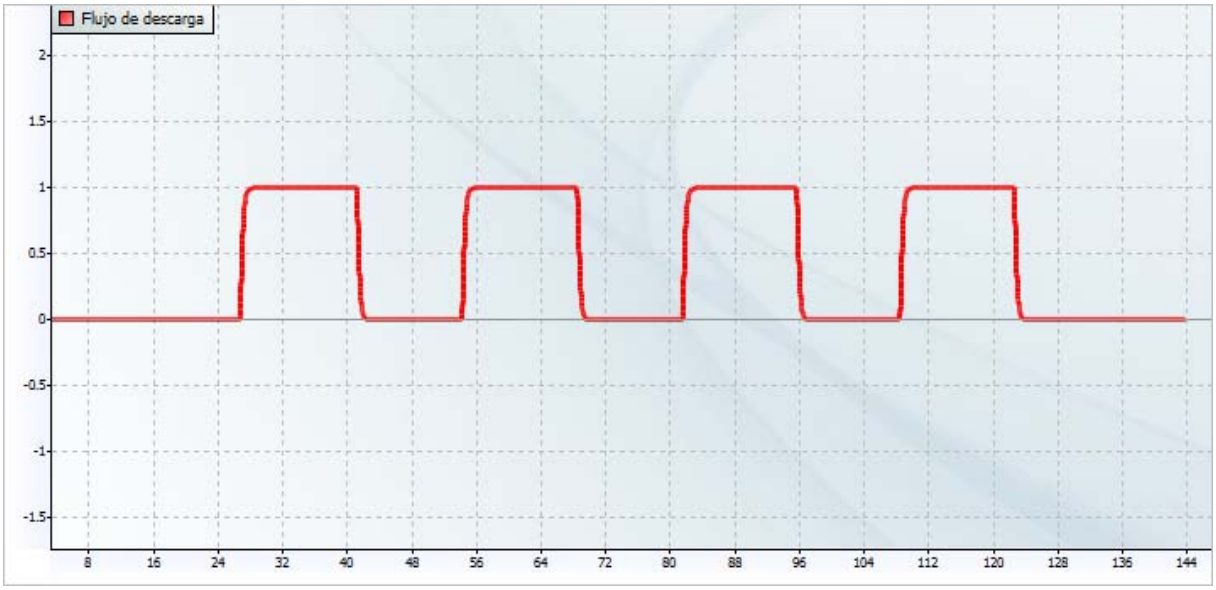

Figura 5.7. Tendencia de flujo de salida

Como parte de las opciones para optimizar el rendimiento del software es posible prescin-

dir de la visualización de los gráficos de tendencia y solo visualizar el proceso animado, para mayor comodidad en modo de pantalla completa, donde el estudiante puede interactuar con los elementos presentados en esta.

Figura 5.8. Vista de animación de proceso en modo de pantalla completa

En base a los resultados mostrados en las gráficas anteriores es posible determinar que las pruebas han resultado ser exitosas, pues el control se realizó sin problemas y justo como fue planeado.

El equipo principal con el que se llevaron a cabo las pruebas tiene las siguientes características técnicas:

- Dell Inspiron i5555
- Sistema operativo Windows 7 Professional 64 Bit
- Procesador AMD A10-8700P Raedon R6, 10 Compute Cores 4C+6G 1.8Ghz
- Memoria RAM 12GB

Con lo cual se ha podido ejecutar el software, utilizando un valor de Δt para los cálculos de 5[ms] mediante el motor de solución, tomando muestras para los gráficos cada segundo y mostrando la animación en alta calidad, que son los valores pre-determinados, por tal motivo se puede determinar que no es necesario tener una gran capacidad de procesamiento para ejecutar el programa bajo estas condiciones, por tal motivo el producto, en su fase inicial se puede considerar como funcional y completo.

5.3. Discusión

Después de realizar las pruebas de EmulSis en conjunto con un controlador industrial, es posible realizar una revisión no solo de los resultados obtenidos, también se puede hacer un análisis de la etapa de diseño, pues el producto de este se ve reflejado en los resultados finales.

Desde el inicio, cuando se plantearon las motivaciones del proyecto, se determinó que el software sería una herramienta de apoyo al desarrollo de las competencias profesionales y al incremento de los niveles cognitivos con el fin de promover el entendimiento de los procesos industriales durante la formación de un profesional en ingeniería de control automático.

