ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL.

Facultad de Ingeniería en Electricidad y Computación

LABORATORIOS REMOTOS: COMUNICACIÓN CLIENTE SERVIDOR Y EJECUCIÓN REMOTA PARA LAS PRÁCTICAS DEL LABORATORIO DE CONTROL AUTOMÁTICO DE LA FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN (FIEC)

TESIS DE GRADO

Previa a la obtención de los Títulos de:

INGENIERO EN COMPUTACIÓN: ESPECIALIZACIÓN SISTEMAS TECNOLÓGICOS

INGENIERO EN COMPUTACIÓN: ESPECIALIZACIÓN SISTEMAS MULTIMEDIA

Presentado por

HUMBERTO RENE AGUILAR IÑIGUEZ

CHRISTIAN ALEXANDER IDROVO WONG

Guayaquil - Ecuador

2010

AGRADECIMIENTO

Principalmente al Ing. Juan Del Pozo, director de tesis, por su ayuda y colaboración en la realización de este trabajo.

A todas las personas que nos brindaron su apoyo y su ayuda incondicional, ya que sin sus palabras de aliento no hubiésemos podido salir adelante y sobre todo a aquellas personas que supieron soportarnos en los momentos más difíciles.

DEDICATORIA

Dedicamos este trabajo a Dios y a nuestros padres, ya que sin su apoyo incondicional no hubiésemos podido culminar con éxito este trabajo.

TRIBUNAL DE SUSTENTACIÓN

MSc. Jorge Aragundi R.

PRESIDENTE DEL TRIBUNAL

MSc. Juan del Pozo Lemos
DIRECTOR DE TESIS

MBA. Marcelo Loor Romero
VOCAL PRINCIPAL 1

MSc. Carmen Vaca Ruiz
VOCAL PRINCIPAL 2

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, nos corresponde exclusivamente; y el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral"

(Reglamento de exámenes y títulos profesionales de la ESPOL)

_____

HUMBERTO RENE AGUILAR IÑIGUEZ

_____

CHRISTIAN ALEXANDER IDROVO WONG

RESUMEN

Los avances tecnológicos impulsan la necesidad de un mayor número de equipos para los laboratorios de prácticas de ciencia y tecnología, incrementándose notablemente los costos de mantenimientos. Por ello los profesores de centros con problemas presupuestarios tienen muchas dificultades para disponer de laboratorios correctamente equipados. Por lo tanto estos avances pueden agrandar aún más la brecha digital. En este proyecto presentamos un laboratorio con experiencias reales accesibles a través de Internet. El objetivo está enfocado en la reducción de costos de los recursos para la enseñanza y sobretodo en el aprovechamiento del tiempo para el uso de dichos recursos. Este laboratorio facilita al estudiante la utilización de equipos a los que habitualmente tiene acceso únicamente en horarios restringidos y sometidos a la disponibilidad de horarios de los responsables del laboratorio. El uso de las nuevas tecnologías de enseñanza, abre nuevos caminos y contribuye a hacer posible el acceso de recursos que mejoran la calidad de la enseñanza en los países menos desarrollados

ÍNDICE GENERAL

CAPÍTULO I

CAPÍTULO II

ÍNDICE DE IMAGENES

INTRODUCCIÓN

Para los estudiantes de ciencia e ingeniería, el laboratorio y las prácticas constituyen una actividad fundamental en su proceso de aprendizaje, ya que contribuyen a mejorar la comprensión de los conceptos teóricos, a familiarizarse con la utilización de aparatos y a desarrollar competencias necesarias para su futura actividad profesional. Por estas razones las mejores instituciones y centros de formación científico-tecnológicos, dedican una gran parte de su presupuesto para el equipamiento de los laboratorios de prácticas docentes. Sin embargo, la rápida evolución de la tecnología convierte en obsoletos a muchos de estos equipos y obliga a su sustitución y al aumento de la inversión dedicada a esta actividad docente. En este contexto las instituciones con presupuestos más limitados tienen grandes dificultades para disponer de laboratorios decentes correctamente equipados. Consecuentemente estos avances tecnológicos tienen a agrandar la brecha digital (1).

Sin embargo y aunque parezca contradictorio, es posible utilizar adecuadamente la tecnología para disminuir los costos de equipamiento de los laboratorios. En efecto la paulatina implantación de la banda ancha y el desarrollo de la nueva generación de herramientas que permite el acceso remoto a programas de simulación y de control de instrumentos hacen de Internet 2.0 la herramienta ideal para optimizar o reducir los costos ligados a la educación, contribuyendo a una mejora sustancial de la misma. (2)

CAPÍTULO I

1. Visión general

1.1 Laboratorios remotos

Se puede definir un Laboratorio Remoto como un usuario con una computadora en un lugar distante que controla remotamente un experimento en una localización específica. Dispositivos de estas características vienen siendo utilizados desde hace más de 20 años en la industria, por ejemplo en el campo aeroespacial para el control de naves y telescopios espaciales. Las primeras aplicaciones de la tele manipulación a través de internet tuvieron lugar hace poco más de una década principalmente en el campo de la robótica. Se puede afirmar que la extensión de la utilización de estas aplicaciones, especialmente en el ámbito de la educación, ha sido posterior y ha venido ligada por una parte al aumento del ancho de banda disponible para los usuarios y al abaratamiento tanto de las tarifas de conexión de alta velocidad como el de las propias computadoras. En este sentido iniciativas como la propuesta por el Profesor Negroponte del MIT (Massachusetts Institute of

Technology) y la fundación One Laptop per Child OLPC (Un Portátil por Niño), para la elaboración de un ordenador portátil muy barato que permiten el acceso a la red a nuevos segmentos de población, contribuirán a extender la utilización de estas y otras aplicaciones similares.

1.1.1 Existencia e importancia

Hoy en día, existen muchos laboratorios remotos alrededor del mundo, la mayoría de los cuales han sido diseñados para resolver problemas específicos o manipular cierta maquinaria, la cual se encuentra pre configurada. Estos laboratorios representan una gran ventaja a la hora de controlar equipos, ya que permite manipularlos desde una ubicación remota cualquiera, a la hora que se decida, abaratando costos de mantenimiento y reduciendo el costo de la infraestructura de un laboratorio.

LabCon quiso dar un paso más allá y lograr un laboratorio totalmente genérico, el cual permite manipular un sinnúmero de prácticas y experimentos de manera remota, ya que este no se limita a la manipulación de parámetros, sino más bien se encuentra diseñado para que el estudiante diseñe sus propios controladores y todo el esquema

necesario para el funcionamiento y manipulación de la maquinaria que se desee, con lo cual se ha logrado optimizar completamente el uso de un laboratorio y todos los equipos; el laboratorio tomado como referencia fue el Laboratorio de Control Automático, el cual consta con un sin número de experimentos disponibles para el uso de los estudiantes, este laboratorio posee una amplia gama de equipos de instrumentación y de control que se usan comúnmente de manera industrial lo cual lo convierte en la mejor opción para el desarrollo del sistema

1.1.2 Beneficios de su creación

La ventaja más clara con la creación de LabCon es que el alumno puede realizar sus prácticas desde su casa o desde la ubicación que él prefiera en el horario deseado. Esto supone una gran reducción en el mantenimiento de las infraestructuras de los laboratorios clásicos. Y no quiere decir que los laboratorios remotos sustituyan a los clásicos, a los manuales, ya que son un complemento. Pero si el alumno no puede acceder a ese laboratorio y no es de calidad, al final va a generar una frustración. Antes de usarlo con los alumnos hay que asegurarse de tener una plataforma robusta y de calidad.

Otro de los grandes beneficios de crear un laboratorio de este tipo, es que se podría ampliar de manera significativa el número de estudiantes que tendrían acceso al laboratorio, es decir aumentar el número de estudiantes registrados al laboratorio, logrando extender el horario de atención a los alumnos a las 24 horas del día, lo cual implica que no solamente estudiantes nacionales podrían darle uso, si no también estudiantes extranjeros o las instituciones que tengan convenio con la nuestra en las cuales la diferencia de horario les permita aprovecharlo.

1.1.3 Justificación

Dada la creciente afluencia de estudiantes a nuestra institución con el pasar de los años, es correcto pensar que existirá un incremento notable en la demanda de laboratorios y de infraestructura para que los estudiantes puedan desarrollar sus prácticas. ¿Qué sucedería si se llegara al punto en el que los laboratorios ya no puedan dar cabida a más alumnos? O peor aún, llegar al punto en el que los bloques ya no puedan albergar más aulas de clases. Lo más óptimo es utilizar los recursos de los que ya disponemos y tratar de que todos los alumnos tengan acceso a ellos si necesidad de equipar nuevos laboratorios y mucho menos realizar grandes inversiones en la compra de maquinaria.

Con la implementación de un laboratorio remoto, se logra aprovechar la infraestructura de los laboratorios, y más aún, aprovechar el horario en el cual los estudiantes pueden realizar sus prácticas, ya que estas están sujetas a un horario establecido y a la disponibilidad de los encargados del mismo, extendiendo el uso de dichos laboratorios a 24 horas del día los 7 días de la semana, facilitando a todos los estudiantes y colaboradores el acceso a todos los recursos.

1.2 Equipos y Tecnologías a usarse

Dentro de la especificación de equipos y tecnologías a usarse especificaremos de manera detallada las herramientas de hardware (equipos físicos en el laboratorio), software que se encuentra dentro del servidor, y las tecnologías que el cliente necesita para poder ejecutar el sistema LabCon.

1.2.1 Equipos

1.2.1.1 Servidor (Ensamblado a gusto de los desarrolladores)

Hay muchos puntos que se deben tener en cuenta al momento de escoger un servidor para poner en producción cualquier tipo de sistema, ¿Cómo saber qué tipo de servidor escoger?, servidores dedicados o un

servidor genérico, ensamblado a petición de los usuarios. En nuestro caso dado que la aplicación es totalmente portable, genérica y se puede ejecutar en diversas plataformas decidimos ensamblar nuestro propio servidor acoplado a las necesidades y al software que se posee previamente.

En nuestro caso decidimos optar por una PC normal y convertirla en servidor, el servidor de LabCon posee un procesador Intel Core™ i7-920 (8 Mb de cache, 2.66GHz, 8CPUs), una memoria RAM de 6 Gb, motherboard Intel DX58SO con un sistema operativo Windows server 2008. Esta máquina posee una configuración de mirror de 1Tb para proveer seguridad a los datos de los usuarios.

La ventaja de usar un servidor ensamblado a gusto del usuario en primer lugar son los costos de ensamblaje y de mantenimiento, ya que la diferencia de precios contra un servidor de marca reconocida es mucha. El motivo por el cual se ensambló este servidor es que al poseer 8 procesadores se puede procesar de manera más rápida y eficiente los requerimientos de todos los usuarios, esto se tomó en consideración debido a que todo proceso se ejecutara en el servidor en tiempo real, y este tipo de operaciones requieren de gran capacidad de procesamiento, y los tiempos de respuesta son críticos en este tipo de

manipulación, ya que no queremos que existan errores con el control de la maquinaria y mucho menos provocar accidentes que puedan dañar o provocar mal funcionamiento en la misma.

1.2.1.1 Compact Field Point (CFP)

Figura 1: Compact Field Point

El Compact FieldPoint 2100 (CFP-2100) de National Instruments es un controlador industrial programable de bajo costo para el control integrado o distribuido de aplicaciones, diseñado para un fácil uso. No se requiere experiencia en programación. Puede crear aplicaciones completas que incluyen registro de datos, miles de entradas y salidas analógicas y digitales, procesamiento avanzado de señales y análisis de algoritmos, PID, e incluso control de movimiento con LabVIEW Real-Time. El cFP-2100 cuenta con un puerto Ethernet 10/100BaseT y un puerto serial RS232 para conexión a periféricos.

Los ingenieros de control suelen utilizar controladores NI cFP-2100 en ambientes industriales para ejecutar lazos de control PID, accionar válvulas y motores, tomar medidas, realizar análisis en tiempo real y de simulación, registrar datos y comunicarse de manera serial, teléfono y Ethernet.

Una vez desplegado, el controlador se puede comunicar de igual a igual con otros controladores inteligentes cFP-2100, o vía Ethernet con otros dispositivos compatibles, el cFP-2100 publica automáticamente controlador de E/S de datos a un PC con LabVIEW, a través de un servidor Web integrado, o su elección de cliente OPC o HMI / SCADA.

Para nuestro caso práctico el cFP-2100 es el que se encarga de la comunicación entre el servidor y los equipos físicos del laboratorio permitiendo así poder manipular las diversas maquinarias del laboratorio de manera automática en tiempo real.

Una de las grandes ventajas del uso de este equipo, es que debido a que posee diversos módulos para insertar tarjetas, se puede tener conectados varios dispositivos y controlar más de una maquinaria, dependiendo de las necesidades del usuario o de las necesidades del laboratorio, logrando así optimizar el uso de recursos.

1.2.2 Tecnologías en el Servidor

En esta sección se detallará de manera general todo el software instalado en el servidor el cual es requerido para realizar las prácticas y el control de la maquinaria.

1.2.2.1 Matlab Server Pages (MSP)

Es un lenguaje técnico de programación web de código abierto que usa Matlab internamente. Se soporta tres niveles de arquitecturas: web, negocios y bases de datos, además incluye computación distribuida y procesamientos paralelos mediante llamadas a procedimientos remotos y servicios web. Es independiente de la versión de Matlab que se encuentre instalado en el servidor.

Dentro de las ventajas de este software podemos recalcar que es una plataforma totalmente independiente, de código abierto, de fácil manipulación para el usuario final ya que posee diferentes interfaces de usuario: browsers (web), dispositivos móviles como PDA, y aplicaciones win32 (Windows), es de fácil configuración; tanto para el servidor como para la estructura del sistema de archivos del sitio web y lo más importante es que nos da la posibilidad de ejecutar casi la

totalidad de operaciones realizadas desde Matlab vía web, logrando así tener un control completo de la maquinaria del laboratorio.

Una vez realizada la instalación del MSP, este instalará por defecto en el servidor el Apache Tomcat, junto con archivos que sirven para configuración de las variables de entorno, y crea automáticamente la estructura de archivos de fácil configuración y manejo para el usuario.

Figura 3: Iconos de acceso del MSP

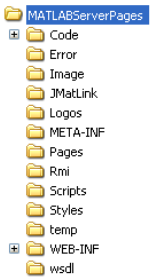


Figura 2: Sistema de archivos del MSP

1.2.2.2 Measurement and Automation Explorer (MAX)

Measurement and Automation Explorer (MAX), es un explorador de National Instruments (NI) que permite tener acceso al software de los diferentes dispositivos conectados. A través de la interface de esta aplicación se lista, configura y prueba el hardware y software, además se crea y edita canales virtuales, tareas e interfaces. Detecta los

dispositivos e instrumentos conectados al sistema y en algunos casos permite realizarles un diagnóstico.

MAX puede ser instalado a través de algunas de las aplicaciones de ambiente de desarrollo de NI tales como LabVIEW, Measurement Studio o bajo cualquier controlador como el caso de las tarjetas de adquisición de datos PCI 6024E utilizadas en el Laboratorio de Control Automático.

MAX interactúa con otras herramientas de NI al igual que con otras del sistema operativo. Tiene un aspecto parecido al explorador de Windows, al marco MMC (Microsoft Management Console) y al administrador de dispositivos, como puede verse en la siguiente figura.

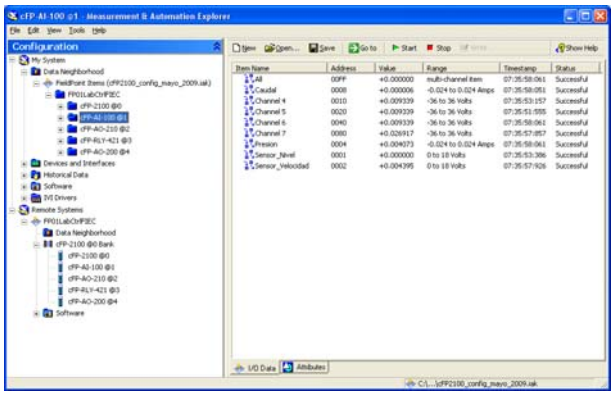


Figura 4: Ventana de configuración del MAX

En la parte izquierda de la ventana, en la sección de *Configuration* se pueden explorar los distintos dispositivos, drivers y programas. A continuación se detallará cada uno de ellos:

Data Neighborhood.- Proporciona acceso a la configuración de canales físicos del sistema mediante la creación de canales virtuales, tareas, etc., Estos canales físicos pueden ser de una DAQ o incluso de canales de los módulos de un FieldPoint que se encuentre conectado de manera remota.

Devices and Interfaces.- Lista los diferentes dispositivos físicos que se encuentran instalados en el computador tales como: tarjetas de adquisición de datos, tarjetas GPIB o aquellas conectadas a través de puertos seriales o paralelos.

Historical Data.- Provee acceso a todas las bases de datos, datos respaldados o simplemente permite visualizarlos.

Software.- Muestra información sobre el software instalado en el computador, tales como: LabVIEW, las librerías VISA, librerías GPIB, librerías DAQ y sobre el propio MAX, además permite actualizar a versiones reciente de los mismos.

Remote Systems.- Se puede visualizar y configurar dispositivos y sistemas conectados mediante Ethernet. Pueden visualizarse al expandir el + que acompaña a Remote Systems.

1.2.2.3 Sybase Central (base de datos)

Una **base de datos** o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En la actualidad, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital (electrónico), que ofrece un amplio rango de soluciones al problema de almacenar datos.

Existen programas denominados sistemas gestores de bases de datos, abreviado SGBD, que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las propiedades

de estos SGBD, así como su utilización y administración, se estudian dentro del ámbito de la informática.

Sybase Adaptive Server Anywhere (ASA) es un Sistema administrador de bases de datos relacionales (RDBMS) de alto rendimiento, que dentro de su funcionalidad incluye gestión de transacciones, un optimizador de consultas auto-afinable, integridad referencial, procedimientos almacenados Java y SQL, triggers, bloqueo a nivel de registro, programación de eventos y recuperación automática. ASA es desarrollado por iAnywhere, subsidiaria de Sybase. Dentro de las principales características podemos destacar.

- Fácil Administración
- Seguridad
- Soporte a un amplio conjunto de plataformas, herramientas y fuentes de datos

1.2.3 Tecnologías cliente

En este capítulo se describirá una breve reseña de las tecnologías utilizadas para el desarrollo y la implementación del sistema que permite la creación y manipulación de modelos vía web (flash panel).

1.2.3.1 Flash

Hace ya bastante tiempo que la Compañía Macromedia desarrolló Flash, para abrir un mundo nuevo de posibilidades de animación y movimiento a la presentación de sitios web.

Hoy en día, cualquier navegador web de cualquier fabricante o plataforma soporta el plug-in (complemento) que permite la correcta visualización de este tipo de elementos web.

Macromedia amplió a Flash más allá de las animaciones simples, convirtiéndolo en una herramienta de desarrollo completa, para crear principalmente elementos multimedia e interactivos para Internet.

El uso de esta herramienta es muy versátil, ya que permite realizar pequeñas animaciones para anuncios o logotipos, pasando por atractivos menús de navegación, hasta complejas aplicaciones que interactúan con el usuario y que van soportadas por una base de datos que almacena toda la información.

El corazón de Flash es un lenguaje de programación llamado ActionScript. Este lenguaje ha ido evolucionando desde su origen hasta convertirse en un complejo sistema que permite desarrollar, prácticamente sin límites cualquier presentación/aplicación que se desee.

Ventajas de Flash

- Capacidad de desarrollar aplicaciones ó animaciones más o menos complejas, que pueden ser incorporadas como partes de una página HTML.
- Las aplicaciones o animaciones realizadas en Flash también nos garantizan la perfecta visualización y funcionamiento de lo que queremos mostrar en cualquier navegador o plataforma desde la cual se cargue.

Inconvenientes de Flash

- Un inconveniente de los sitios que incorporan contenidos desarrollados en Flash es la carga por parte de los navegadores, ya que son aplicaciones que pueden ser relativamente pesadas, y tardar un poco en cargar. Obviamente

esto cada día es más relativo, ya que cada vez hay una mayor parte de la población que posee acceso de banda ancha a la red.