En muchos casos dentro de la industria no se ha dado un cambio total a las nuevas tecnologías de automatización, así pues un profesional que es capaz de implementar soluciones con estas, tiene la capacidad de ser aún más efectivo si durante su desarrollo ha tenido acercamientos a los procesos industriales, haciéndolo capaz no solo de identificar y describir un proceso, también tendrá la habilidad de diseñarlos, categorizarlos y resolverlos aunque no esté familiarizado con alguno en particular.

A su vez siempre existe una base de conocimientos teóricos necesarios para describir cualquier sistema que se pueda encontrarse en la industria de manera analítica, pues al final, el comportamiento dinámico debe ser considerado como punto de partida para optar por una u otra solución de control. Por esto se aborda en el Capítulo 3 la integración del modelo matemático del, proceso industrial seleccionado, pues sirve como plataforma de prueba para poder implementar cualquier otro proceso sin importar la naturaleza de su sistema dinámico.

En relación a lo anterior se eligieron tres métodos numéricos para llevar a cabo la solución de los sistemas de EDO los cuales cuentan con las características necesarias para poder llevar a cabo la emulación en tiempo real de los procesos, se eligieron, ya que a partir de ellos se hace factible la implementación, pues la complejidad y el número de cálculos por

segundo se volvió un apartado importante durante el desarrollo.

Considerando lo anterior como el apartado más importante, se desarrollo el software desde las primeras etapas, pues se visualizó como la integración de un sistema modular, lo cual se llevó a cabo fragmentando los objetivos en tareas especificas y así poder alcanzar el resultado final, que es un software funcional pero teniendo la posibilidad de ser actualizado y mejorado constantemente con el fin de cubrir constantemente las necesidades formativas, así como elevar gradualmente los niveles cognitivos del estudiante.

Por último se verifico la calidad del desarrollo y si su objetivo fue alcanzado, por medio de pruebas en conjunto con distintos controladores industriales, usando los disponibles en el Laboratorio de Automatización y algunos otros a los que se tuvo acceso, como caso particular el controlador Delta $DVP-14SS2$, es un ejemplo de la gran cantidad de fabricantes que se pueden encontrar en el mercado y sin importar cual sea EmulSis está preparado para preparado para comunicarse con ellos debido a la integración de protocolos industriales utilizados lo que da como resultado fluidez y estabilidad en la emulación, lo cual se verificó mediante la solución de un problema industrial real emulado y resuelto por medio de un controlador industrial, teniendo resultados favorables bajo diversas condiciones de operación.

```
procedure eSolver(sis: TSis; u: array of Double);
var
  xi: array of Double;
  i: Integer;
begin
  setlength(xi, nEstados);
  setlength(sis, nEstados);

  for i := 0 to nEstados - 1 do
  begin
    xi[i] := x[i][t - 1];
  end;

  for i := 0 to nEstados - 1 do
  begin
    x[i][t] := xi[i] + dt * eval(sis[i], xi, u[i]);
  end;
end;

procedure ieSolver(sis: TSis; u: array of Double);
var
  k1, k2, xh1, xh2, xi: array of Double;
  i: Integer;
begin
  setlength(xi, nEstados);
  setlength(k1, nEstados);
  setlength(k2, nEstados);
  setlength(xh1, nEstados);
  setlength(xh2, nEstados);
  setlength(sis, nEstados);
  for i := 0 to nEstados - 1 do
  begin
    xi[i] := x[i][t - 1];
  end;

  for i := 0 to nEstados - 1 do
  begin
    k1[i] := eval(sis[i], xi, u[i]);
  end;

  for i := 0 to nEstados - 1 do
  begin
    xh1[i] := xi[i] + k1[i] * dt;
  end;

  for i := 0 to nEstados - 1 do
  begin
    k2[i] := eval(sis[i], xh1, u[i]);
  end;
  for i := 0 to nEstados - 1 do
  begin
    x[i][t] := xi[i] + (dt / 2) * (k1[i] + k2[i]);
  end;
end;

procedure rkSolver(sis: TSis; u: array of Double);
var
  k1, k2, k3, k4, xi, h1, h2, h3, h4: array of Double;
  i: Integer;
begin
  setlength(xi, nEstados);
  setlength(k1, nEstados);
  setlength(k2, nEstados);
  setlength(k3, nEstados);
  setlength(k4, nEstados);
  setlength(h1, nEstados);
  setlength(h2, nEstados);
  setlength(h3, nEstados);
  setlength(h4, nEstados);
  setlength(sis, nEstados);

  for i := 0 to nEstados - 1 do
  begin
    xi[i] := x[i][t - 1];
  end;

  for i := 0 to nEstados - 1 do
  begin
    k1[i] := eval(sis[i], xi, u[i]);
  end;
  for i := 0 to nEstados - 1 do
  begin
    h1[i] := xi[i] + (dt / 2) * k1[i];
  end;
  for i := 0 to nEstados - 1 do
  begin
    k2[i] := eval(sis[i], h1, u[i]);
  end;
  for i := 0 to nEstados - 1 do
```