- Las aplicaciones o animaciones realizadas en Flash son en sí mismas una aplicación, es decir un archivo que es transmitido desde el servidor hasta el navegador y que una vez cargado por completo, comienza a ser ejecutado por el plug-in (complemento) que hay en el computador del cliente. Esto plantea el inconveniente de que en desarrollos muy complejos de **Flash**, al ser ejecutados por el computador que carga la página, va a depender de este la velocidad de ejecución, pudiendo no ser óptima o la deseada por el desarrollador de la aplicación.
- Adobe Flash Player (plug-in para visualizar los elementos flash vía internet) generalmente no viene incluido en los exploradores de internet más frecuentes por lo que se hace necesaria su instalación antes de poder hacer uso de las aplicaciones flash. Este proceso de instalación es sencillo.

Diferencias de la herramienta de desarrollo y reproducción

Adobe Flash

Adobe Flash es una aplicación en forma de estudio de animación que trabaja sobre "Fotogramas" destinado a la producción y entrega de contenido interactivo para diferentes audiencias alrededor del mundo sin importar la plataforma. Es actualmente escrito y distribuido por Adobe Systems, y utiliza gráficos vectoriales e imágenes ráster, sonido, código de programa, flujo de vídeo y audio bidireccional.

En sentido estricto, Flash es el entorno de desarrollo y Flash Player es el programa de máquina virtual utilizado para ejecutar los archivos generados con Flash.

Los archivos de Flash, que tienen generalmente la extensión de archivo SWF, pueden aparecer en una página web para ser vista en un navegador, o pueden ser reproducidos independientemente por un reproductor Flash.

Adobe Flash Player

Adobe Flash Player es una aplicación en forma de reproductor multimedia creado inicialmente por Macromedia y actualmente distribuido por Adobe Systems. Permite reproducir archivos SWF que pueden ser creados con la herramienta de autoría Adobe Flash, con Adobe Flex o con otras herramientas de Adobe y de terceros.

Estos archivos se reproducen en un entorno determinado (en un sistema operativo tiene el formato de aplicación del sistema, mientras que si el entorno es un navegador, su formato es el de un Plug-in u objeto ActiveX). También es utilizado para mejorar la calidad de vídeo sobre todo de Internet; cada nueva versión que sale de este producto hace que la carga de video sea más óptima y más rápida.

1.2.3.2 ActionScript

ActionScript es un lenguaje de programación orientado a objetos (OOP), que permite ampliar las funcionalidades que Flash ofrece en sus paneles de diseño y además permitir la creación de animaciones y aplicaciones web con altísimo contenido interactivo.

Provee a Flash de un lenguaje que permite al diseñador o desarrollador añadir nuevos efectos o incluso construir la interfaz de usuario de una aplicación compleja, puesto que está basado en el estándar ECMAScript (ligeramente inspirado en Java y otros lenguajes del estilo de C), un estándar para JavaScript, de ahí que ActionScript se parezca tanto a JavaScript.

Fue lanzado con la versión 4 de Flash, y desde entonces hasta ahora, ha ido ampliándose poco a poco, hasta llegar a niveles de dinamismo y versatilidad muy altos en la versión 10 (Adobe Flash CS4) de Flash.

1.2.3.3 XML

XML (siglas en inglés de eXtensible Markup Language) es un metalenguaje que permite diseñar nuestro propio lenguaje de etiquetas en base a un conjunto de reglas, fue desarrollado por el World Wide Web Consortium (W3C).

Viéndolo desde un punto de vista simple y sin entrar en definiciones muy técnicas. XML nos ayuda a tener nuestra información estructurada jerárquicamente por medio de etiquetas o Tags que

nosotros mismos crearemos. Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.

CAPÍTULO II

2. Desarrollo del Sistema

2.1 Análisis y especificaciones

Dentro de esta sección se tratará de especificar de manera detallada y clara la funcionalidad y el diseño propuesto.

2.1.1 Funcionales

Dentro de las especificaciones funcionales podemos destacar los siguientes aspectos importantes

- Validación de usuarios
- Creación de modelos
- Configuración de parámetros
- Ejecución de modelos
- Manipulación de maquinarias en tiempo real
- Visualización de resultados

2.1.1.1 Validación de usuarios

El sistema LabCon con respecto a lo referente a usuarios debe permitir controlar los siguientes aspectos.

- Permitir la creación y eliminación de nuevos usuarios.
- Permitir el ingreso solamente a personas autorizadas.
- Impedir que múltiples usuarios accedan a una práctica la cual haya sido configurada para manipulación única.

2.1.1.2 Creación de modelos

El sistema LabCon pondrá a disposición del usuario una amplia gama de componentes con los cuales el usuario podrá

- Desarrollar un modelo propio dependiendo de las especificaciones del profesor o guía
- Configurar los parámetros de los componentes del modelo
- Guardar los modelos para evitar perdida de datos
- Descargar la información de la practica o relacionada con ella
- Cargar un modelo guardado

2.1.1.3 Configuración de parámetros

Dentro de la gama de componentes provistos al usuario se le debe permitir poder realizar las acciones siguientes con dicho componente dependiendo de sus especificaciones

- Modificar los valores de las constantes
- Seleccionar bloques de la biblioteca y ubicarlos en la parte del área de dibujo deseada
- Eliminar bloques del área de dibujo
- Mover la orientación de los bloques
- Interconectar bloques
- Modificar el número de entradas o salidas para ciertos bloques específicos.
- Cambiar el nombre de los bloques (para ciertos bloques específicos)

2.1.1.4 Ejecución de modelos

Dentro de lo que respecta a la ejecución de modelos el sistema debe permitirle al usuario realizar las siguientes opciones

- Configurar un tiempo para la ejecución
- Iniciar la ejecución del modelo

- Pausar la ejecución del modelo
- Detener la ejecución del modelo
- Auto pausar la ejecución del modelo al terminar el tiempo designado para su ejecución

2.1.1.5 Manipulación de maquinarias en tiempo real

En lo que respecta a la manipulación en tiempo real, el sistema debe permitir que el usuario manipule la maquinaria sin ningún inconveniente logrando ver reflejados los cambios realizados de manera automática ya sea desde una red cercana al sitio donde se encuentre el servidor o una red externa, esto quiere decir que si el usuario modifica un valor en el modelo este tiene que verse reflejado en la maquinaria en fracciones de segundo logrando así cambios automáticos.

2.1.1.6 Visualización de resultados

Se debe dar al usuario una retroalimentación constante, ya sea en tiempo real o con los datos de la simulación para lo cual el sistema deberá permitir

- Visualizar las señales deseadas por el usuario

- Modificar las escalas para una optima visualización
- Almacenar y descargar los datos de la simulación realizada por el usuario.

2.1.2 No funcionales

2.1.2.1 Tiempos de respuesta

En lo referente a tiempos de respuesta el sistema debe responder de manera inmediata, ya que al trabajar con maquinarias es riesgoso que estas no respondan como es debido, y peor aún tener retrasos en las respuestas, con lo cual este aspecto es crítico, se debe tratar de optimizar estos tiempos de respuesta, ya sea con una red de alta velocidad (Internet 2) o con un servidor robusto, para lograr así procesar de manera más eficiente la información. En general los tiempos de respuesta del servidor para lograr una manipulación de maquinaria deben ser menores a 1 segundo o para el caso más óptimo estar en el rango de los mili segundos.

2.1.2.2 Rendimiento

Con lo referente a rendimiento ya que el servidor requiere procesar gran cantidad de información y dar servicio a múltiples usuarios es de

suma importancia que este pueda procesar varias transacciones simultáneamente y en paralelo (mínimo de 2 por usuario), logrando así, aligerar la carga de procesamiento y evitar saturar el procesador, para lograr esto es necesario un procesador de gama alta y una buena memoria para el procesador

2.2 Diseño

2.2.1 Arquitectura

En esta sección se presenta como está estructurado el sistema ya sea a nivel de software, de hardware o a nivel de redes, estos esquemas especifican en qué nivel se encuentra cada uno de los componentes, y cómo interactúan entre sí. El siguiente esquema muestra toda la arquitectura del sistema en general.

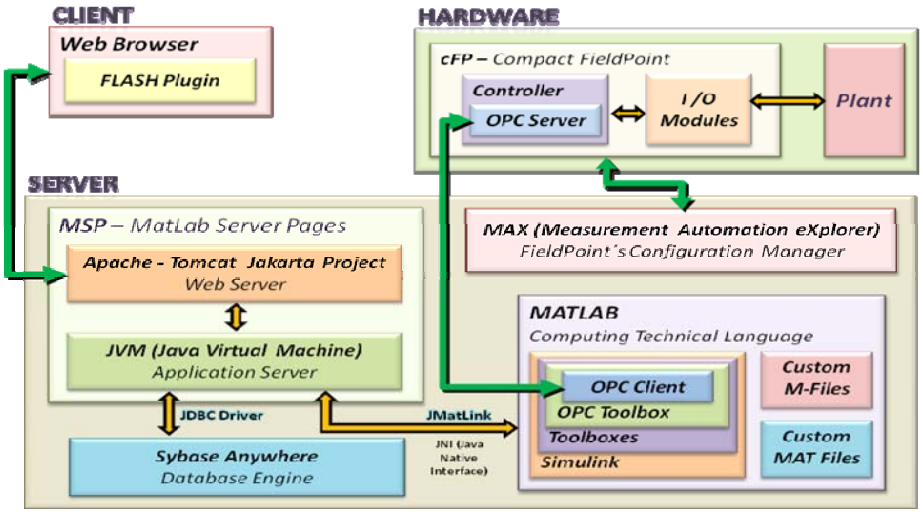


Figura 5: Arquitectura del sistema LabCon

2.2.1.1 Hardware

Dentro de la arquitectura del hardware podemos especificar el siguiente funcionamiento:

- El cFP posee módulos de entrada y salida de datos.
- Dentro del cFP se encuentra un controlador el cual posee un OPC Server.
- El controlador se comunica directamente con los módulos de entrada y salida.
- Mediante los módulos de entrada y salida se manipula la maquinaria.

2.2.1.2 Software

Dentro de la arquitectura de software podemos especificar a nivel de servidor y cliente el siguiente funcionamiento

- En el servidor se encuentra instalado el Matlab Server Pages (MSP), Sybase Anywhere (Base de datos), Matlab y el Measurement Automation Explorer (Max).
- En el paquete MSP se instalan por defecto el Apache Tomcat y JVM (Java virtual Machine).
- Matlab posee una herramienta denominada Simulink, que incluye el OPC toolbox y un OPC cliente.
- El cliente realiza un requerimiento desde el navegador web.
- El requerimiento es recibido por el apache Tomcat y enviado a la JVM para procesarlo.
- Si el requerimiento es un llamado a la base de datos, la JVM se comunica con ella mediante el JDBC Driver.
- Si el requerimiento es procesamiento de Matlab, la JVM se comunica con este mediante la librería JMatlink (librería Java).
- El requerimiento es recibido por Matlab y si es un procesamiento de manipulación de maquinaria es enviado a ella mediante el OPC Cliente.

2.2.1.3 Redes

En la arquitectura de redes podemos especificar los siguientes aspectos.

- El cliente accede a LabCon mediante la dirección web www.labcon.espol.edu.ec.
- El requerimiento es enviado vía internet a LabCon.
- El servidor DNS principal de la ESPOL recibe el requerimiento y re direcciona el requerimiento al servidor de LabCon vía intranet.
- El servidor de LabCon recibe los requerimientos a ser procesados.
- Los requerimientos de manipulación de maquinaria son enviados vía intranet hacia el Compact FieldPoint el cual se encuentra conectado a la maquinaria mediante las tarjetas de entrada y salida de datos del Compact FieldPoint.

2.2.2 Componentes (a nivel de servidor)

En esta sección se especificara que componentes se requieren en el servidor para poder poner en funcionamiento el sistema.

2.2.2.1 Librería de componentes

La librería de componentes es un archivo de extensión .mdl el cual contiene un conjunto de bloques ordenados por categorías, los cuales son necesarios para poder armar los modelos de las practicas, este archivo es llamado al momento que un usuario inicia sesión en el sistema y funciona como una referencia de la cual se copian los componentes y se insertan en el modelo del usuario, esta librería es muy importante ya que si un componente que se desee ingresar no se encuentra dentro de ésta, no se podrá insertar en el modelo del usuario, en otras palabras este bloque no existiría en el sistema LabCon.

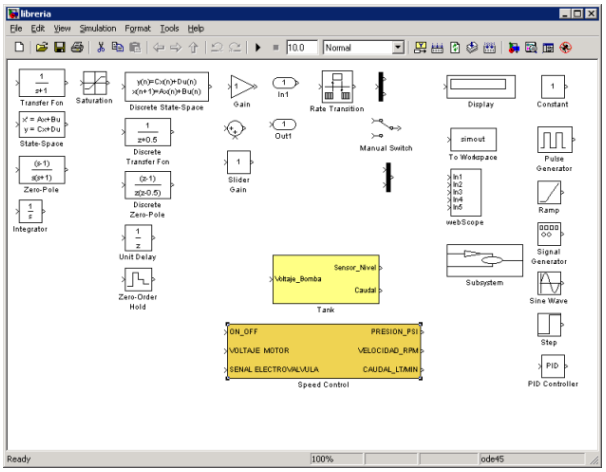


Figura 6: Librería de componentes de Matlab

2.2.2.2 Bloques

Son todos y cada uno de los componentes que provee la librería Simulink y que se encuentran a disposición dentro de la librería de componentes

2.2.2.3 Bloques especiales

Son los bloques que requirieron un diseño detallado en algunos casos un rediseño completo por parte del grupo de trabajo, para acoplarlo a las especificaciones del sistema

2.2.2.3.1 Osciloscopio

El bloque WebScope fue desarrollado dentro de la investigación de este trabajo como resultado de la necesidad de contar con un elemento que nos permita en las simulaciones presentar el desarrollo de las señales en el dominio de tiempo. Es de anotar, que el bloque utilizado por Matlab denominado Scope no fue posible interpretar su funcionamiento para utilizarlo en este trabajo. Por esta razón, el bloque WebScope cuenta con una arquitectura diferente pudiéndose resaltar el hecho que puede presentar hasta cinco señales

simultáneamente, las mismas que pueden ser ajustadas independientemente para una mejor visualización.

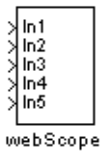


Figura 7: WebScope; imagen en librería de componentes

El funcionamiento de este bloque consiste básicamente en tomar todas y cada una de las señales de entrada (máximo cinco), almacenarlas en la base de datos con sus respectivos identificadores de usuario, para que el navegador web mediante las tecnologías cliente sea el encargado de visualizar las señales que son leídas de la base de datos. La escritura en la base de datos la realiza el Matlab en forma continua a 100 muestras por segundo; mientras que la lectura la realiza el navegador web. Esta operación se la realiza en forma tal que no altera la simulación que realiza el Matlab en el servidor, se puede decir que ocurre paralelamente.

Diseño:

Dentro del diseño del WebScope podemos encontrar el siguiente esquema; el cual es el encargado de tomar las señales de entrada asignadas, y enviarlas al bloque encargado de su proceso.

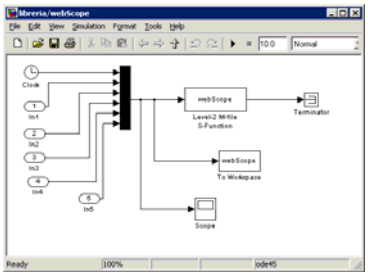


Figura 8: WebScope: Diseño interno

Programación:

Dentro del bloque WebScope.mdl se encuentra un componente de tipo S-Function denominado “Level 2- M file“, el cual es un bloque programable, al cual se lo asignó la función esencial de tomar todas y cada una de las señales de entrada y almacenar 100 muestras cada segundo en la base de datos, incorporando con esta información los identificadores de usuario. Cada punto es guardado en formato XML. El resultado de esta operación nos proporciona una ejecución de pseudo-tiempo-real de la simulación, habiéndose detectado una desviación promedio de 0.5 segundos en 100 segundos.

Una de las principales ventajas de esta implementación es que se puede disponer de los datos en cualquier momento de la simulación tanto para uno o múltiples usuarios, sin necesidad de hacer ningún tipo de interrupción en la simulación.

En el archivo de configuración podemos encontrar:

- La conexión a la base de datos.
- El número de entradas a procesar (máximo cinco).
- Cada muestra es tomada con un intervalo de 0.01 segundo.

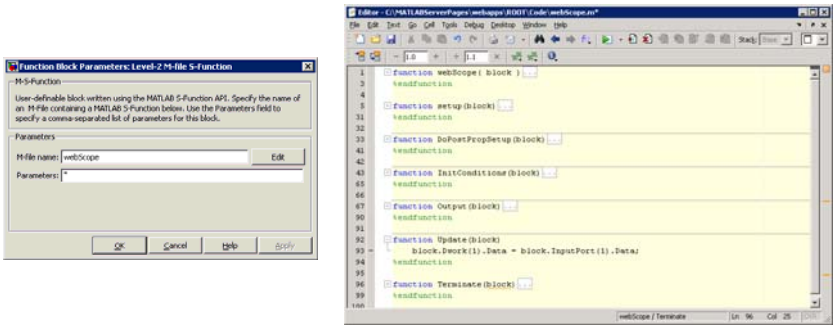


Figura 9: WebScope: Archivo de configuración

Funcionamiento:

El cliente entra al sitio web y hace uso del panel de diseño para crear su modelo, y ejecutarlo.

En la máquina del cliente el sistema Flash, realiza consultas por intervalos de tiempo al servidor a través del archivo GetPuntos.jsp ubicado en la ruta: C:\MATLABServerPages\webapps\ROOT\Panel\funciones.

Para obtener los datos de las señales que hayan sido registradas en la base de datos hasta el momento que se realiza la consulta. Estos

datos son obtenidos por el Flash en formato XML, el cual es interpretado y utilizado para la graficación de las señales. Este proceso se repite durante el tiempo que dure la simulación

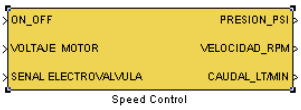
En el servidor se realiza el siguiente proceso:

- Se recibe los datos del modelo a ejecutarse enviados por el sistema Flash (bloques, líneas, parámetros de configuración, usuario, práctica).
- Se interpretan los datos por medio del archivo "generaMoldelo.jsp" el cual convierte cada una de las instrucciones en comandos interpretados por Matlab.
- Se ejecutan cada uno de los comandos generando un archivo con extensión ".mdl" de la práctica.
- Se inicia la simulación del archivo generado.
- El bloque WebScope establece una conexión con la base de datos.
- Se registran en la base de datos los valores de las señales obtenidos.

Observación: Estos dos procesos, escritura y lectura, se realizan de forma simultánea.

2.2.2.3.2 Motor

Es un bloque especialmente creado y configurado para poder manipular señales específicas y con ello un experimento definido al cual se lo denomino **Speed Control**, este bloque es un componente que requiere de la librería Opc toolbox (librería de Simulink) para poder ser creado y configurado. El funcionamiento de este bloque consiste en tomar las señales de entrada en su respectivo canal y enviarlas mediante las tarjetas cFP-AO-210, y la tarjeta cFP-RLY-421 que se encuentran el slots del cfp-2100 a la maquinaria y mediante las tarjetas cFP-AI-100 regresan los datos al sistema LabCon para que estos puedan ser almacenados, graficados, o lo que el usuario desee hacer con ellos. Cabe recalcar que cada una de las señales debe ser previamente configurada en el cfp-2100 mediante el programa MAX para que puedan ser detectadas por la aplicación.



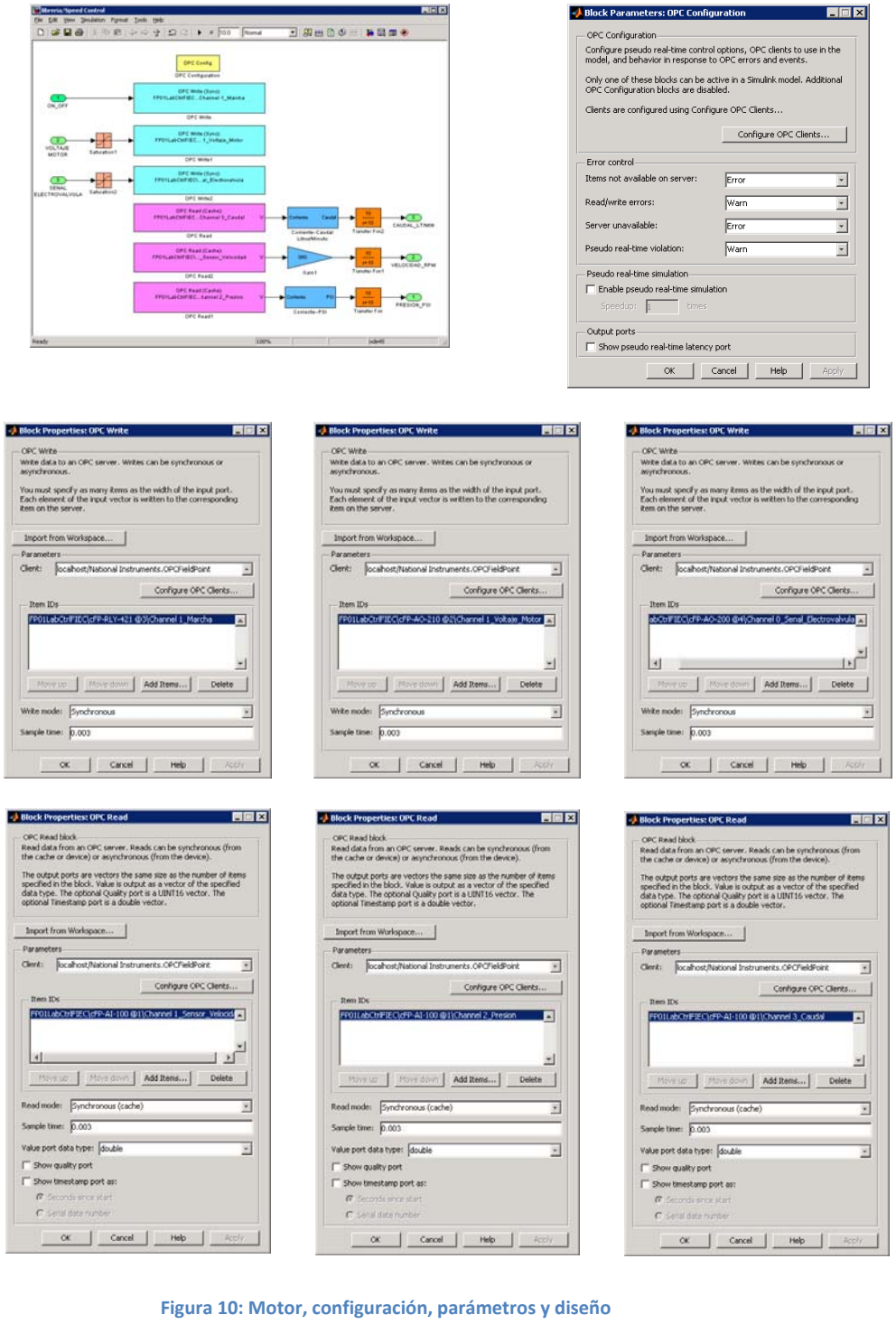


Figura 10: Motor, configuración, parámetros y diseño

2.2.2.3.3 Tanque

Es un bloque especialmente configurado para poder manipular un experimento específico denominado por los desarrolladores como **Tank**, este bloque es un componente que al igual que el Speed Control, requiere de la librería Opc toolbox para poder ser creado y configurado, el principio de funcionamiento es el mismo que el del bloque especial motor, los datos de entrada son enviados mediante la tarjera cFP-AO-210 hacia la maquinaria y los datos de salida enviados al sistema LabCon mediante la **tarjeta** cFP-AI-100 para poder ser interpretados y usados.

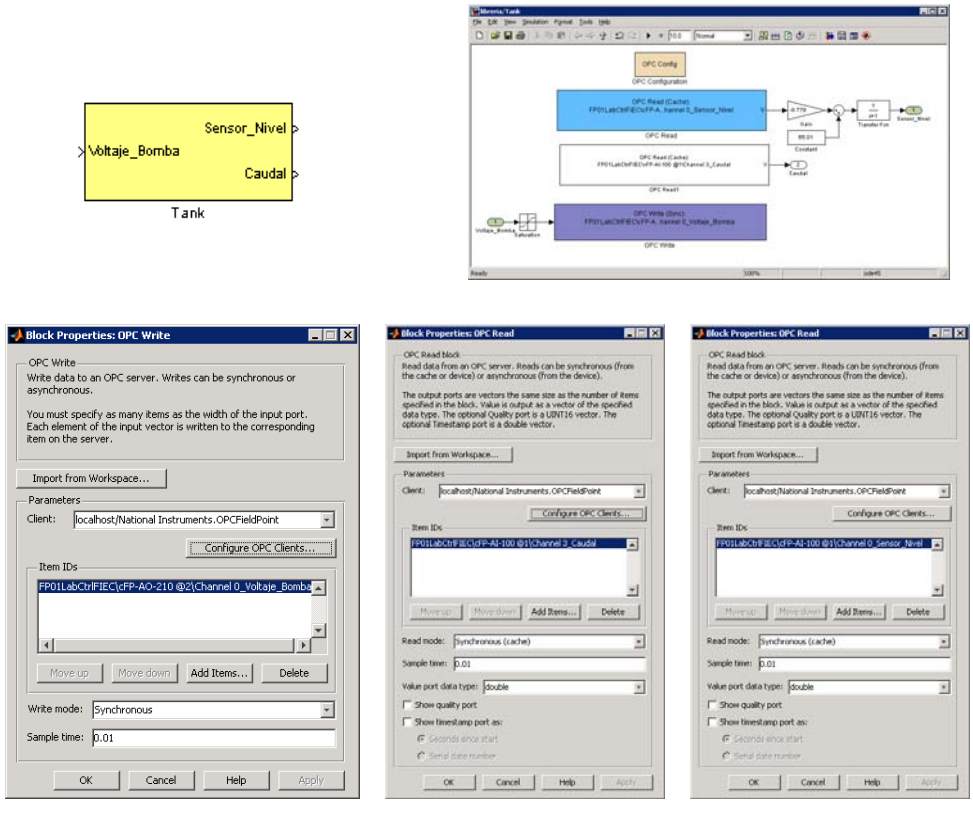


Figura 11: Tanque, configuración, parámetros y diseño

2.2.3 Especificación de bloques

Se proporcionará una breve descripción de los bloques de la **Librería** y su uso.

Transfer Fcn

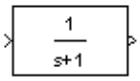


Figura 12: Transfer Fcn

El bloque Transfer Fcn modela un sistema lineal por una función de transferencia en el dominio de Laplace.

State-Space

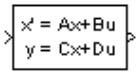


Figura 13: State-Space

El bloque State-Space implementa un sistema cuyo comportamiento se define como

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

Donde x es el vector estado, u es el vector de entrada, y es el vector de salida. Los coeficientes de la matriz deben tener las siguientes características:

A debe ser una matriz de nXn, donde n es el número de estados.

B debe ser una matriz de nXm, donde m es el número de entradas.

C debe ser una matriz de rXn, donde r es el número de salidas.

D debe ser una matriz de rXm

Zero-Pole

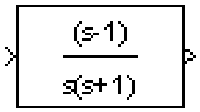


Figura 14: Zero-Pole

El bloque Zero-Pole modela un sistema que se define con ceros, polos y la ganancia de una función de transferencia de Laplace. Este bloque puede modelar: entrada-simple, salida-simple (SISO) y entrada-simple, salida-múltiple (SIMO) sistemas.

Integrator

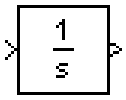


Figura 15: Integrator

El bloque Integrator envía como salida la integral de la señal de entrada. La siguiente ecuación representa la salida del bloque en función de la entrada U y una condición inicial Yo, donde Y y U son funciones vectoriales del tiempo de simulación.

Saturation

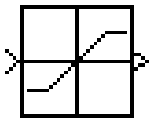


Figura 16: Saturation

El bloque de saturación impone límites inferiores y superiores a una señal, cuando la señal de entrada se encuentra dentro del rango especificado por el límite inferior y los parámetros de límite superior, la señal de entrada pasa a través del bloque sin cambios. Cuando la señal de entrada está fuera de estos límites, el bloque asigna la señal de

salida en el nivel superior o inferior dependiendo del rango en el cual se encuentre.

Discrete State-Space

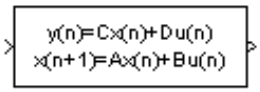


Figura 17: Discrete State-Space

El bloque Discrete State-Space implementa un sistema definido por

$$x(n+1) = Ax(n) + Bu(n)$$

$$y(n) = Cx(n) + Du(n)$$

Donde u es la entrada, x es el estado y y la salida. La matriz de coeficientes debe tener las siguientes características

A debe ser una matriz de nXn, donde n es el número de estados

B debe ser una matriz de nXm, donde m es el número de entradas

C debe ser una matriz de rXn, donde r es el número de salidas

D debe ser una matriz de rXm

Discrete Transfer Fcn

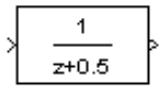


Figura 18: Discrete Transfer Fcn

El bloque Discrete Transfer-Fcn implementa la transformada Z de una función de transferencia siguiendo el siguiente esquema

$$H(z) = \frac{num(z)}{den(z)} = \frac{num_0 z^m + num_1 z^{m-1} + \cdots + num_m}{den_0 z^n + den_1 z^{n-1} + \ldots + den_n}$$

Donde m+1 y n+1 son los coeficientes del numerador y del denominador respectivamente, num y den contienen los coeficientes del numerador y el denominador en potencias decrecientes de z, num puede ser un vector o una matriz, den debe ser un vector, y se deben especificar ambos parámetros en el cuadro de dialogo. El orden del denominador debe ser mayor o igual a la orden del numerador.

Discrete Zero-Pole

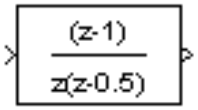


Figura 19: Discrete Zero-Pole

El bloque Discrete Zero-Pole modela un sistema discreto definido por seros, polos y la ganancia en el dominio de z de una función de transferencia de Laplace. Este bloque asume que la función de transferencia tiene el siguiente modelo.

$$H(z) = K\frac{Z(z)}{P(z)} = K\frac{(z - Z_1)(z - Z_2)\dots(z - Z_m)}{(z - P_1)(z - P_2)\dots(z - P_n)}$$

Donde Z representa el vector seros, P el vector polos y K la ganancia. El número de polos debe ser mayor o igual al número de ceros (m ≥ n). Si los polos y ceros son complejos, entonces deben ser pares conjugados complejos.

Unit Delay

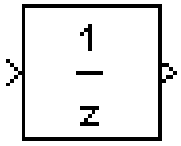


Figura 20: Unit Delay

El bloque Unit-Delay retrasa una entrada por un periodo de tiempo especificado. El bloque acepta una entrada y genera una salida, que

puede ser tanto escalares o vectoriales. Si la entrada es un vector, todos los elementos del vector se retrasan por el período del retraso.

Zero-Older Hold

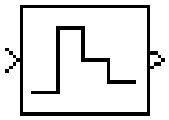


Figura 21: Zero-Older Hold

El bloque Zero-Order Hold muestrea y mantiene la entrada de una señal durante el periodo que se le especifique, El bloque acepta una entrada y genera una salida, cada señal puede ser un escalar o un vector. Si la señal de entrada es un vector, el bloque bloquea todos los elementos del vector durante el periodo especificado.

Gain

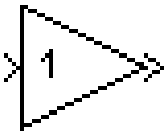


Figura 22: Gain

El bloque Gain multiplica el valor de entrada por una constante determinada, la entrada y la ganancia pueden ser escalares, vectoriales o matrices.

Sum

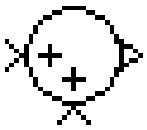


Figura 23: Sum

El bloque Sumador realiza sumas o restas con las señales de entrada, este bloque puede sumar o restar escalares, vectores o matrices

Slider Gain

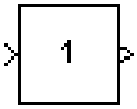


Figura 24: Slider Gain

El bloque Slider Gain facilita la modificación de parámetros durante la ejecución de un modelo, desplegando un menú en el cual se muestra una barra desplegable con la cual se define los valores

In

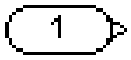


Figura 25: In

El bloque in define el enlace de una señal desde un sistema externo, comúnmente usado para definir las entradas de un subsistema

Out

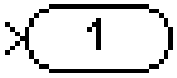


Figura 26: Out

El bloque Out define la salida de un sistema, comúnmente usado para definir las salidas de un subsistema

Rate Transition

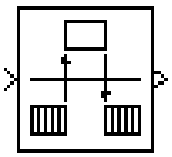


Figura 27: Rate Transition

El bloque Rate Transition transfiere los datos de la salida de un bloque operando a una **taza** de muestreo, a la entrada de otro bloque funcionando a una diferente tasa de muestreo, este bloque se usa para transferir datos íntegros, para lograr una rápida respuesta y bajos requerimientos de memoria.

Demux

Figura 28: Demux

El bloque Demux extrae los componentes de una señal de entrada y envía cada uno de esos componentes en señales separadas.

Manual Switch

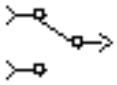


Figura 29: Manual Switch

El bloque Manual Switch es un interruptor que selecciona a una de sus dos entradas para enviarlas como salida del bloque.

Mux

Figura 30: Mux

El bloque Mux combina las señales de entrada y las envía como un vector único de salida.

Display

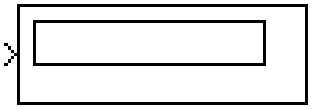


Figura 31: Display

El bloque Display sirve como una pantalla mostrando el valor numérico de una señal

Pulse Generator

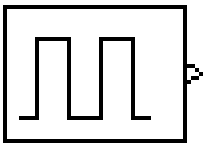


Figura 32: Pulse Generator

El bloque Pulse Generator genera impulsos de ondas cuadradas a intervalos regulares, los parámetros del bloque: forma de onda, amplitud, ancho del pulso, periodo y el retardo de la fase determinan la forma de onda a la salida del bloque.

Ramp

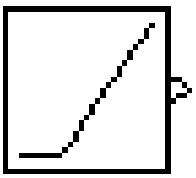


Figura 33: Ramp

El bloque Ramp genera una señal que empieza en un tiempo determinado, y su valor cambia dependiendo de un especificado rango de valores.

Signal Generator

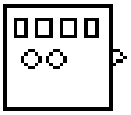


Figura 34: Signal Generator

El bloque Signal Generator puede generar una de las siguientes señales de onda, onda sinusoidal, onda cuadrada, onda de diente de sierra, y una onda al azar; los parámetros de la señal pueden ser expresados en Hertz (valor por defecto) o en radianes por segundo.

To Workspace

Figura 35: To Workspace

El bloque TO Workspace recibe una entrada como señal y escribe los datos en el workspace de Matlab (espacio de trabajo donde se almacenan los datos); este bloque puede escribir los datos como arreglos o como estructuras con el nombre que se le designe en la variable nombre, el formato con el que sea grabado los datos determinan el formato de salida.

Subsystem

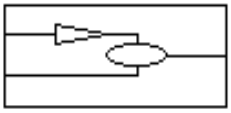


Figura 36: Subsystem

El bloque Subsystem crea un subsistema dentro de un modelo, logrando así optimizar el uso de espacio o una mejor organización

Constant

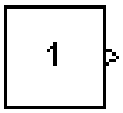


Figura 37: Constant

El bloque Constant genera una señal de un valor constante o complejo dependiendo de las configuraciones.

Sine Wave

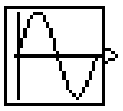


Figura 38: Sine Wave

El bloque de onda sinusoidal proporciona una sinusoide. El bloque puede funcionar basado en tiempo o en modo basado en muestras.

Step

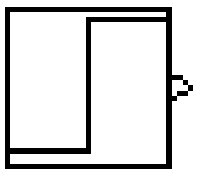


Figura 39: Step

El bloque Step proporciona un paso entre dos niveles definidos en un tiempo determinado, si el tiempo de simulación es menor que el valor del parámetro de paso, la salida del bloque es el valor del parámetro “valor inicial”, en cambio un mayor tiempo de simulación o igual que el de paso de tiempo da como valor de salida del bloque el parámetro “valor final”.

PID Controller

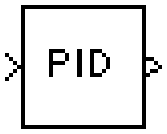


Figura 40: Pid Controller

El bloque PID Controller implementa un controlador de tiempo ya sea continuo o discreto. Las ganancias del controlador Pid son configurables ya sea manualmente o automáticamente; el ajuste automático requiere diseño de control mediante Simulink

2.2.4 Diseño de los datos (Flash Panel)

Describe las estructuras de datos y procedimiento de los principales elementos utilizados en el sistema del panel Flash para poder llevar a cabo el manejo de los datos.

2.2.4.1 Bloques

ID	Número - Identificador único del bloque
GRUPO	Texto - Nombre del grupo al que pertenece
TIPO	Texto - Nombre del tipo de bloque
NPIN	Número - Cantidad de puertos de entradas del bloque
NPIN_NEW	Número - Cantidad de puertos de entradas del bloque
NPOUT	Número - Cantidad de puertos de salidas del bloque
NPOUT_NEW	Número - Cantidad de puertos de salidas del bloque
DETALLE	Texto - Descripción del bloque
ESTADO	Booleano - Indica si el bloque es válido o no
VAR_TIPO	Arreglo - Nombres del tipo de parámetros configurable
VAR_LABEL	Arreglo - Nombres de las etiquetas de los parámetros
VAR_NOMBRE	Arreglo - Nombres de las variables de los parámetros

VAR_VALOR	Arreglo - Valores de las variables de los parámetros.
VAR_VALORFINAL	Arreglo - Valores finales asignadas de las a los parámetros.
VAR_ESTADO	Booleano - Indica si el parámetro ha sido modificado.
COLOR_LINEA	Color - Asigna el color con el cual se dibujarán las líneas de conexión salientes
VENTANA_TIPO	Número - Factor proporcional que determina porte de la ventana de configuración de parámetros del bloque
ROTAR	Número - Indica si el bloque está en estado normal o rotado. 0: Estado normal 1: Está rotado
NIVEL	Número - Indica en qué nivel está ubicado el bloque, subsistemas

2.2.4.2 Líneas

NODOA	Número - Identificador del puerto de salida
NODOA_POSICION	Número - Posición del puerto de salida

NODOA_PADRE	Número - Identificador del bloque contenedor del puerto de salida
NODOB	Número - Identificador del bloque contenedor del puerto de entrada
ENTRADAID	Número - Identificador del puerto de entrada
ESTADO	Booleano - Indica si la línea es válida o no
DISPONIBLE	Booleano - Indica si el puerto de entrada de la línea está disponible o no

2.2.4.3 Puertos de entrada y de salida

Puertos de entrada

ID	Número - Identificador único del puerto de entrada del bloque
POSICION	Número - Indica la posición del puerto de entrada en el bloque
DISPONIBLE	Booleano - Indica si el puerto de entrada de la línea está disponible o no

PADRE	Número - Identificador del bloque contenedor del puerto de entrada

Puertos de salida

ID	Número - Identificador único del puerto de salida del bloque
POSICION	Número - Indica la posición del puerto de salida en el bloque
PADRE	Número - Identificador del bloque contenedor del puerto de salida

2.2.4.4 Datos externos (XML)

Grupos, bloques y parámetros

Grupo (group)

Name	Texto - Nombre del grupo

Bloque (block)

Name	Texto - Nombre del bloque
NpIN	Número - Número de entradas del bloque

NpOUT	Número - Número de salidas del bloque
sWIN	Número - Número proporcional que determina el alto de la ventana de propiedades del bloque
desc	Texto - Descripción del bloque
enable	Número - Indica si el bloque está habilitado (1) o no (0)

Eventos especiales (start)

Name	Texto - Nombre del evento
Values	Texto - Valores utilizados según el evento

Parámetros (param)

Name	Texto - Tipo de parámetro
Label	Texto - Etiqueta del nombre del parámetro
varname	Texto - Nombre de la variable utilizado en MATLAB
def	Texto - Valor por defecto del parámetro al crearse el bloque

validation	Texto - Variables del bloque que dada una condición especial, permite realizar una acción predeterminada
image	Texto - Nombre de la variable que al darse una condición, cambia la imagen representativa del bloque
ports	Texto - Variables del bloque que dada una condición especial, permite realizar una acción predeterminada referente a los puertos

Validación de variables

Variable

Name	Texto - Nombre del tipo de parámetro que se valida

name: TextInput

name	Texto - Nombre de la variable a validar
restrict	Texto - Dominio permitido para la variable a validar
maxchar	Número - Longitud máxima permitida en la variable a validar

2.2.4.5 Sesiones (XML)

Nodo

Evento	Texto - Tipo de evento que se registra al guardar/cargar la sesión
Evento:	Scope
Time	Número - Valor del tiempo que ha sido configurado en la sesión
PXmin	Número - Valor mínimo del eje X para la graficación en la sesión
PXmax	Número - Valor máximo del eje X para la graficación en la sesión
pY1min	Número - Valor mínimo del eje Y1 (eje A) para la graficación en la sesión
pY1max	Número - Valor máximo del eje Y1 (eje A) para la graficación en la sesión
pY2min	Número - Valor mínimo del eje Y2 (eje B) para la graficación en la sesión
pY2max	Número - Valor máximo del eje Y2 (eje B) para la graficación en la sesión

pY3min	Número - Valor mínimo del eje Y3 (eje C) para la graficación en la sesión
pY3max	Número - Valor máximo del eje Y3 (eje C) para la graficación en la sesión
pY4min	Número - Valor mínimo del eje Y4 (eje D) para la graficación en la sesión
pY4max	Número - Valor máximo del eje Y4 (eje D) para la graficación en la sesión
pY5min	Número - Valor mínimo del eje Y5 (eje E) para la graficación en la sesión
pY5max	Número - Valor máximo del eje Y5 (eje E) para la graficación en la sesión
Evento:	add_block
ID	Número - Identificador único del bloque utilizado en la sesión
Tipo	Texto - Nombre del tipo de bloque utilizado en la sesión
Px	Número - Indica la ubicación horizontal del bloque con referencia a la raíz del sistema (base)

pY	Número - Indica la ubicación vertical del bloque con referencia a la raíz del sistema (base)
flip	Número - Indica si el bloque está en su estado normal o rotado 0: Estado normal 1: Está rotado
Nivel	Número - Indica el nivel donde se encuentra agregado el bloque 0: El bloque está ubicado en la base 1 o más: El bloque está ubicado dentro de algún subsistema
nameBlock	Texto - Nombre del bloque personalizado por el usuario
Evento:	set_param
ID	Número - Identificador único del bloque utilizado en la sesión
nombreVar	Texto - Nombre de la variable que ha sido configurada en la sesión

valorFinal	Texto - Valor de la variable que ha sido configurada en la sesión
Evento:	add_line
idA	Número - Identificador del bloque que conecta su puerto de salida
posa	Número - Posición del puerto de salida del bloque que se conecta
idB	Número - Identificador del bloque que conecta su puerto de entrada
posa	Número - Posición del puerto de entrada del bloque que se conecta

2.2.5 Diseño de la interfaz (Flash Panel)

Se detallan los principales elementos visuales utilizados para formar el flash panel.