Capítulo 6

Conclusiones

El trabajo presentado se fundamentó en el desarrollo e implementación de una herramienta de apoyo para la práctica y la enseñanza de control automático industrial; encaminado a lograr que los procesos de aprendizaje, realizados en el aula-laboratorio, sean significativos con el uso del software desarrollado.

Este desarrollo fue realizado en varias etapas, se comenzó por una investigación documental de herramientas de software con propósitos similares, se experimentó con cada una de ellas a nivel usuario y esto permitió realizar un análisis de *Benchmark*, para así definir los alcances del software en materia de funcionalidad, pues una parte importante del desarrollo de un software es conocer cuales son las herramientas existentes y que cualidades ofrecen, esto último con el fin de identificar las áreas de oportunidad más destacadas e implementar nuevas funcionalidades que hagan destacable al producto.

La construcción del modelo matemático fue una de las partes más importantes del desarrollo, pues es la que sienta las bases para la implementación técnica del software, ya que de ahí se derivan otras componentes fundamentales para su funcionamiento. El modelo fue generado recopilando e integrando los modelos matemáticos de los diferentes elementos que conforman la dinámica del llenado y el vaciado del tanque atmosférico, el comportamiento de la bomba centrífuga y de la válvula tipo cuchilla.

Para la obtención de las trayectorias de la dinámica del sistema, se implementó un motor de solución de sistemas de EDO, el cual está dotado de la capacidad de elegir entre tres métodos de solución numérica, este algoritmo puede obtener las trayectorias solución de cualquier sistema en esta representación, sin importar el número de ecuaciones o si estas son lineales o no lineales. Esta cualidad, permite a la herramienta de software desarrollada, agregar fácilmente nuevos sistemas dinámicos que representen procesos industriales; basta con incluir las ecuaciones que rigen su comportamiento dinámico y seleccionando el

método de solución.

La interfaz de usuario utilizada para la presentación del programa fue pensada en brindar la mayor facilidad de uso del software, evitando las configuraciones complicadas, haciendo que no se requiera una capacitación prolongada para utilizarlo, buscando administrar el tiempo de una mejor forma, debido a que este es un recurso valioso en el proceso de la enseñanza-aprendizaje.

Finalmente podemos concluir que se desarrolló un software el cual ha superado las evaluaciones satisfactoriamente en base a los objetivos propuestos. Pues se obtuvo una herramienta que ayuda a fortalecer las competencias por medio de su integración a los planes de estudio, para así generar un acercamiento del profesional en ingeniería a los procesos industriales, a la vez que se puede considerar como un producto terminado de alto valor comercial debido a la importancia de la emulación con fines de entrenamiento en la industria. Además dando la posibilidad sumar a más participantes al desarrollo de nuevas funciones que permitan obtener una mejor interacción entre el estudiante y el profesor, por medio de la integración de funciones de comunicación, evaluación y seguimiento, haciendo uso de la base funcional con la que se cuenta actualmente.

Apéndice A

Motor de solución de ecuaciones diferenciales ordinarias

El siguiente código muestra la declaración de objetos *TState* y *TSis*, así como la definición de los algoritmos de solución declarados como *eSolver* para utilizar el Método de Euler, *ieSolver* para el método de Euler mejorado y *rkSolver* para el método de Runge-Kutta de 4to orden.