2.2.5.1 Pantalla principal

Es el principal área del sistema, la base donde se manipulan y agregan los diferentes elementos disponibles.

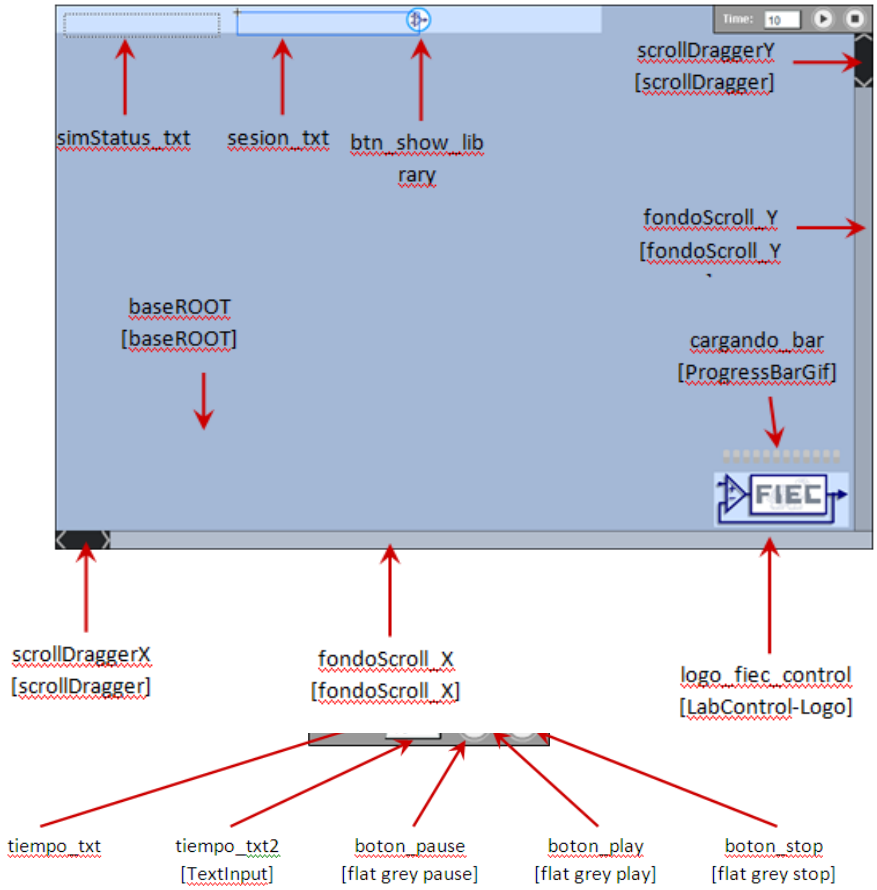


Figura 41: Componentes de la pantalla principal de diseño

2.2.5.2 Biblioteca de recursos

En la biblioteca de recursos, están todos los clips de película, botones, componentes, figuras utilizadas para formar el flash panel. Está organizado su contenido en varios niveles según su utilidad.

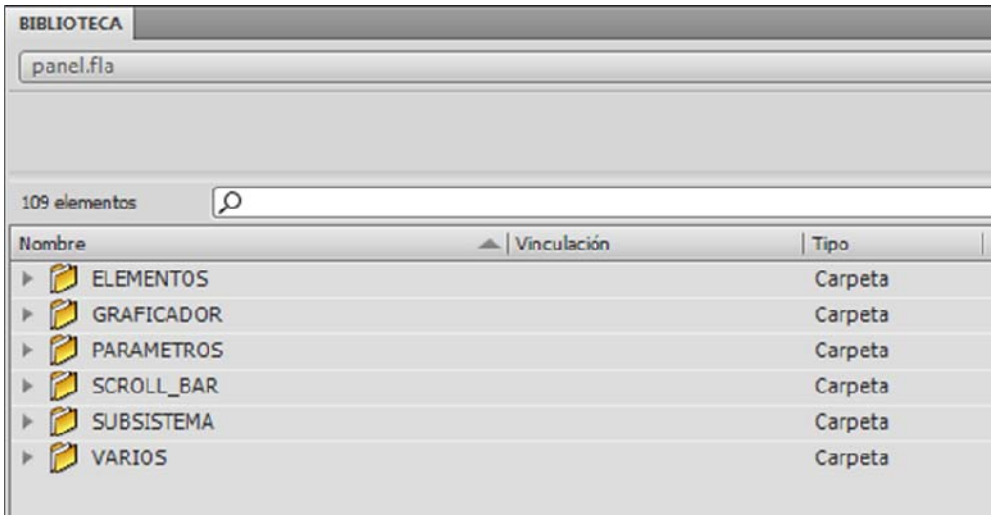


Figura 42: Biblioteca de recursos

2.2.5.2.1 Elementos

Agrupa todos los elementos que forman los bloques.

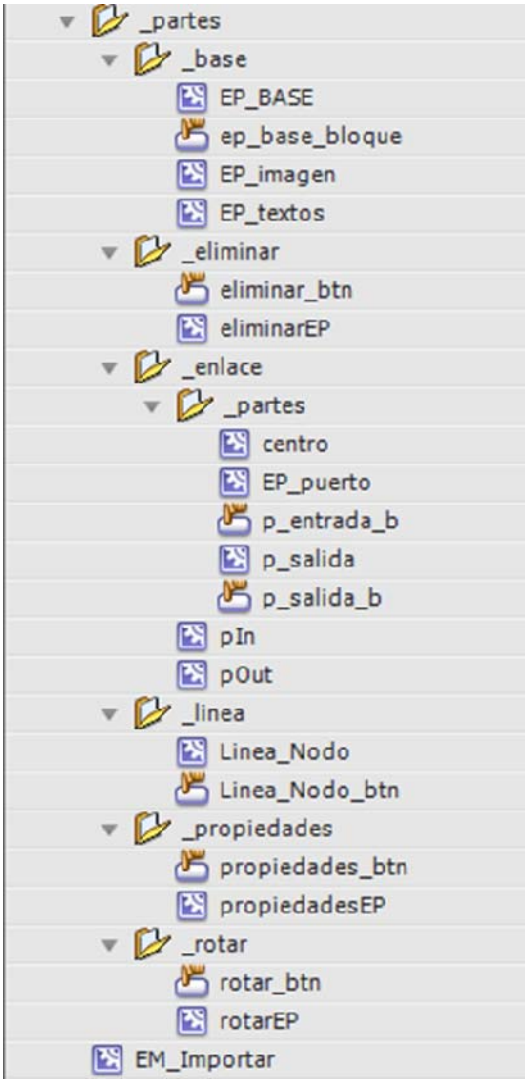


Figura 43: Elementos de los bloques

EP_BASE

Es la **base** visual de un bloque, contiene los elementos que la forman para mostrar visualmente al usuario.

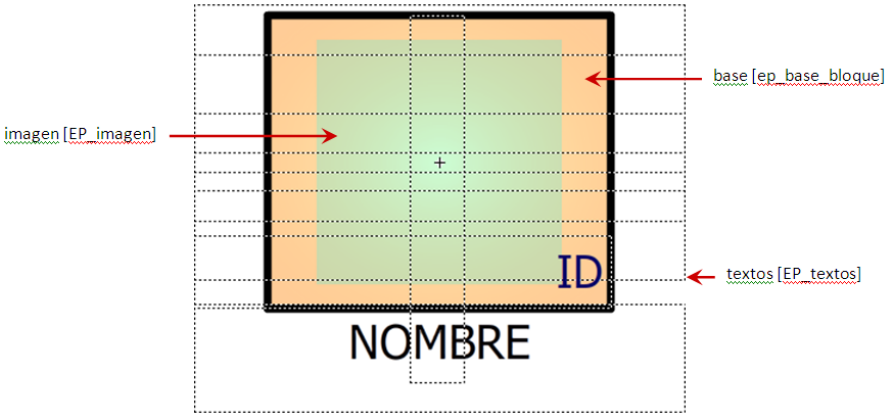


Figura 44: Bloque base

EP_textos

Indica las variables disponibles para cambiar las leyendas de texto del bloque.

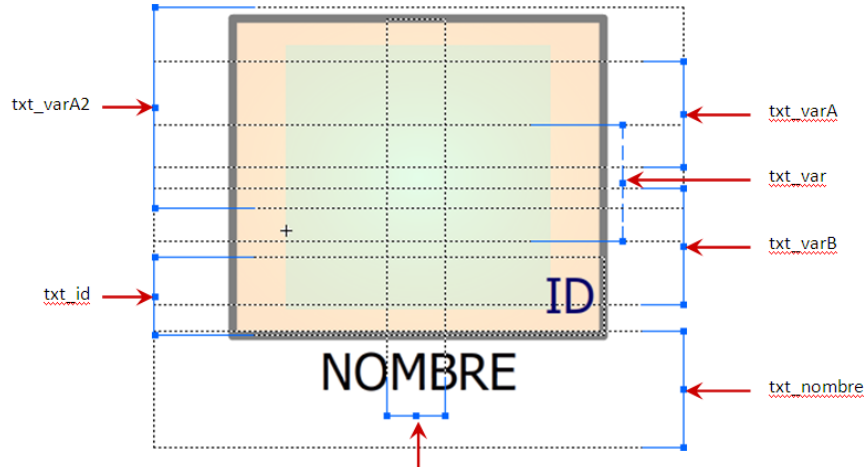


Figura 45: Leyendas de texto en un bloque base

EM_Importar

Es un clip de película utilizado para automatizar el proceso de agregar bloques.

eliminarEP

Botón utilizado para eliminar bloques.

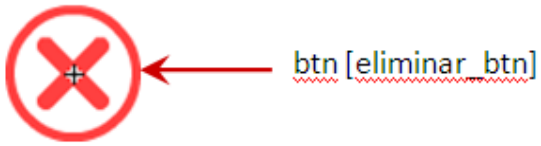


Figura 46: Botón eliminar

propiedadesEP

Botón utilizado para ver las propiedades del bloque.

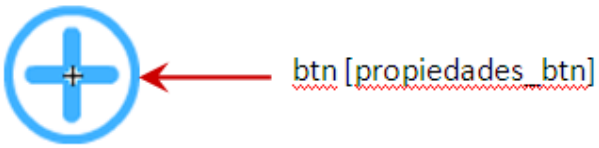


Figura 47: Botón de propiedades

rotarEP

Botón utilizado para poder rotar la orientación horizontal del bloque.

Figura 48: Botón para rotar un bloque

pIn

Indica los elementos que forman el puerto de entrada del bloque.

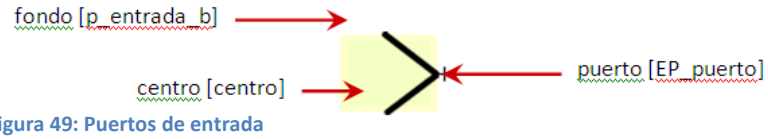
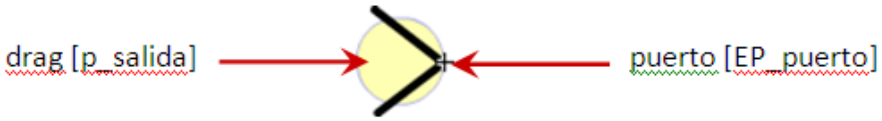


Figura 49: Puertos de entrada

pOut

Indica los elementos que forman el puerto de salida del bloque.



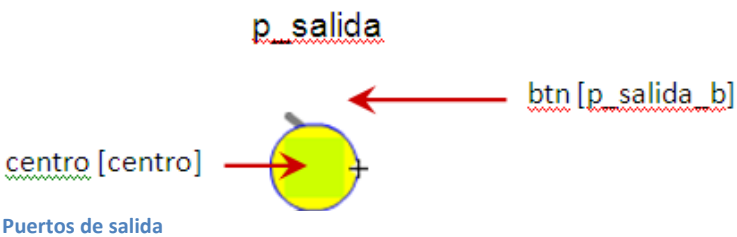


Figura 50: Puertos de salida

Linea_Nodo

Indica los elementos que forman del nodo de conexión del bloque.

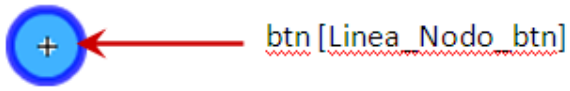


Figura 51: Nodos para el trazado de líneas

2.2.5.2.2 Graficador

Agrupa todos los elementos que forman el graficador.

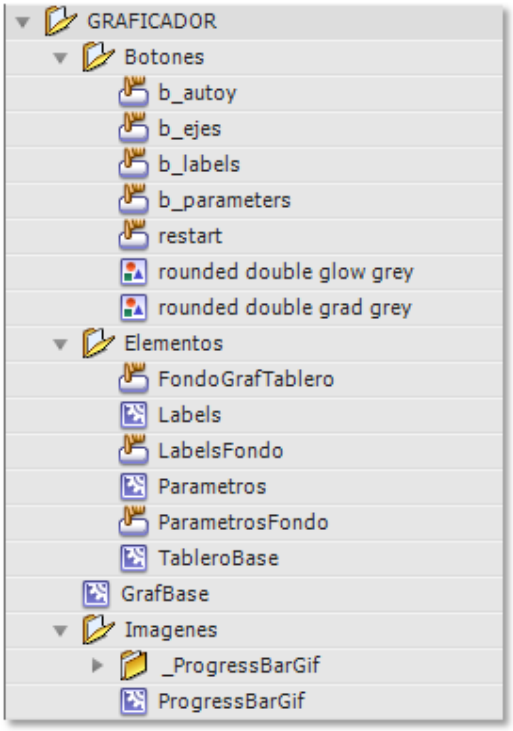


Figura 52: Elementos del graficador

GrafBase

Es la principal área del graficador, la base donde se visualiza y configura los diferentes elementos disponibles.

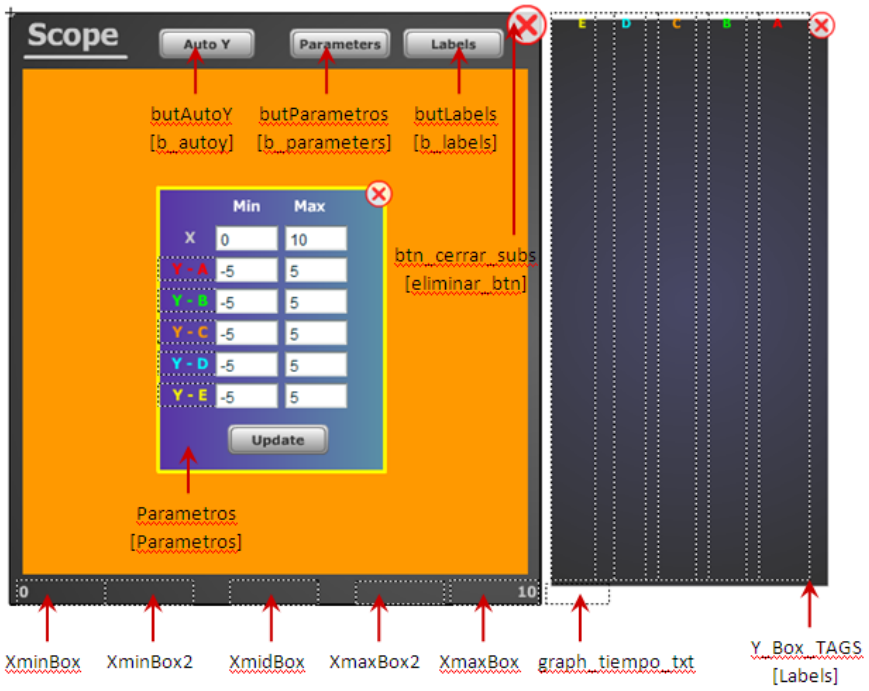


Figura 53: Diseño del Web Scope

2.2.5.2.3 Parámetros

Indica los elementos que forman la ventana de configuración de los ejes del graficador.

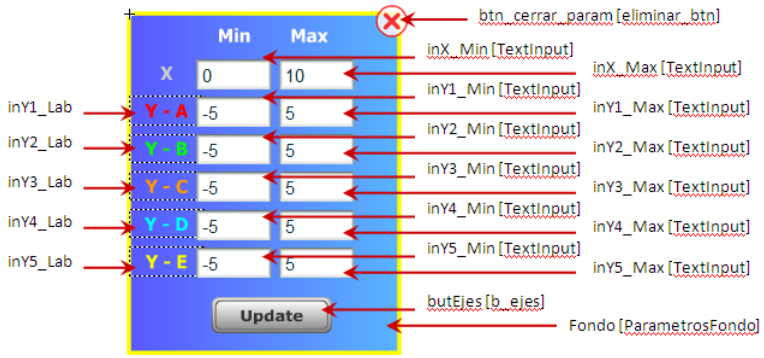


Figura 54: Ventana de configuración de parámetros

Labels

Indica los elementos que forman la ventana de visualización de valores.

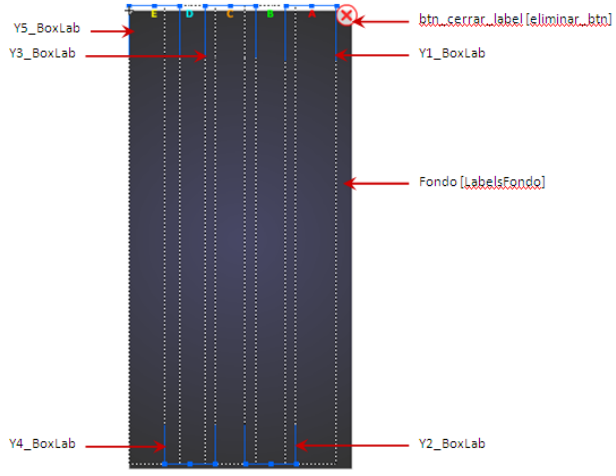


Figura 55: Ventana de escalas de medición

Parámetros

Agrupa todos los elementos que forman el graficador.

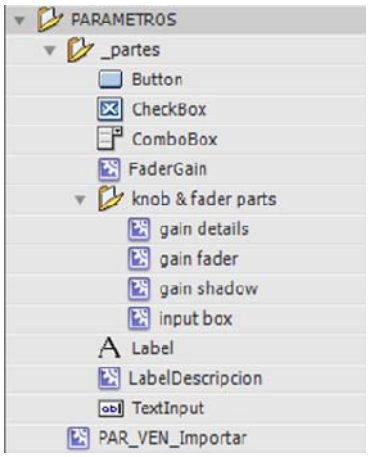


Figura 56: Parámetros del graficador

PAR_VEN_Importar

Es un clip de película utilizado para automatizar el proceso de agregar parámetros a la ventana de configuración del bloque.

Button

Recurso proporcionado originalmente en el flash.

Figura 57: Botones

CheckBox

Recurso proporcionado originalmente en el flash.

Figura 58: CheckBox

ComboBox

Recurso proporcionado originalmente en el flash.

Figura 59: ComboBox

FaderGain

Recurso proporcionado originalmente en el flash.

Figura 60: Slider

Label

Recurso proporcionado originalmente en el flash.

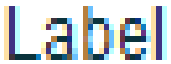


Figura 61: Etiquetas

LabelDescripcion

Recurso proporcionado originalmente en el flash.

Figura 62: Descripción de etiquetas

TextInput

Recurso proporcionado originalmente en el flash.

Figura 63: Entrada de texto

2.2.5.2.4 Subsistemas

Indica los elementos que forman los subsistemas del modelo.

Figura 64: Elementos de Subsistemas

SubSistema

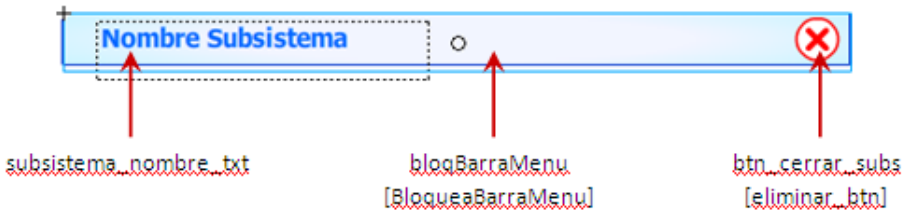


Figura 65: Cabecera de un subsistema

2.2.5.2.5 Varios

Son elementos varios utilizados por el sistema para permitir la interacción con el usuario.

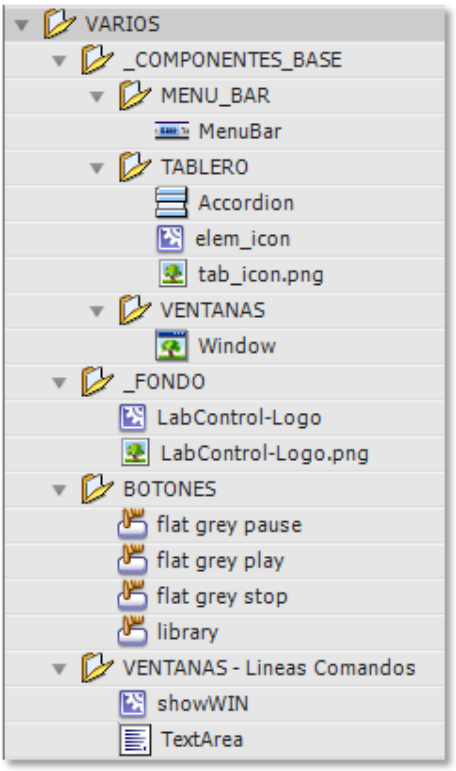


Figura 66: Elementos para interacción del usuario

MenuBar

Componente utilizado para la barra de menú de opciones.

Figura 67: barra de menú

Accordion

Componente utilizado para generar la librería de bloques.

Figura 68: Accordion

elem_icon

Recurso gráfico para ser utilizado en la librería de bloques.

Figura 69: Icono de la librería de componentes

Window

Componente utilizado para generar las ventanas de configuración de parámetros de los bloques.

Figura 70: diseño de ventanas

2.2.6 Diseño de procedimientos (Flash Panel)

Detalla las diferentes funciones disponibles para generar, estructurar, organizar, exportar, almacenar, graficar, interactuar, entre otras posibilidades del sistema.

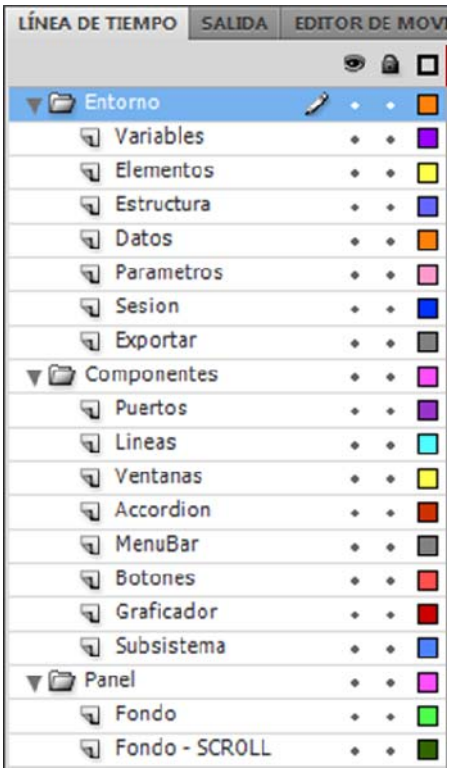


Figura 71: Diseño de procedimientos

2.2.6.1 Entorno

Se encuentran especificadas las variables y funciones que estructuran los elementos del sistema Flash. Así también los procedimientos para la manipulación de sesiones y la comunicación con el servidor para generar los modelos y su simulación.

2.2.6.1.1 Variables

estado_genModelo	Texto - Indica el estado de la simulación
array_elemPanel	Arreglo - Contiene todos los bloques agregados en la sesión
array_entradas	Arreglo - Contiene todos los puertos de entrada de los bloques agregados en la sesión
array_salidas	Arreglo - Contiene todos los puertos de salida de los bloques agregados en la sesión
array_lineas	Arreglo - Contiene todas las líneas de conexión establecidas en la sesión
array_eliminarEP	Arreglo - Contiene todos los botones de eliminar de los bloques agregados en la sesión

redibujarLineas	Booleano - Indica si las líneas de conexión establecidas deben ser redibujadas o no
colorLineaArray	Arreglo - Contiene todos los colores disponibles para asignar a las líneas de conexión entre bloques
colorLineaActual	Color - Indica el color actual que se asignará al bloque que se agregue a la sesión.
mostrarLibrary	Booleano - Indica si la librería de bloques se muestra o no
miAccordion	Clip de Película - Elemento correspondiente a la librería de bloques activa
ERROR_SESION	Booleano - Indica si ha ocurrido algún error en la sesión
ACTIVE_username	Texto - Nombre del usuario activo en la sesión
ACTIVE_numPractice	Texto - Nombre del número de práctica activa en la sesión
ACTIVE_bloquePractica	Texto - Nombre del bloque especial que se activa al ser una práctica especial predeterminada

estado_SESION Booleano - Indica si ha ocurrido algún error en la gestión de carga o almacenamiento de la sesión

url_XML_Graficador_PATH Texto - Ruta de la carpeta donde se encuentran las funciones para transmisión de los datos del graficador

numeroFunciones Número - Indica el número de funciones activas en el graficador

str_ErrorLogs Texto - Indica los errores que han ocurrido en el sistema

array_subsistemas Arreglo - Contiene todos los subsistemas agregados en la sesión

NIVEL_SISTEMA_ACTUAL Número - Indica el nivel en el que encuentra ubicado al visualizar el sistema

0: Se encuentra en la base

1 o más: Se encuentra ubicado dentro de algún subsistema

2.2.6.1.2 Elementos

crearElemMenu

- Permite agregar bloques a la librería de bloques
- Permite agregar bloques al panel de trabajo
- Permite validar acciones según el tipo de bloque

addElemPanel

- Agrega bloques al panel de trabajo
- Asigna los valores importados de los datos externos a la estructura de dato del bloque
    - Asigna los datos a la estructura de datos del bloque
    - Asigna la imagen de presentación al bloque
    - Agrega ventana y efectos de arrastre al bloque
    - Agrega botón eliminar bloque
    - Agrega botón propiedades del bloque
    - Agrega botón rotar bloque
    - Asigna etiquetas de texto del bloque
- Actualiza configuraciones del bloque acorde indicaciones preestablecidas
- Agrega puertos de entrada y de salida al bloque

addElemPanel_SUM

- Actualiza un bloque del tipo 'Sum' agregado al panel de trabajo, aplicando validaciones especiales

addElemPanel_MUX

- Actualiza un bloque del tipo 'Mux' agregado al panel de trabajo, aplicando validaciones especiales

addElemPanel_MANUALSWITCH

- Actualiza un bloque del tipo 'Manual Switch' agregado al panel de trabajo, aplicando validaciones especiales

getElemPanel

- Retorna el bloque (clip de película) que se haya solicitado de acuerdo a su identificador ingresado

getElemPanelTotal

- Retorna el número total de bloques que hayan sido agregados en la sesión

getElemPanelPorVentana

- Retorna el bloque (clip de película) que se haya solicitado de acuerdo a una ventana (clip de película) ingresada

getElemPanelPorSesionID

- Retorna el bloque (clip de película del bloque) que se haya solicitado de acuerdo a su identificador registrado en una sesión previa

hayBloque

- Indica (booleano) si el bloque consultado (tipo de bloque) está siendo utilizado o no en la sesión

2.2.6.1.3 Estructura

crearMC

- Permite agregar clips de película en el sistema

crearMC_EP

- Permite agregar clips de película en el sistema utilizados para crear los bloques del sistema

iniciarArrastreEP

- Permite arrastrar y desplazar el bloque en el panel de trabajo

setBlockImage

- Permite asignar una imagen de presentación a un bloque

addEliminarEP

- Crea y asigna el botón de eliminar bloque a un bloque

addPropiedadesEP

- Crea y asigna el botón de propiedades a un bloque

addRotarEP

- Crea y asigna el botón de rotar bloque a un bloque

addPuertosEP

- Crea y asigna los puertos de entrada y de salida a un bloque

addPuertosEP_ADICIONAL

- Crea y asigna los puertos de entrada y de salida (dada una validación especial predeterminada) a un bloque

addVentanaEP

- Crea y asigna la ventana de configuración de parámetros a un bloque.

addImagenEP

- Asigna la imagen representativa a un bloque

addParametros

- Asigna los parámetros a la estructura de datos de un bloque

esValidoEP

- Indica (booleano) si el bloque consultado (identificador del bloque) es válido o no

esValidoENTRADA

- Indica (booleano) si el bloque consultado (identificador del puerto de entrada) es válido o no

esValidoLINEA

- Indica (booleano) si la línea de conexión consultada (identificador de la línea) es válida o no

mostrarPropiedadesEP

- Muestra la ventana de configuración de parámetros del bloque consultado

rotarEP

- Rota horizontalmente el bloque indicado

estaRotado

- Indica (número) si el bloque consultado (clip de película del bloque) está rotado o no
    - Retorna 0 ó 2: Estado normal
    - Retorna 1 ó 3: Estado rotado

eliminarEP

- Elimina el bloque indicado (clip de película del bloque)
- Elimina las líneas de conexión asociadas al bloque eliminado
- Realiza acciones predeterminadas a bloques indicados

ocultarEliminarEP

- Oculta los botones eliminar bloque de todos los bloques agregados en la sesión

ocultarEliminarEPporID

- Oculta el botón eliminar bloque del bloque indicado (identificador del bloque)

ocultarEliminarEP_Subsistemas

- Oculta los botones eliminar bloque de los bloques predeterminados de los subsistemas

mostrarEliminarEP

- Muestra los botones eliminar bloque de todos los bloques agregados en la sesión

2.2.6.1.4 Datos

agregarElementoAlAccordion

- Permite agregar de forma automática los bloques (datos externos importados) a la librería de bloques
- Permite determinar la posición de los bloques agregados en la librería de bloques

addElementos_BLOQUES

- Obtiene los datos externos de los bloques
- Permite agregar los datos y parámetros al bloque indicado

modificarEP

- Permite realizar modificaciones predeterminadas en los bloques

load_XML_Grupos_Bloques

- Importa, parsea y almacena los datos externos de los grupos, bloques y parámetros en el sistema

get_GruposMenu

- Obtiene un arreglo con los nombres de los grupos importados de los datos externos

get_BloquesMenu

- Obtiene un arreglo con los nombres de los bloques importados de los datos externos utilizando el nombre del grupo a obtener sus bloques

get_BloquesDatos

- Obtiene un arreglo con los datos de los bloques importados de los datos externos utilizando el nombre del bloque a obtener sus datos

is_BloquesDatos_Enable

- Indica (booleano) si el bloque consultado (tipo de bloque) está habilitado o no

get_BloquesParametros

- Obtiene un arreglo con los datos de los parámetros importados de los datos externos utilizando el nombre del bloque a obtener sus parámetros

get_BloquesParametros_START

- Obtiene un arreglo con los datos de los parámetros importados de los datos externos utilizando el nombre del bloque a obtener sus parámetros y el tipo de evento inicial

get_BloquesParametros_ESPECIAL

- Obtiene un arreglo con los datos de los parámetros importados de los datos externos utilizando el tipo de evento inicial, nombre del bloque a obtener sus parámetros, y la variable a ser modificada

load_XML_Variables_Validation

- Importa, parsea y almacena los datos externos de las validaciones de variables del sistema

get_Variable_Validation

- Obtiene un arreglo con los datos de las validaciones de variable importados de los datos externos utilizando el nombre de la variable y el tipo de parámetro

replace_Parameter_Description

- Reemplaza las combinaciones especiales de textos de la descripción obtenida de los datos externos, para mejorar su presentación en la vista de propiedades del bloque

2.2.6.1.5 Parámetros

agregarParametrosVentana

- Agrega los parámetros de un bloque a su ventana asignada a
- Automatiza las posiciones de los diferentes elementos de la ventana
- Realiza diferentes validaciones según el tipo de parámetro

setParamDescripcion

- Asigna la descripción de un bloque y su formato a la ventana de propiedades asignada del bloque

setParamValidation

- Asigna las validaciones de variables a los parámetros obtenidos de los datos externos

addBotonOK

- Agrega el botón de aceptar cambios en la ventana de propiedades del bloque
- Realiza acciones predeterminadas según cada tipo de parámetro del bloque al hacer clic sobre este botón

addBotonCANCEL

- Agrega el botón de cancelar cambios en la ventana de propiedades del bloque
- Realiza acciones predeterminadas según cada tipo de parámetro del bloque al hacer clic sobre este botón

getParameterFaderGain_LEVEL

- Obtiene el nivel del regulador deslizante de ganancia indicando el bloque (clip de película)

setParameterFaderGain_GAIN

- Asigna el valor de la ganancia indicando el bloque (clip de película) y el nivel del regulador deslizante de ganancia

updateParameterFaderGain

- Actualiza los valores del parámetro del regulador deslizante de ganancia y los campos de entrada

set_ParametersEvents

- Asigna validaciones especiales predeterminadas según las variables y tipo de bloque, permitiendo agregar eventos en los parámetros para mejorar la interacción entre diferentes parámetros

updateBlockImageTotal

- Actualiza la imagen de presentación de un bloque, verificando todos sus posibles eventos según las configuraciones de sus parámetros

updateBlockPortsTotal

- Actualiza los puertos de entrada y de salida de un bloque, verificando todos sus posibles eventos según las configuraciones de sus parámetros.

2.2.6.1.6 Sesión

cargarSesion_SCOPE

- Asigna los datos del graficador obtenidos de importar la sesión almacenada

cargarSesion_ADD

- Agrega el bloque obtenido de importar la sesión almacenada

cargarSesion_PARAM

- Asigna al bloque importado, los valores de sus parámetros obtenidos de importar la sesión almacenada

cargarSesion_LINE

- Agrega la línea de conexión entre bloques obtenidos de importar la sesión almacenada

sesion_generaCMD_EXPORT_XML

- Retorna un texto con el formato XML con los datos estructurados de la sesión a almacenar

sesion_LOAD_XML

- Importa, parsea y gestiona los datos externos de la sesión almacenada, indicando el nombre de la sesión a cargar

sesion_separarCMD

- Obtiene un arreglo de textos, teniendo como base un solo texto y un patrón para dividirlo

sesion_LOAD

- Si la sesión del usuario es válida, prepara y carga la sesión almacenada previamente

sesion_SAVE

- Si la sesión del usuario es válida, prepara y almacena la sesión en el sistema

sesion_LOAD_USER_DATA

- Indica (booleano) si la sesión del usuario es válida o no
- Asigna el nombre de usuario y número de práctica activo a la variable global

sesion_DESHABILITAR_PANEL

- Si la sesión del usuario no es válida, deshabilita algunos elementos importantes del sistema

workspace_SAVE

- Si la sesión del usuario es válida, prepara y almacena el espacio de trabajo (workspace) de la sesión del sistema.

2.2.6.1.7 Exportar

exportarDatos_generaCMD

- Prepara líneas de código que entiende el servidor MSP para interpretar la creación del modelo al momento de iniciar la simulación
    - Recorre el arreglo de bloques agregados a la sesión, verificando que sean válidos
        - Diferencia el nivel de cada bloque para validar la creación del modelo con subsistemas
        - Prepara los comandos 'add_block' con los bloques válidos activos para transmitirlos al servidor y genere los requerimientos del usuario
        - Registra la posición relativa del bloque en el panel
    - Recorre el arreglo de bloques agregados a la sesión, verificando que sean válidos y consultando si algún parámetro ha sido modificado

        - Diferencia el nivel de cada bloque para validar la creación del modelo con subsistemas
        - Prepara los comandos 'set_param' con los bloques válidos activos para transmitirlos al servidor y genere los requerimientos del usuario
    - Recorre el arreglo de líneas de conexión establecidas en la sesión, verificando que sean válidas
        - Diferencia el nivel de cada bloque para validar la creación del modelo con subsistemas
        - Prepara los comandos 'add_line' con los bloques válidos activos para transmitirlos al servidor y genere los requerimientos del usuario
- Retorna un conjunto de líneas de código con formato estructurado para su comunicación e interpretación con el servidor

exportarDatos_generaCMD_Parametros

- Prepara líneas de código que entiende el servidor MSP para interpretar las modificaciones de parámetros del modelo al tener iniciada la simulación

exportarDatos_Iniciar

- Se comunica con el servidor e inicia los procesos de simulación, transfiriendo las líneas de códigos interpretadas del modelo generado
- De generarse algún error por parte del usuario al crear el modelo, se reporta sobre las posibles causas

exportarDatos_Pausar

- Se comunica con el servidor y pausa los procesos de simulación

exportarDatos_Detener

- Se comunica con el servidor y detiene los procesos de simulación

exportarDatos_CambiarParametroAgregar

- Prepara líneas de código que entiende el servidor MSP para interpretar las modificaciones de parámetros del bloque y variable indicadas

exportarDatos_CambiarParametro

- Se comunica con el servidor y envía las líneas de código que entiende el servidor MSP para interpretar las modificaciones de parámetros del modelo al tener iniciada la simulación

exportarDatos_CambiarParametroTodos

- Se comunica con el servidor y envía las líneas de código que entiende el servidor MSP para interpretar todos los parámetros del modelo al tener iniciada la simulación

showErrorList

- Muestra los errores generados al iniciar la simulación

str_replace

- Permite reemplazar cadenas de texto

2.2.6.2. Componentes

Se detallan los procedimientos según los elementos o componentes principales utilizados en el sistema.