La integración de un nuevo sistema se debe de llevar a cabo de la siguiente manera:

1. Declarar un arreglo de n estados *TState*
2. Declarar un sistema *TState*, el cual se define como un arreglo o *array* de *TState*
3. Por último ordenar la solución del sistema llamando al método de solución integrando el sistema como un parámetro de entrada, junto con un arreglo u que representa al vector de entradas al sistema.

```
unit Solvers;

interface
//Declación de tipos de datos
type
  TState = function(s: array of Double; u: Double): Double;
  TSis = Array of TState;

//Funciones para resolver ED
//Eval -  ejecuta la función de estado
function eval(f: TState; v: Array of Double; inp: Double): Double;

//Algortimos de solución de parametros Tsis Sistema, u vector de entrada
//Algoritmo Método de Euler
procedure eSolver(sis: TSis; u: array of Double);
//Algoritmo Método de Euler Mejorado
procedure ieSolver(sis: TSis; u: array of Double);
//Algortimo Método de Runge-Kutta
procedure rkSolver(sis: TSis; u: array of Double);

implementation

uses Main;

function eval(f: TState; v: array of Double; inp: Double): Double;
begin
  result := f(v, inp);
end;
```

```
  begin
    h2[i] := xi[i] + (dt / 2) * k2[i];
  end;
  for i := 0 to nEstados - 1 do
  begin
    k3[i] := eval(sis[i], h2, u[i]);
  end;
  for i := 0 to nEstados - 1 do
  begin
    h3[i] := xi[i] + dt * k3[i];
  end;
  for i := 0 to nEstados - 1 do
  begin
    k4[i] := eval(sis[i], h3, u[i]);
  end;
  for i := 0 to nEstados - 1 do
  begin
    x[i][t] := xi[i] + (dt / 6) * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]);
  end;

end;

end.
```

Referencias

[Badia, *et al.*, 1999] Badia, A., Giménez, A., Bellido, S., y Andújar, S. (1999). *Técnicas para la gestión de la calidad: control de la calidad-ISO 9000, gestión por procesos, diagramas de proceso, gestión de la calidad total, benchmarking-reingeniería*. Colección Estado y sociedad. Tecnos. (Citado en página 31.)

[Badiru, 2005] Badiru, A. (2005). *Handbook of Industrial and Systems Engineering*. Systems Innovation Book Series. CRC Press. (Citado en página 12.)

[Beaverstock, *et al.*, 2011] Beaverstock, M., Greenwood, A., Lavery, E., Nordgren, W., y Warr, S. (2011). *Applied Simulation: Modeling and Analysis Using FlexSim*. BookBaby. (Citado en página 6.)

[Blondel, *et al.*, 1913] Blondel, A., Mailloux, C., y Adams, C. (1913). *Synchronous motors and converters: theory and methods of calculation and testing*. McGraw-Hill Book Company. (Citado en página 26.)

[Clemente y Laburu, 2005] Clemente, M. y Laburu, C. (2005). La herramienta de benchmarking: ¿estrategia de imitación o innovación? Memoria del IX Congreso de Ingeniería de Organización Gijón, 8 y 9 de sep 2005. (Citado en página 30.)

[Concannon, *et al.*, 2007] Concannon, K., Elder, M., Hindle, K., Tremble, J., y Tse, S. (2007). *Simulation Modeling with Simul8*. Visual8 Corporation. (Citado en página 6.)

[Creus Solé, 2012] Creus Solé, A. (2012). *Instrumentación Industrial*. Alfaomega Marcombo. (Citado en página 23.)

[D'Arthenay, 2015] D'Arthenay, D. H. (2015). Desarrollo de un simulador de procesos industriales bajo configuración hardware-in-the-loop para la práctica-enseñanza de control lógico y regulatorio mediante plc. Tesis de maestría en ingeniería automatización industrial, Universidad Nacional de Colombia. (Citado en página 4.)

[De Magalhães, 2012] De Magalhães, A. (2012). *Industrial Automation Practices: Specification and Programming of Logic Control Aplications in the ITS PLC Training Environment*. Real Games LDA. (Citado en página 6.)

[Espinosa y Ortega, 1994] Espinosa, G. y Ortega, R. (1994). State observers are unnecessary for induction motor control. *Systems & Control Letters*, 23(5). (Citado en página 26.)

[Ferrer Rojas, 2017] Ferrer Rojas, A. (2017). VirPLC: una metodología para el desarrollo de capacidades, habilidades y autoestima mediante la estimulación de la lógica con una herramienta sencilla, funcional y dinámica. *Education in the Knowledge Society (EKS)*, 18(2). (Citado en página 6.)

[Goodhew, 2010] Goodhew, P. J. (2010). *Teaching Engineering All you need to know about engineering education but were afraid to ask*. The higher education academy. (Citado en páginas 1 y 8.)

[Iwanitz y Lange, 2010] Iwanitz, F. y Lange, J. (2010). *OPC Fundamentals, Implementation and Application*. Laxmi Publications Pvt Limited. (Citado en página 15.)