2.2.6.2.1. Puertos

addEntradas

- Agrega puertos de entrada al sistema y los asigna al bloque indicado

getEntradas

- Retorna el puerto de entrada (clip de película) que se haya solicitado de acuerdo a su identificador ingresado

getEntradasTotal

- Retorna el número total de puertos de salida que hayan sido agregadas en la sesión

getEntradasPorElementoPosicion

- Retorna el puerto de entrada (clip de película) que se haya solicitado de acuerdo a su posición y al identificador del bloque contenedor

getNumeroEntradasElemento

- Retorna el número de puertos de entrada del bloque (clip de película) indicado

showEntradasN

- Permite cambiar el número de puertos de entrada que se muestran en el bloque (clip de película) indicado

showSalidasN

- Permite cambiar el número de puertos de salida que se muestran en el bloque (clip de película) indicado

isBloqueConectado

- Indica (booleano) si el bloque está conectado algún otro bloque o no

isBloqueEntradasConectadas

- Indica (booleano) si algún puerto de entrada del bloque está conectado o no

isBloqueSalidasConectadas

- Indica (booleano) si algún puerto de salida del bloque está conectado o no

addSalidas

- Agrega puertos de salida al sistema y los asigna al bloque indicado

getSalidas

- Retorna el puerto de salida (clip de película) que se haya solicitado de acuerdo a su identificador ingresado

getSalidasTotal

- Retorna el número total de puertos de salida que hayan sido agregadas en la sesión

getSalidasPorElementoPosicion

- Retorna el puerto de salida (clip de película) que se haya solicitado de acuerdo a su posición y al identificador del bloque contenedor

getNumeroSalidasElemento

- Retorna el número de puertos de salida del bloque (clip de película) indicado

moverPuertoSalida_LOAD

- Registra la posición relativa inicial del nodo de conexión del puerto de salida en el momento de intentar establecer una conexión entre bloques

moverPuertoSalida_ENTERFRAME

- Registra la posición relativa durante el desplazamiento del puerto nodo de conexión del puerto de salida en el momento de intentar establecer una conexión entre bloques

moverPuertoSalida_PRESS

- Habilita la opción de desplazamiento del nodo de conexión del puerto de salida y oculta la librería de bloques

moverPuertoSalida_RELEASE

- Valida los posibles casos para permitir establecer conexión entre bloques
- Si el puerto de entrada está disponible
    - No
        - El nodo de conexión del puerto de salida regresa a su posición relativa inicial
    - Si
        - Se establece línea de conexión y el nodo de conexión del puerto de salida regresa a su posición relativa inicial
- Se muestra la librería de bloques

2.2.6.2.2 Líneas

crearMC_Linea

- Permite agregar clips de película de líneas en el sistema

crearLineaUnion

- Grafica diferentes líneas de conexión entre bloques
- Utiliza el color asignado al bloque del puerto de salida
- Agrega varios nodos intermedios para generar las líneas de conexión
- Automatiza la forma en que se grafican las líneas en base a la posición y ubicación del bloque con el puerto de salida y el bloque de la puerta de entrada
- Habilita las opciones para poder eliminar la línea de conexión

getRedibujarLineas

- Retorna el estado (booleano) de que si el sistema debe redibujar las líneas de conexión o no

setRedibujarLineas

- Asigna el estado (booleano) de que si el sistema debe redibujar las líneas de conexión o no

RedibujarLineasComenzar

- Comienza el proceso de redibujar líneas de conexión en el sistema

RedibujarLineasDetener

- Detiene el proceso de redibujar líneas de conexión en el sistema

RedibujarLineasDetener_tiempo

- Detiene el proceso de redibujar líneas de conexión en el sistema
- Se utiliza en conjunto a una función que se activa en determinado tiempo

getColorLinea

- Obtiene el color activo actual para poder ser asignado a un bloque
-

addLineas

- Registra la línea de conexión entre bloques que ha sido creada

getLineas

- Retorna la línea de conexión (clip de película) que se haya solicitado de acuerdo a su identificador ingresado

getLineasTotal

- Retorna el número total de líneas de conexión que hayan sido agregadas en la sesión

eliminarLineas

- Elimina todas las líneas de conexión asociadas al bloque solicitado por su identificador

eliminarLineasPorEntrada

- Elimina todas las líneas de conexión asociadas al puerto de entrada, solicitado por su identificador

enlazarBloques_BaseDestino

- Asigna los datos de la conexión entre bloques a la línea creada indicando el bloque base y el puerto de entrada destino

enlazarBloques

- Asigna los datos de la conexión entre bloques a la línea creada indicando los puertos de entrada, de salida y su respectiva posición

2.2.6.2.3 Ventanas

mostrarVentana

- Crea la ventana de configuración de parámetros del bloque
- Agrega los parámetros del bloque
- Valida según el tipo de bloque diferentes acciones predeterminadas

2.2.6.2.4 Librería de bloques

agregarAccordion

- Agrega y configura el componente 'Accordion' utilizado para mostrar la librería de bloques disponibles para crear los modelos
- Automatiza el proceso de agregar bloques categorizados por grupos a la librería de bloques

setPracticaBloque

- Se comunica con el servidor y asigna la práctica activa al sistema
- Actualiza la librería de bloques acorde la práctica activa

getPracticaBloque

- Obtiene el valor de la práctica activa del sistema

show_libraryBrowser

- Muestra la librería de bloques

hide_libraryBrowser

- Oculta la librería de bloques

2.2.6.2.5 Menú de opciones

agregarMenuBar

- Agrega y configura el componente 'menuBar' utilizado para mostrar la barra del menú de opciones del sistema
- Automatiza el proceso de agregar bloques categorizados por grupos a la librería de bloques

file_new

- Prepara el sistema para reiniciar la creación de un modelo

view_libraryBrowser

- Permite mostrar u ocultar la librería de bloques

simulation_start

- Si la sesión del usuario y el estado de la sesión del sistema es válida, permite iniciar el proceso de simulación

simulation_pause

- Permite pausar el proceso de simulación

simulation_stop

- Permite detener el proceso de simulación

menu_refresh

- Actualiza el menú de opciones

2.2.6.2.6 Botones

boton_play

- Al ser presionado, inicia la simulación

boton_pause

- Al ser presionado, pausa la simulación

boton_stop

- Al ser presionado, detiene la simulación

tiempo_txt

- Al ser cambiado, actualiza el valor del eje X del graficador

btn_show_library

- Al ser presionado, muestra u oculta la librería de bloques

2.2.6.2.7. Graficador

addGraficador

- Agrega y configura el clip de película 'GrafBase' que permite configurar y mustrar el graficador del sistema

mostrarGraficador

- Permite mostrar u ocultar el graficador

ocultarGraficador

- Oculta el graficador

setNumeroFuncionesGraficador

- Verifica las líneas de conexión establecidas en los puertos de entrada del graficador, para configurar y habilitar la cantidad de funciones a mostrar

orderTagsLabel

- Verifica las líneas de conexión establecidas en los puertos de entrada del graficador, para configurar y organizar las etiquetas de los eje Y de las funciones

getWebScope

- Obtiene el bloque (clip de película) que corresponde al graficador agregado al sistema

Variables del clip de película 'GrafBase'

Procedimientos del clip de película 'GrafBase'

drawBoard

- Dibuja el área del tablero

drawAxes

- Dibuja los ejes de coordenadas del graficador

drawGraph

- Dibuja los arreglos de puntos coordenadas en el graficador

xtoPix

- Convierte la unidad numérica del punto X a pixeles

ytoPix

- Convierte la unidad numérica del punto Y a pixeles

cambiarEjes

- Actualiza los valores del eje X y del eje Y

cambiarEjes_X

- Actualiza los valores del eje X

cambiarEjes_Y

- Actualiza los valores del eje Y

getCambiarEjes_y

- Proceso que retorna los valores del eje Y con formato preestablecido

iniciarGrafico

- Inicia el proceso de graficar funciones

detenerGrafico

- Detiene el proceso de graficar funciones

resetGrafico

- Reinicia a los valores iniciales el proceso de graficar funciones

roundDecimal

- Redondea una cifra numérica a decimal indicado

CargarXML

- Prepara y carga el XML obtenido de comunicarse con el servidor y la base de datos del sistema

cargar_XML_Real

- Procesa los datos obtenidos de los puntos coordenada como una simulación en tiempo real
- Almacena los datos disponibles de los puntos coordenada más recientes y que no hayan sido graficados

cargar_XML_Reset

- Se comunica con el servidor y limpia los datos de los puntos coordenada almacenados en la base de datos del usuario y número de práctica activo

graficarArreglo

- Utiliza los datos almacenados de los puntos coordenada, prepara y permite graficar las diferentes funciones disponibles en el sistema

get_nSubDiv

- Obtiene la cantidad de subdivisiones de un entero, con el que se realiza la simulación

get_nSubDiv_MANUAL

- Obtiene la cantidad de subdivisiones de un entero, con el que se realiza la simulación, ingresando dos valores manualmente

get_tiempoTotalXML

- Obtiene el tiempo total de los datos de los puntos coordenada almacenados

get_escalaY

- Obtiene la escala de la función indicada por el arreglo de escalas y su identificador de función

scopeEntradaEstaConectada

- Indica (booleano) si el puerto de entrada del graficador de la función indicada está conectado o no

initAutoY

- Inicia los valores para auto asignar los valores de las diferentes escalas del eje Y

setAutoY

- Auto asignar los valores de las diferentes escalas del eje Y en base los últimos puntos coordenada procesados

getAutoY

- Actualiza los valores configurables de los parámetros mínimo y máximo de los ejes Y de las funciones habilitadas

resetAutoY

- Asigna valores predefinidos a los valores configurables de los parámetros mínimo y máximo de los ejes Y de las funciones habilitadas

minimoArray

- Obtiene el valor mínimo del arreglo

maximoArray

- Obtiene el valor máximo del arreglo

2.2.6.2.8 Subsistemas

getNivelActual

- Obtiene el nivel (subsistema) en el que el usuario se encuentra

getNivel

- Obtiene el nivel al que pertenece el bloque (clip de película) que se indica

hideSubsistema

- Oculta el subsistema al cual pertenece el bloque indicado por su identificador

showSubsistema

- Muestra el subsistema al cual pertenece el bloque indicado por su identificador

addSubSistema

- Agrega un subsistema a la sesión
- Agrega y configura los bloques básicos que pertenecen a cada subsistema

addSubSistema_SESION

- Agrega un subsistema a la sesión importada

getSubSistemas

- Obtiene el subsistema (clip de película) indicado por su identificador

getSubSistemasPorPadre

- Obtiene el subsistema (clip de película) indicado por el identificador del padre contenedor

getSubSistemasTotal

- Retorna el número total de subsistemas hayan sido agregadas en la sesión

eliminarSubSistemas

- Elimina el subsistema indicado por el identificador del padre contenedor
- Elimina todos los elementos contenidos en el subsistema

hideBloquesTodos

- Oculta todos los bloques y líneas de conexión agregadas a la sesión

showBloquesTodos

- Muestra todos los bloques y líneas de conexión agregadas a la sesión

showBloquesPorNivel

- Muestra todos los bloques y líneas de conexión agregadas a la sesión que pertenezcan al nivel indicado.

2.2.7 Diseño de pruebas

Para probar el funcionamiento del sistema LabCon, básicamente se realizaron dos tipos de pruebas con las cuales se analizó el comportamiento del sistema bajo condiciones de uso real cuyos resultados se mostraran en el capítulo siguiente.

2.2.7.1 Pruebas de usabilidad

Este tipo de prueba se enfoca básicamente en verificar que tan fácil e intuitivo resulta el uso del sistema para un usuario el cual desconoce la aplicación. Para realizar esta prueba se tuvo en consideración los siguientes aspectos.

- Se selecciona un grupo de estudiantes, los cuales desconocen totalmente el uso de la aplicación
- Se les entrega una práctica con su debida documentación, la cual los estudiantes tienen que reproducir y ejecutar

- Se debe tener muy en cuenta que el usuario deberá resolver todos los problemas por sí mismo sin intervención de ninguna persona
- Luego de esta prueba se realizara una encuesta al usuario con la finalidad de obtener sugerencias sobre posibles mejoras al sistema

2.2.7.2 Pruebas de rendimiento

La finalidad de esta prueba es verificar el rendimiento en general del sistema, tomando como puntos críticos el tiempo de respuesta y el porcentaje de procesamiento que se genera al atender los requerimientos de los usuarios, para llevarla a cabo, se tratará de simular la peor de las situaciones, la cual se consideró sería el acceso simultaneo de múltiples usuarios los cuales hacen requerimientos a la vez; para llevarla a cabo se tomó en consideración los siguientes aspectos.

- Debido a que el servidor debe realizar procesamiento de datos para poder atender los requerimientos de los usuarios, se trató de saturar al sistema con múltiples requerimientos simultáneos.
- Se monitorea el porcentaje de memoria usado y el uso de los procesadores para observar posibles procesos que saturen o

puedan perjudicar el rendimiento del servidor, de existir alguno se deberá tratar de optimizar.

- Se debe tomar muy en consideración los tiempos de respuesta del servidor a los requerimientos de los usuarios, de ser estos muy elevados se deberá de depurar la aplicación para reducir estos tiempos.

CAPÍTULO III

3. Ejecución Remota

3.1 Procesos del Sistema para Creación y Manipulación de Modelos (Flash Panel)

Para entender el funcionamiento de como el flash panel manipula e interpreta los requerimientos de los usuarios para crear y manipular los modelos vía web, se detallan brevemente los principales procesos y secuencias que son utilizadas para su funcionamiento.

Se describen los procesos para manipulación de bloques, líneas, sesión y simulación.

3.1.1 Inicialización del sistema

- Se inicia el sistema
- Se inicializan variables internas del sistema
- Se verifica estado de la sesión activa

- Se consulta script externo en el servidor para obtener datos de la sesión activa
- El script retorna mensajes de estado válidos para el flash

- ¿La sesión del usuario es válida?
  - Si
    - Se cargan datos externos (XML) del sistema: Grupos, bloques, validaciones de variables del bloque
      - Se carga la librería de bloques (Accordion)
        - Se crean bloques organizados por grupos en la librería de bloques, validando los datos según la práctica
    - Se carga el menú de opciones
    - Se habilitan los botones de ejecución
      - Se oculta el botón de detener
    - Se agrega el sistema graficador a la sesión activa
      - Se oculta el graficador hasta que se cumplan los requisitos básicos para su uso
  - No
    - No se muestra la librería de bloques
    - No se muestra el menú de opciones
    - No se muestran los botones de ejecución

Se muestra un mensaje de error, indicando que debe iniciar sesión

3.1.1.1 Carga de datos externos (XML)

- Se carga el XML que contiene los datos de grupos, bloques y parámetros de bloques al sistema
- ¿La carga del XML es válida?
  - Si
    - Se parsea los datos del XML, recorriendo inicialmente el primer nivel de grupos y almacenando el arreglo de grupos en el sistema
      - Por cada nodo grupo, se recorre sus nodos hijos, y se almacena en arreglo de bloques en el sistema
        - Por cada nodo bloque, se recorre sus nodos hijos, y se almacena en arreglo de parámetros del bloque en el sistema
          - Según el tipo de parámetro, se procede de diferentes formas predeterminadas su almacenamiento en el sistema
  - No
    - No se muestra la librería de bloques
    - No se muestra el menú de opciones
    - No se muestran los botones de ejecución
- Se carga y actualiza la librería de bloques

3.1.1.2 Carga de la librería de bloques

-Se importa el componente "Accordion" y se configuran sus parámetros

-Utilizando los datos externos cargados (XML) se crean los diferentes bloques

- Por cada bloque se importa el clip de película "EM_Importar" el cual permite agregar bloques de forma automática a la librería de bloques
- Agrega los bloques categorizándolos por el grupo al que pertenece
- Asigna el orden y ubicación del bloque en la librería de forma automática
- Habilita los bloques validando el número de práctica activa
- Valida acciones a realizar con los bloques según su tipo
- Asigna y configura los bloques según los datos externos cargados
    - Número de puertos de entrada y salida del bloque
    - Nombre del bloque
    - Imagen de presentación del bloque
    - Descripción y parámetros del bloque

3.1.2 Bloques

Procesos para entender el funcionamiento de cómo se agregan los bloques, se manipula su contenido y eliminación.

3.1.2.1 Cargar un bloque

- Se agrega un bloque a la librería de bloques
    - Se asigna la imagen representativa del bloque
    - Se habilita acciones al hacer clic sobre el bloque
        - Según el tipo de bloque, se valida diferentes acciones predeterminadas para agregar un bloque al panel de trabajo
        - Agrega el bloque seleccionado al panel de trabajo y lo mantiene en estado de arrastre y desplazamiento hasta que se haga clic en la ubicación donde se desee ubicar
        - Agrega parámetros al bloque seleccionado
        - Según el tipo de bloque, se valida diferentes acciones predeterminadas de modificación del bloque
    - Verifica si el bloque está habilitado para ser utilizado en la sesión del sistema
    - Asigna los textos descriptivos al bloque

- Se agrega un bloque al panel de trabajo
    - Se asignan los datos externos importados del XML
    - Se asigna la imagen representativa del bloque
    - Se agrega la ventana de configuración de parámetros del bloque
    - Se agrega el botón de eliminar bloque
    - Se agrega el botón de propiedades del bloque
    - Se agrega el botón de rotar bloque
    - Asigna los textos descriptivos al bloque
    - Agrega los puertos de entrada y de salida al bloque

3.1.2.2 Cambiar parámetros (propiedades) de un bloque

-Para mostrar la ventana de configuración de parámetros se accede haciendo clic sobre el botón de propiedades o doble clic sobre la imagen representativa del bloque

-Se inicia el proceso de mostrar ventana de configuración de parámetros del bloque
    - Según el tipo de bloque, se realiza acciones predeterminadas
    - Se agrega y configura la ventana al sistema

- Importa el clip de película 'PAR_VEN_Importar' el cual dependiendo del tipo de bloque, agrega y muestra todos los parámetros que pertenecen al bloque

  - Recorre el arreglo de parámetros del bloque, según el tipo de parámetro realiza acciones y validaciones predeterminadas para mostrarlo en la ventana
  - Agrega los botones de aceptar y cancelar

3.1.2.3 Eliminar un bloque

-El proceso se inicia el hacer clic sobre el botón de eliminar bloque

-Elimina todas las líneas de conexión asociadas el bloque

  - Elimina líneas de conexión de los puertos de entrada
  - Elimina líneas de conexión de los puertos de salida

-Realiza acciones predeterminadas según el tipo de bloque a eliminar

  - Si es un graficador, lo oculta
  - Si es un subsistema, elimina todos sus elementos contenidos

-Actualiza el número de funciones que están habilitadas en el graficador.

3.1.3 Líneas de conexión

Procesos para entender el funcionamiento de cómo se agregan y establecen las líneas de conexión entre bloques, su manipulación y eliminación.

3.1.3.1 Enlazar dos bloques

- El usuario selecciona y arrastra el nodo de conexión del puerto de salida del bloque y lo suelta en el puerto de entrada del bloque destino, ¿existe contacto?
  - Si
    - ¿El puerto de entrada del bloque destino está libre? ¿Se intenta establecer un enlace de conexión con distintos bloques?
      - Si
        - Se agrega la línea de conexión entre los nodos conectados
        - Se cambia el estado del puerto de entrada conectado a 'no disponible'
        - Se dibuja las líneas de conexión entre los nodos conectados, relativas a su posición en el panel

- No
    - El nodo de conexión regresa a su punto de partida origen

- No
    - El nodo de conexión regresa a su punto de partida origen

3.1.3.2 Eliminar una línea

-Para eliminar una línea de conexión, puede ocurrir en diferentes casos:

- Al hacer doble clic sobre la línea de conexión
    - Elimina la línea, cambia el estado de del puerto de entrada en la que estaba conectada a 'disponible'
- Eliminar un bloque que tenía establecidas líneas de conexión
    - Recorre la lista de líneas válidas que han sido agregadas al sistema, y busca la que coincida con el identificador de bloque que ha sido indicado
    - Elimina la línea, cambia el estado de del puerto de entrada en la que estaba conectada a 'disponible'

3.1.4 Sesiones

Procesos para entender el funcionamiento de cómo se guarda una sesión y carga sesiones almacenadas en el sistema

3.1.4.1 Guardar sesión

- Se inicia el proceso de guardar la sesión del modelo creado
- El proceso continúa si la sesión del usuario es válida
- Genera las líneas de código en formato XML que van a ser almacenadas en el servidor
    - Recorre y almacena los datos de configuración del tiempo y variables del graficador en el panel indicando el evento 'scope '
    - Recorre el arreglo de bloques agregados a la sesión y verifica que sean válidos
        - Almacena las variables que indican su tipo, ubicación, nivel en el panel indicando el evento 'add_block'
    - Recorre el arreglo de bloques agregados a la sesión y verifica que sean válidos
        - Verifica el arreglo de sus parámetros y verifica que hayan sido modificados sus valores originales

        - Almacena las variables que indican los parámetros modificados del bloque agregado en el panel indicando el evento 'set_param'
    - Recorre el arreglo de líneas agregadas a la sesión y verifica que sean válidas
        - Verifica que sean válidos los puertos de entradas de las líneas
            - Almacena las variables que indican las línea de conexión establecidas entre bloques agregados en el panel indicando el evento 'add_line'

- Se establece comunicación con un script externo en el servidor para enviar los requerimientos del proceso
    - El script consultado retorna mensajes de estado válidos para el flash
    - ¿La conexión establecida con el servidor fue exitosa?
        - Si
            - Genera un archivo XML con los datos estructurados de la sesión transmitidos por el flash panel

- Le indica al usuario que todo el proceso fue exitoso

▪ No

- Le indica al usuario que existió algún inconveniente en el proceso

3.1.4.2 Cargar sesión

- Se inicia el proceso de cargar la sesión del modelo almacenado
- El proceso continúa si la sesión del usuario es válida
- Se prepara y se carga el XML de la sesión
  - ¿Es exitosa la carga del XML de la sesión?
    - Si
      - Se limpia el modelo de la sesión actual
      - Se recorre y parsea el XML de la sesión
      - Según el tipo de evento (scope, add_block, set_param, add_line) se realizan acciones predeterminadas para cargar la sesión almacenada
    - No

Se detiene el proceso de carga y se le indica al usuario que ha ocurrido un error.

3.1.5 Simulación

Procesos para entender el funcionamiento de iniciar, pausar, reanudar y detener la simulación.

3.1.5.1 Iniciar simulación

- Comienza el proceso de iniciar la simulación
- El proceso continúa si la sesión del usuario es válida
- El proceso continúa si el valor del tiempo es numérico
- Se realizan modificaciones en los botones del panel
- Se oculta la librería de bloques
- Se limpian los datos del graficador
- El proceso continúa si el usuario ha agregado un bloque del graficador
- Se inhabilitan los botones de eliminar bloque de todos los bloques válidos
- Se inhabilita la propiedad de arrastrar y mover de todos los bloques válidos

- Se inhabilita la opción de poder establecer nuevas líneas de conexión
- Se inhabilita la opción de poder eliminar líneas de conexión
- Se genera las líneas de código que contienen la información estructurada de los datos de la sesión
- Se establece comunicación con un script externo en el servidor para enviar los requerimientos del proceso
- ¿La comunicación se realiza exitosamente?
    - Si
        - Se transmite las líneas de código con información de la sesión
        - El script consultado hace uso de las líneas de código transmitidas
            - Retorna un mensaje al panel flash que lo interpreta para indicar el estado de la simulación
        - El script externo del servidor, establece comunicación con el MATLAB Server Page y genera el modelo interpretado

- El modelo generado por el MSP inicia su simulación
    - Se establece conexión con la base de datos y almacena de forma estructurada los datos de los puntos coordenada de las funciones activas del graficador
- El flash panel al recibir por parte del script externo del servidor que la simulación se ha iniciado exitosamente
    - Se inicia de forma continua las consultas a un script externo que establece conexión con la base de datos del sistema
        - Los datos obtenidos de la base, los retorna en formato XML que interpreta el flash panel
        - El flash panel recibe, parcea y almacena los datos obtenidos del XML de los puntos
    - Se inicia el proceso de simulación y su graficación

- No

- Se indica la usuario de que se ha generado un error en el proceso

3.1.5.2 Pausar simulación

- Comienza el proceso de pausa de la simulación
- Se realizan modificaciones en los botones del panel
- Se establece comunicación con un script externo en el servidor para enviar los requerimientos del proceso
    - ¿La comunicación se realiza exitosamente?
        - Si
            - Se transmite las líneas de código con información de la sesión
            - El script consultado hace uso de las líneas de código transmitidas
                - Retorna un mensaje al panel flash que lo interpreta para indicar el estado de la simulación

- Se pausa el proceso de simulación y su graficación

▪ No

- Se indica la usuario de que se ha generado un error en el proceso

3.1.5.3 Reanudar simulación

Comienza el proceso de reanudación de la simulación

- Se realizan modificaciones en los botones del panel
- Se establece comunicación con un script externo en el servidor para enviar los requerimientos del proceso.

o ¿La comunicación se realiza exitosamente?

▪ Si

- Se transmite las líneas de código con información de la sesión
- El script consultado hace uso de las líneas de código transmitidas

- Retorna un mensaje al panel flash que lo interpreta para indicar el estado de la simulación

- Se reanuda el proceso de simulación y su graficación

- No

- Se indica la usuario de que se ha generado un error en el proceso

3.1.5.4 Detener simulación

- Comienza el proceso de detener la simulación
- Se realizan modificaciones en los botones del panel
- Se muestra la librería de bloques
- Se limpian los datos del graficador
- Se habilitan los botones de eliminar bloque de todos los bloques válidos
- Se habilita la propiedad de arrastrar y mover de todos los bloques válidos
- Se habilita la opción de poder establecer nuevas líneas de conexión
- Se habilita la opción de poder eliminar líneas de conexión

- Se establece comunicación con un script externo en el servidor para enviar los requerimientos del proceso
    - ¿La comunicación se realiza exitosamente?
        - Si
            - Se transmite las líneas de código con información de la sesión
            - El script consultado hace uso de las líneas de código transmitidas
                - Retorna un mensaje al panel flash que lo interpreta para indicar el estado de la simulación
            - El modelo generado por el MSP detiene su simulación
            - El flash panel recibe por parte del script externo del servidor que la simulación se ha detenido exitosamente
        - No
            - Se indica la usuario de que se ha generado un error en el proceso

3.2 Procesos del sistema para la interpretación y manipulación de datos (nivel del servidor)

Una vez que el cliente ha realizado todas las tareas en el navegador se envían los datos al servidor para ser interpretados, como un conjunto de instrucciones la cual se ejecutan secuencialmente una luego de otra dependiendo, de la instrucción a ejecutarse el sistema realiza una operación especifica.

3.2.1 Creación de componentes

- Si la instrucción es de agregar un componente
- Se selecciona el nombre del bloque de la instrucción
- Se ejecuta el comando para la inserción de un bloque siguiendo el siguiente formato add_block('Librería/Bloque', 'Modelo/Nombre') donde "**Librería"** hace referencia al a librería de componentes **"Bloque"** es el componente que se desea agregar **"Modelo"** es el modelo del usuario al cual se agregara el componente y **"Nombre"** es el nombre con el cual deseamos se inserte el componente al modelo del usuario.

3.2.2 Modificación de parámetros de componentes

- Si la instrucción es la de modificar componentes.
- Se selecciona el nombre del componente, parámetro a modificar y los valores a ingresar de la instrucción.
- Se ejecuta el comando para la modificación de parámetros siguiendo el siguiente formato set_param('Modelo/Bloque','Parametro1','valor1','Parametro2','valor2') donde "'Modelo**"** hace referencia al modelo del usuario en ejecución **"Bloque"** es el componente al que se le desea modificar los parámetros **"Parametro1"** es el parámetro a ser modificado **"Valor1"** es el valor a ser ingresado en el parámetro 1, **"Parametro2"** es el parámetro a ser modificado **"Valor2"** es el valor a ser ingresado en el parámetro 2 , teniendo en consideración que si el bloque posee más parámetros se pueden configurar todos a la vez o uno por uno independientemente.

3.2.3 Interconexión de componentes

- Si la instrucción es de interconectar dos componentes.
- Se selecciona el nombre de los dos componentes a interconectar así como el punto de salida y el punto de entrada para realizar la conexión.

- Se ejecuta el comando para la conexión de dos bloques siguiendo el siguiente formato add_line('Modelo','BloqueInicio/numSalida','BloqueFin/numEntrada') donde **"'Modelo"** hace referencia al modelo del usuario **"BloqueInicio"** es el componente desde donde se empezara la conexión **"numSalida"** es un numero entero que especifica cuál es la salida que se está usando (1 ,2 ,3 …), **"BloqueFin"** es el componente con el cual terminará la conexión **"numEntrada"** es un numero entero que especifica cuál es la entrada que se está usando (1 ,2 ,3 …) .

3.2.4 Configuración del Run Time

- Si la instrucción es de configurar el Run Time (Tiempo de ejecución del modelo).
- Se selecciona tiempo el cual se desea ejecutar el modelo.
- Se ejecuta el comando para la modificación del Run Time siguiendo el siguiente formato set_param(Modelo,'Stop time','Tiempo'') donde **Modelo"** hace referencia al modelo del usuario **"Stop time"** es la variable tiempo con la cual Matlab identifica al tiempo de ejecución y **"Tiempo"** es el tiempo que se desea configurar el cual será ingresado en ms (milisegundos).

3.2.5 Iniciar, pausar, reanudar, detener simulación

- Si la instrucción es de iniciar, pausar, reanudar o detener la ejecución.
- Se selecciona la instrucción que se desea realizar.
- Se ejecuta el comando siguiendo el siguiente formato set_param(Modelo,'SimulationCommand','Start/Stop/Pause')

  donde **“Modelo”** hace referencia al modelo del usuario **“SimulationCommand”** es la variable con la cual Matlab identifica que se realizaran cambios en parámetros de simulación **“Start/Stop/Pause”** son las tres opciones de comandos que se pueden usar, cabe recalcar que solamente se usa uno a la vez para iniciar, parar, pausar.

3.3 Procesos del sistema para la manipulación de maquinaria (servidor - maquinaria)

3.3.1 Elementos usados para la manipulación

Una vez interpretados todos los datos enviados por el usuario hacia el servidor, este interpreta todos los datos procesa la información y tratara de ejecutar el modelo diseñado por el cliente de la siguiente manera

3.3.2 Software relacionado

- Apache Tomcat recibe mediante el protocolo http las instrucciones del usuario.
- La programación de LabCon convierte las instrucciones en código interpretable por Matlab.
- Matlab crea el modelo del usuario en un nuevo archivo .mdl con el nombre del usuario y el número de su práctica en base al código generado.
- Matlab ejecuta el modelo.

3.3.3 Hardware relacionado

- Las señales son enviadas mediante el cFP-2100 a los equipos.
- Se verifica que señal, en que canal y a que tarjeta del cFP-2100 va a ser enviada.
- Se modifican los valores.

3.3.4 Como se realizará esta manipulación

- El usuario modifica valores en la página web.
- Se envía al servidor para proceso.
- Servidor recibe la información.

- Sistema LabCon procesa información convierte en código interpretable por Matlab.
- Matlab interpreta código, genera modelo y ejecuta.
- Matlab mediante configuraciones internas se conecta al cFP-2100 mediante la librería OPC –toolbox.
- Se escriben datos en el cFP-2100.
- Mediante las tarjetas cFP-AO-210, y la tarjeta cFP-RLY-421 se escriben los datos.
- Se envía las señales a los equipos.

3.3.5 Seguridades

En cuanto a seguridades se refiere existen varias seguridades que permiten que los equipos no sufran daños.

- Supresores de pico a la entrada de las plantas (reguladores de voltaje).
- Fusibles previos a todos los cFP.
- Bloques de saturación previos a la entrada de datos para evitar niveles fuera de rango.
- Varios métodos diseñados para interrupción de los ejercicios.

3.4 Pruebas y análisis

Dentro del desarrollo de las pruebas, se llevaron a cabo de acuerdo a la planificación

Prueba de rendimiento

Se realizaron varias pruebas para comparar el rendimiento del servidor se realizaron pruebas comparativas con diversos sistemas operativos y diversos tipos de procesadores para probar el rendimiento en las peores situaciones, la cual es trabajar con múltiples usuarios conectándose al mismo tiempo, y ejecutando procesos en el servidor.

Análisis de la prueba de rendimiento

Se observó que el rendimiento en cuanto a los tiempos de respuesta con referencia a otros procesadores es mucho menor, no existe retrasos en la comunicación, el sistema operativo es más estable y no existen inconvenientes.

Resultado

Se decide definir como sistema operativo: Windows 2008 Server y como procesador para el servidor un Intel Core™ i7-920 con su respectivo hardware compatible.

Primera Prueba de usabilidad

Se realizo la prueba con 10 alumnos de la materia de control automático, se les asigno su respectivo usuario para que todos realicen sus experimentos simultáneamente.

Análisis de la primera prueba de usabilidad

Luego de hacer un análisis de la primera prueba se detectaron 2 errores críticos en el sistema el cual no permitía la ejecución simultánea de varios usuarios, y no permitía graficar los resultados simultáneamente, se recibieron críticas constructivas en cuanto al uso del sistema y su interfaz.

Resultado

La prueba fue un rotundo fracaso ya que el servidor no respondió como debía, e inclusive quedó fuera de servicio debido a errores en varias

ocasiones, se da por terminada la prueba sin mayores resultados, luego de lo cual se realizaron las respectivas correcciones en el módulo de procesamientos de datos y se pone al sistema en funcionamiento.

Segunda prueba de usabilidad

Se realizó la prueba con 10 alumnos de la materia de control automático, diferentes de la primera prueba, se les asigno su respectivo usuario para que todos realicen sus experimentos simultáneamente.

Análisis de la segunda prueba de usabilidad

Luego de hacer un análisis de la primera prueba no se detectaron errores que afecten el funcionamiento del sistema, el servidor responde como se esperaba, no existe saturación en los procesos, todas las validaciones en la maquinaria funcionan como es debido, se reciben sugerencias en cuanto a la interfaz del usuario y al manejo de ciertos controles; se da por terminada la prueba.

Resultado

Se da como resultado una prueba exitosa y se toman en cuenta las sugerencias de los estudiantes en cuanto al diseño de la página web y a ciertos controles, se modifica la interfaz y posteriormente se pone en ejecución el sistema actual.

Conclusiones y Recomendaciones

Conclusiones

1. Se cumplió con los objetivos planteados al inicio del proyecto, logrando realizar un sistema web que permite controlar las plantas del laboratorio de control automático en forma remota.

2. Se culmina el proyecto de tal manera que futuros desarrolladores o administradores, puedan retomar y continuar con el desarrollo de la idea fundamental de este trabajo.

3. Se provee toda la documentación e información relacionada con el desarrollo e implementación del proyecto de una manera clara y fácil de entender para cualquier futuro desarrollador que retome el desarrollo del sistema.

4. La solución implementada, puede aplicarse a gran escala, no solo al laboratorio de Control Automático, sino también puede ampliarse hacia otros laboratorios de la facultad.

5. Es posible incorporar otros centros de educativos a la utilización de esta herramienta, logrando así crear una red de cooperación entre universidades, fortaleciendo la calidad de la educación mutuamente por medio del intercambio de recursos de laboratorio. En la actualidad se ha iniciado esta actividad con el Instituto de Automatización y Control (IRT) de la Universidad de Aachen en Alemania.

6. El sistema tiene gran adaptabilidad a diversos tipos de maquinarias y software de control, ya sean los utilizados en el desarrollo del proyecto (cFP – MAX de National Instrument) u otros equipos que usen tecnología OPC.

Recomendaciones

1. Es conveniente realizar una revisión del software de administración de los recursos puesto a disposición para la operación en forma remota del laboratorio, ya que el realizado consta de herramientas básicas para el correcto funcionamiento del sistema.

2. Incorporar nuevas herramientas a medida que se realicen futuras implementaciones de plantas o experimentos, ya que el sistema está diseñado para adaptarse a nuevas funcionalidades según se requiera.

3. Es necesario el apoyo institucional por intermedio de la facultad para incorporar otras áreas y laboratorios en la adopción de esta herramienta.

4. Extender la cooperación entre universidades e integrar una red laboratorios “LabCon” tanto nacional como internacional.

Anexos

Preparación del Servidor LabCon

El primer paso para la preparación del servidor LabCon es instalar todas las herramientas necesarias para su funcionamiento dentro de las cuales tenemos las siguientes

1. Matlab: de preferencia versiones anteriores al 2009
2. Sybase (base de datos)
3. MAX Gestor de la maquinaria el cual posee el OPC Server
4. MSP
5. jdk de java

Observaciones

Debido a limitaciones del sistema MSP, el sistema operativo debe ser de 32 bits; para el servidor LabCon se seleccionó Windows Server 2008 el cual recomendaremos para futuras implementaciones, para mayor referencia visitar el sitio oficial **http://sourceforge.net/projects/msp**

Instalaciones

Todos y cada uno de los programas listados anteriormente tienen sus asistentes de instalación, lo cual no requiere ningún conocimiento especializado para llevar a cabo esa tarea, tan solo el usuario debe seguir las instrucciones.

Una vez realizada la tarea de instalar todas las herramientas se debe localizar la ruta C:\MATLABServerPages\webapps\ROOT y borrar todo su contenido, y en su lugar copiar el contenido de la carpeta ROOT provista por los desarrolladores la cual contiene el sistema LabCon.

Configuraciones

Directorios

Para el correcto funcionamiento del sistema LabCon, debemos configurar correctamente el sistema de archivos, logrando con esto que el sistema re direccione automáticamente todos y cada una de las imágenes y documentos usados, para ello debemos configurar 3 archivos

Edit Matlab Server Pages Properties

Register Matlab as Com Server

Start up the server

Edit Matlab Server Pages Properties

Figura 72: Ícono de acceso directo a edición de propiedades

Este archivo permite configurar las carpetas en las cuales se encontraran los documentos y las imágenes; se puede configurar de dos maneras, ingresando mediante el acceso directo que se crea en el escritorio o ingresando mediante el menú inicio

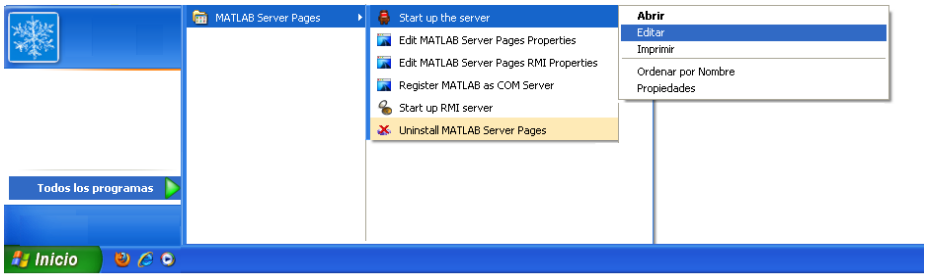


Figura 73: Ingreso a edición de propiedades por medio del menú inicio

Una vez abierto el archivo encontraremos las siguientes líneas de texto:

Figura 74: Configuración por defecto del MSP

Reemplazarlas por:

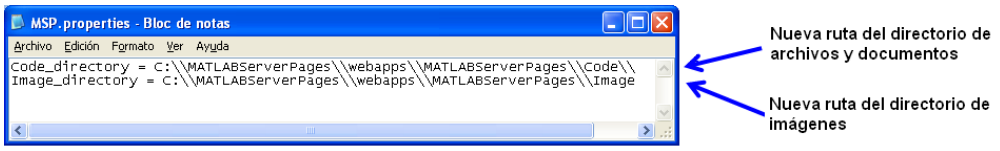


Figura 75: Nueva configuración de MSP

Las rutas mencionadas en la nueva configuración dependen del criterio de los desarrolladores para una mejor administración; Guardar los cambios y cerrar

Register Matlab as Com Server

Figura 76: Ícono para registrar Matlab como servidor

Este archivo permite configurar con que versión de Matlab se trabajará; en él se debe configurar la ruta en la cual haya sido instalado Matlab en el servidor. Se puede configurar de dos maneras, ingresando mediante el acceso directo que se crea en el escritorio o ingresando mediante el menú inicio

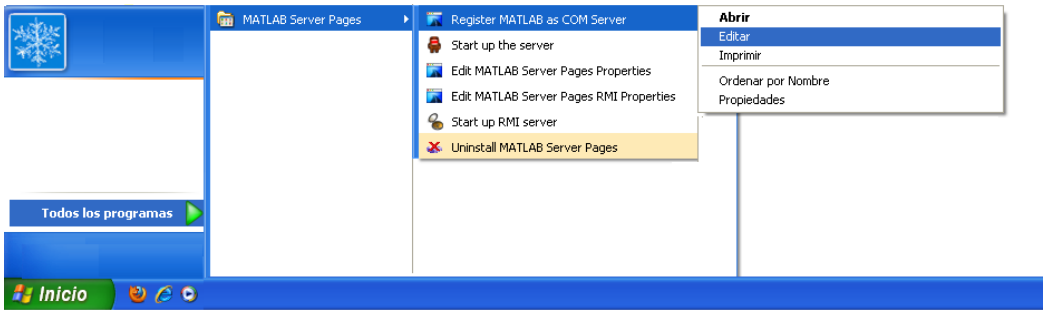


Figura 77: Ingreso a registrar Matlab como servidor desde el menú inicio

Una vez abierto el archivo encontraremos las siguientes líneas de texto:

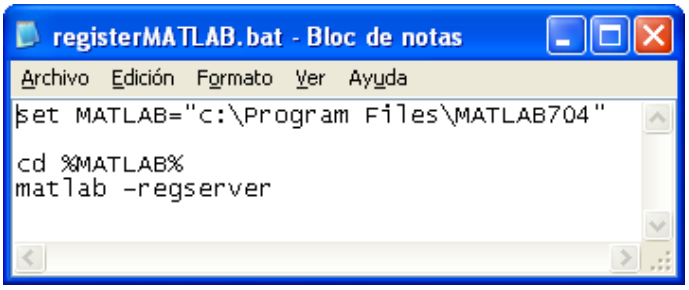


Figura 78: Configuración por defecto del archivo registerMATLAB

Editar la primera línea con la ruta en la cual haya sido instalado Matlab en el servidor:

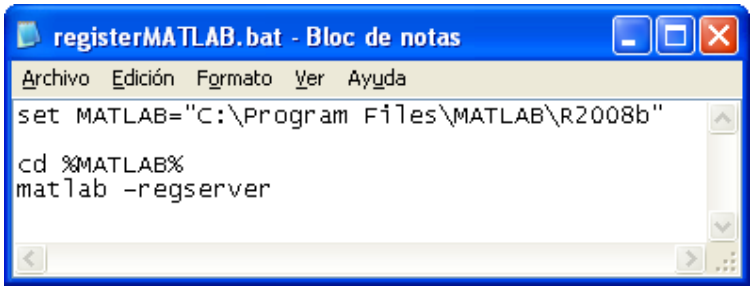


Figura 79: Nueva configuración del archivo registerMATLAB

Guardar la configuración, cerrar el documento y ejecutarlo para que el archivo se encargue de registrar el nuevo servidor

Start up the server

Figura 80: Ícono para ejecutar el servidor

Este archivo permite configurar el servidor; en él se debe configurar la ruta en la cual haya sido el jdk de java en el servidor. Se puede configurar de dos maneras, ingresando mediante el acceso directo que se crea en el escritorio o ingresando mediante el menú inicio.