[Janevska, 2013] Janevska, G. (2013). Mathematical modeling of pump system. *University St. Kliment Ohridski*. (Citado en página 24.)

[John y Tiegelkamp, 2010] John, K. H. y Tiegelkamp, M. (2010). *IEC 61131-3: Programming Industrial Automation Systems Concepts and Programming Languages, Requirements for Programming Systems, Decision-Making Aids*. Springer Publishing Company, Incorporated, 2nd edición. (Citado en página 5.)

[Kamel y Kamel, 2016] Kamel, E. y Kamel, K. (2016). *Hands On PLC Programming with RSLogix 500 and LogixPro*. McGraw-Hill Education. (Citado en página 6.)

[Krause, *et al.*, 2002] Krause, P., Wasynczuk, O., y Sudhoff, S. (2002). *Analysis of electric machinery and drive systems*. IEEE Press series on power engineering. IEEE Press. (Citado en página 26.)

[Krishnan, 2001] Krishnan, R. (2001). *Electric motor drives: modeling, analysis, and control*. Prentice Hall. (Citado en página 26.)

[Kulakowski, *et al.*, 2007] Kulakowski, B., Gardner, J., y Shearer, J. (2007). *Dynamic Modeling and Control of Engineering Systems*. Cambridge University Press. (Citado en páginas 21 y 22.)

[Laurillard, 2002] Laurillard, D. (2002). *Rethinking University Teaching: A Conversational Framework for the Effective Use of Learning Technologies*. Rethinking University Teaching: A Conversational Framework for the Effective Use of Learning Technologies. Routledge/Falmer. (Citado en página 2.)

[Liu, *et al.*, 1989] Liu, X., Verghese, G., Lang, J., y Onder, M. (1989). Generalizing the blondel-park transformation of electrical machines: necessary and sufficient conditions. *Circuits and Systems, IEEE Transactions on*, 36(8):1058 –1067. (Citado en página 26.)

[Mahnke, *et al.*, 2009] Mahnke, W., Leitner, S., y Damm, M. (2009). *OPC Unified Architecture*. SpringerLink: Springer e-Books. Springer. (Citado en página 15.)

[Meisel, 1984] Meisel, J. (1984). *Principles of electromechanical-energy conversion*. R.E. Krieger. (Citado en página 26.)

[Moctezuma, 2015] Moctezuma, M. (2015). *Sistemas dinámicos en tiempo continuo: Modelado y simulación*. Omniascience. (Citado en página 19.)

[Prince y Felder, 2006] Prince, M. J. y Felder, R. M. (2006). Inductive teaching and learning methods: Definitions, comparisons, and research bases. *Journal of Engineering Education*, 95(2):123–138. (Citado en página 2.)

[Sánchez*, et al.*, 2002] Sánchez, J., Morilla, F., Dormido, S., Aranda, J., y Ruiperez, P. (2002). Virtual and remote control labs using java: a qualitative approach. *IEEE Control Systems*, 22(2):8–20. (Citado en página 5.)

[Sánchez Del Pozo Fernández, 2012] Sánchez Del Pozo Fernández, A. J. (2012). Simulador para control y automatización utilizando un entorno virtual interactivo y configurable. Tesis. Departamento de Sistemas y Automática, Escuela Superior de Ingenieros, Universidad de Sevilla, Departamento de Sistemas y Automática, Escuela Superior de Ingenieros, Universidad de Sevilla. (Citado en página 5.)

[Seely, 1962] Seely, S. (1962). *Electromechanical energy conversion.* Electrical and electronic engineering s. McGraw-Hill. (Citado en página 26.)

[Vargas*, et al.*, 2014] Vargas, H., Costa-Castello, R., Pavez, S., Farias, G., Hermosilla, G., Morales, J., y Dormido, S. (2014). Laboratorio de prácticas para la enseñanza de sistemas de control de tiempo real. *Congreso Latinoamericano de Control Automático. Memorias del XVI Congreso Latinoamericano de Control Automático, CLCA 2014: octubre 14-17, 2014, Cancún, Quintana Roo, México. Cancún, Quintana Roo: 2014, p. 1385-1391.* (Citado en página 5.)

[Wardhaugh, 2003] Wardhaugh, J. (2003). Benchmarking to drive a continuous improvement process. Hydrocarbon Asian Journal. 13:64, 66, 68–71. (Citado en página 30.)