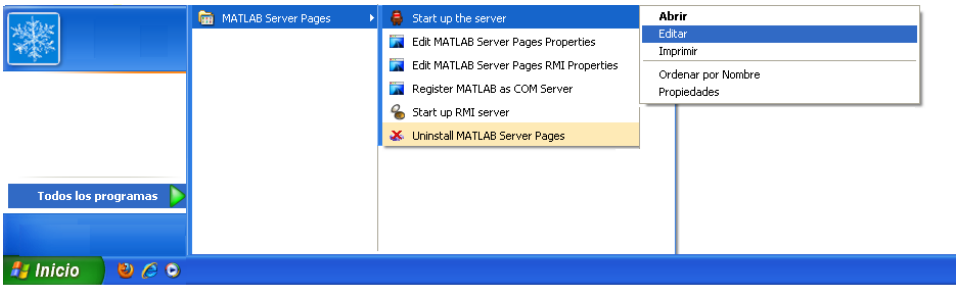


Figura 81: Ejecutar el servidor desde el menú inicio

Una vez abierto el archivo encontraremos las siguientes líneas de texto:

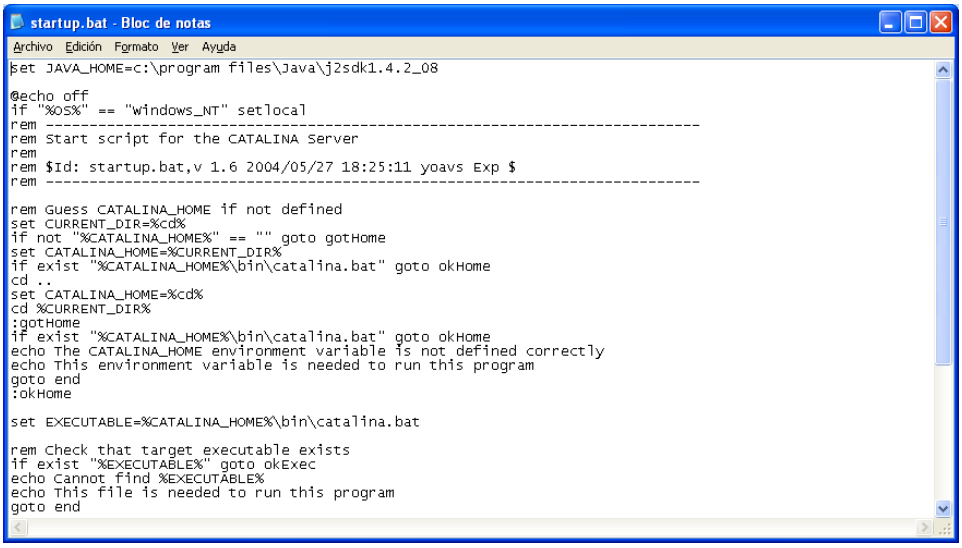


Figura 82: Configuración original del archivo Startup

Se edita la primera línea con la ruta de donde está instalado el jdk en el servidor, es decir:

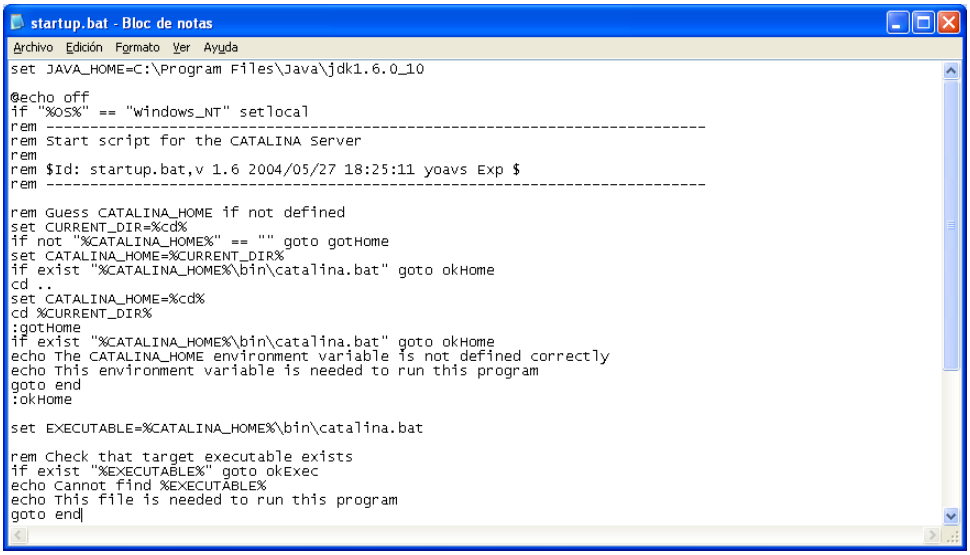


Figura 83: Nueva configuración del archivo Startup

Guardar la configuración, cerrar el documento y ejecutarlo para que el archivo se encargue de ejecutar por primera vez el servidor

Prueba de verificación

Para verificar que todo esté en perfecto funcionamiento debemos ejecutar el servidor y probar con la página de bienvenida del sistema, para ello debemos ejecutar nuestro navegador preferido y copiar en la barra el siguiente link

http://localhost:8080/MATLABServerPages/

De salir alguna pantalla con un mensaje de error se deberá cerras todas las aplicaciones abiertas y reiniciar el servidor; si luego de esto el servidor sigue aún sin funcionar se deberá desinstalar todas las aplicaciones y reinstalar todo desde el principio

Base de Datos

Si es la primera vez que se va a ejecutar el sistema LabCon se deberá borrar el archivo labcon.log que se encuentra en el directorio

"c:\MATLABServerPages\webappps\ROOT\baseDatos", para que no interfiera con la base de datos

Copiar el archivo jconn2.jar que está en la ruta: c:MATLABServerPages\webapps\root\WEB-INF\lib y ubicarlo en la carpeta jar de Matlab la cual en nuestro servidor se encuentra en la ruta: c:\Archivo de programa\MATLAB\R2008b\java\jar

Ubicar y editar el siguiente archivo: c:Archivo de programa/MATLAB/R2008b/ toolbox/ local/ classpath.txt, añadiéndole al final la siguiente línea $matlabroot/java/jar/jconn2.jar

Conexión de la base de datos

Modo manual

Ejecutar la base de datos mediante el acceso directo en el escritorio o mediante el menú inicio

Figura 84 Ícono para ejecutar base de datos

Figura 85: Ingreso a base de datos desde el menú inicio

1. Clic en el icono de conect o presionar f11

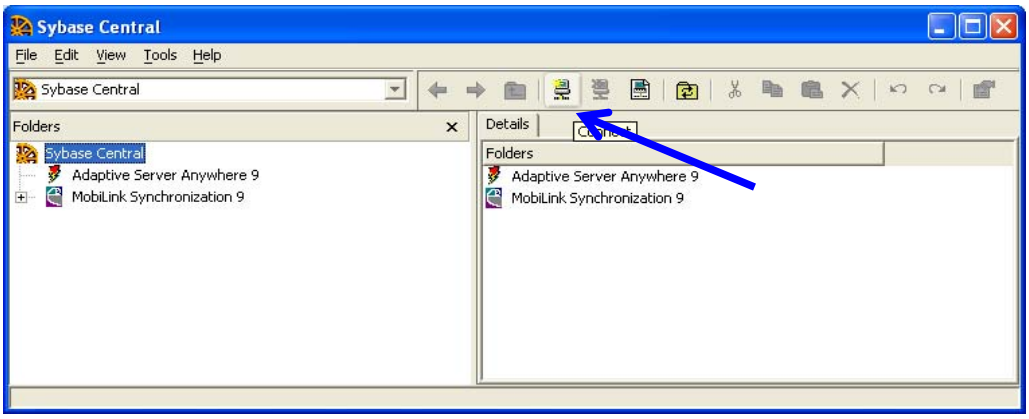


Figura 86: Conexión manual a la base de datos: Paso 1

2. Seleccionar “Adaptive Server Anywhere 9” y dar clic en OK

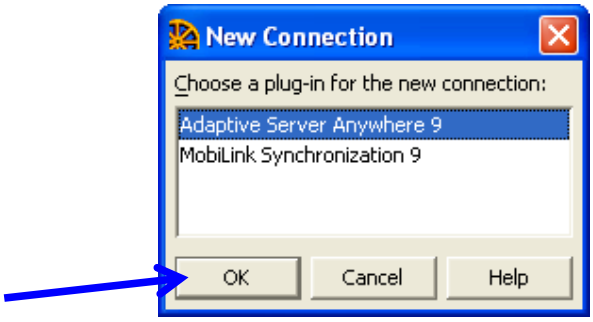


Figura 87: Conexión manual a la base de datos: Paso 2

3. Llenar los datos de usuario y clave con los datos por defecto de la base de datos, los cuales son para el usuario: dba y la clave: sql

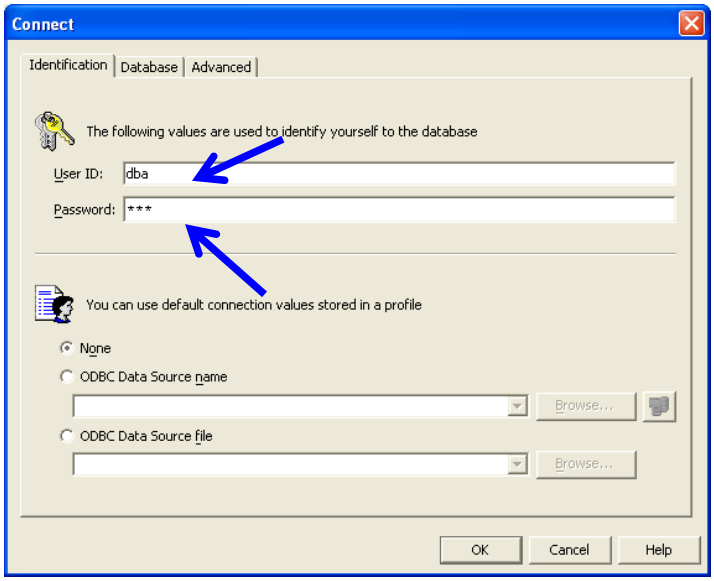


Figura 88: Conexión manual a la base de datos: Paso 3

4. Seleccionar la pestaña de Database y dar clic en browse

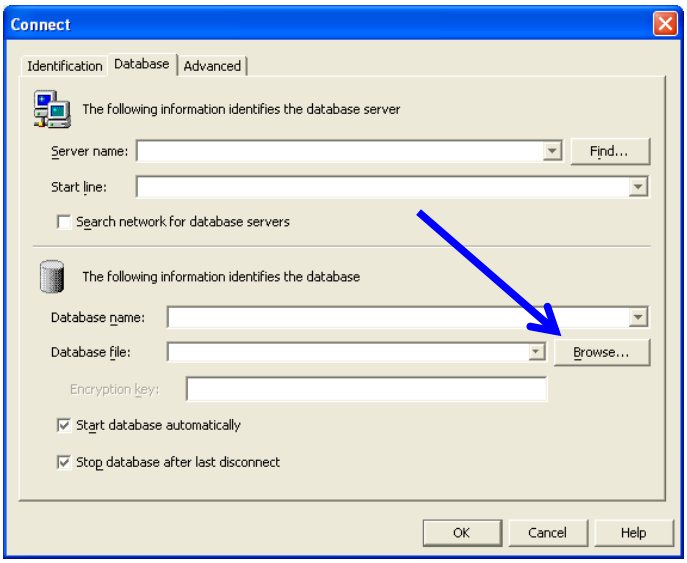


Figura 89: Conexión manual a la base de datos: Paso 4

5. Seleccionar la base de datos en la cual se trabajará, buscarla en la ruta del sistema LabCon y dar click en abrir

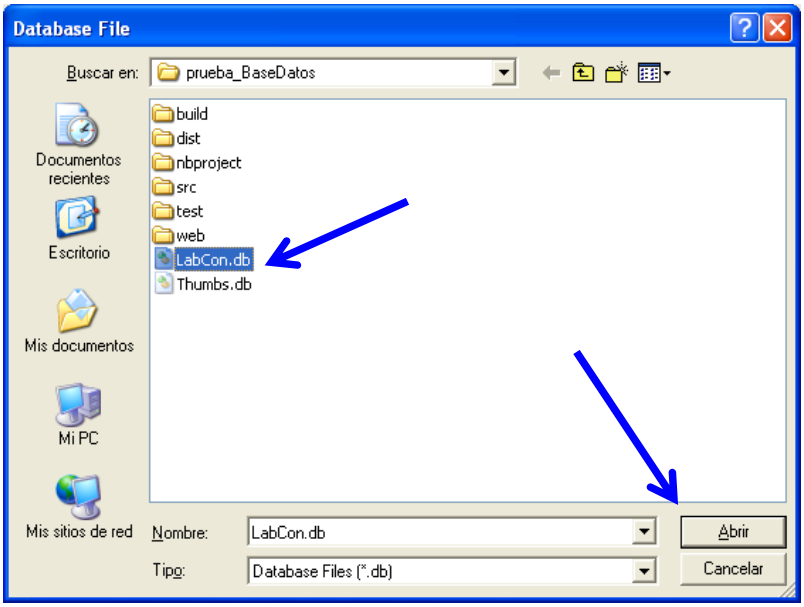


Figura 90: Conexión manual a la base de datos: Paso 5

6. Posteriormente en el campo de Database file aparecerá la ruta de la base de datos; confirmar que la ruta sea la correcta y dar click en ok.

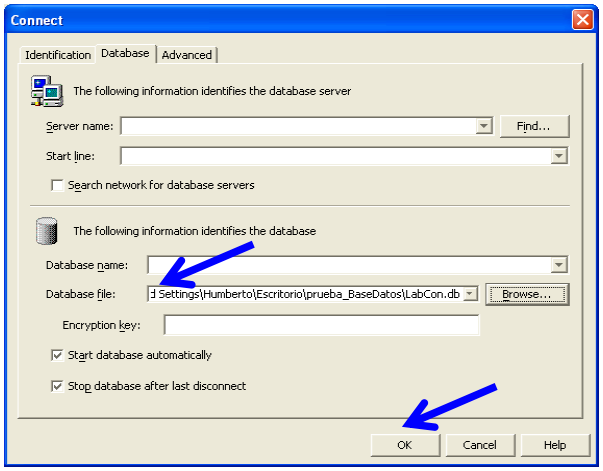


Figura 91: Conexión manual a la base de datos: Paso 6

Luego de estos pasos se abrirá la base de datos y el sistema estará listo para usarse

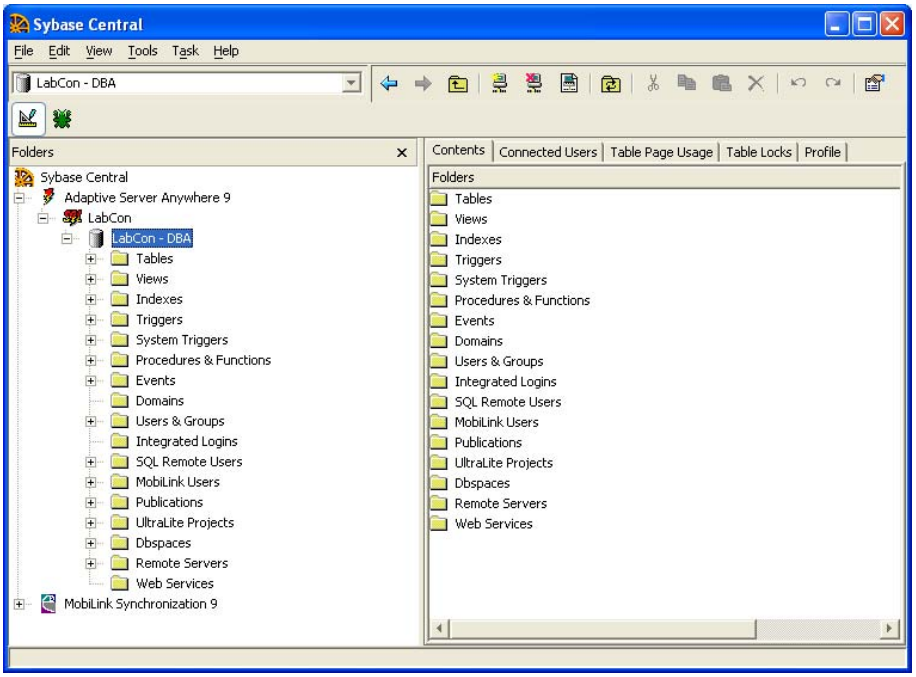


Figura 92: Ventana de administración de la base de datos

Modo Automático

El modo automático sirve para que la base de datos se ejecute automáticamente al abrir Sybase, este método es el más recomendado ya que de no existir algún administrador cerca el servidor, este puede ejecutar automáticamente todas las aplicaciones de manera automática.

1. Clic en el “Tools - > Conection Profiles” o presionar f9

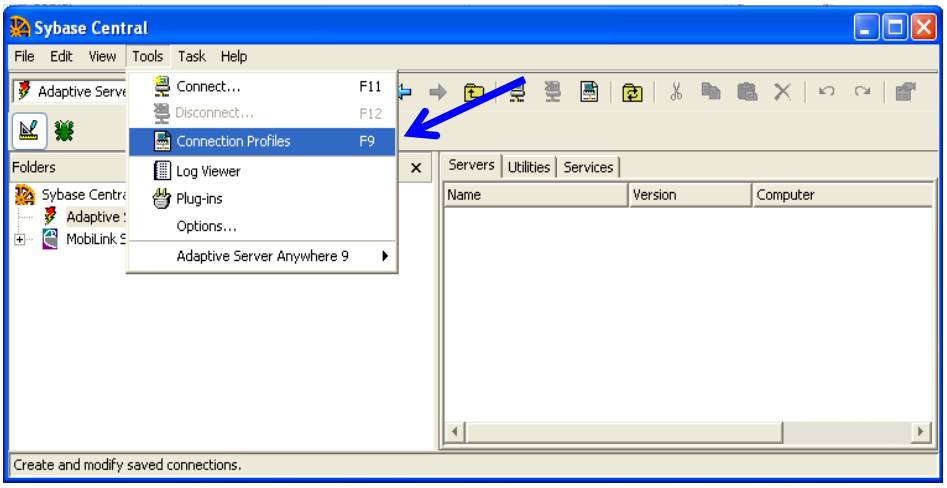


Figura 93: Conexión automática a la base de datos: Paso 1

2. Seleccionar la opción de New para crear un nuevo perfil de conexión

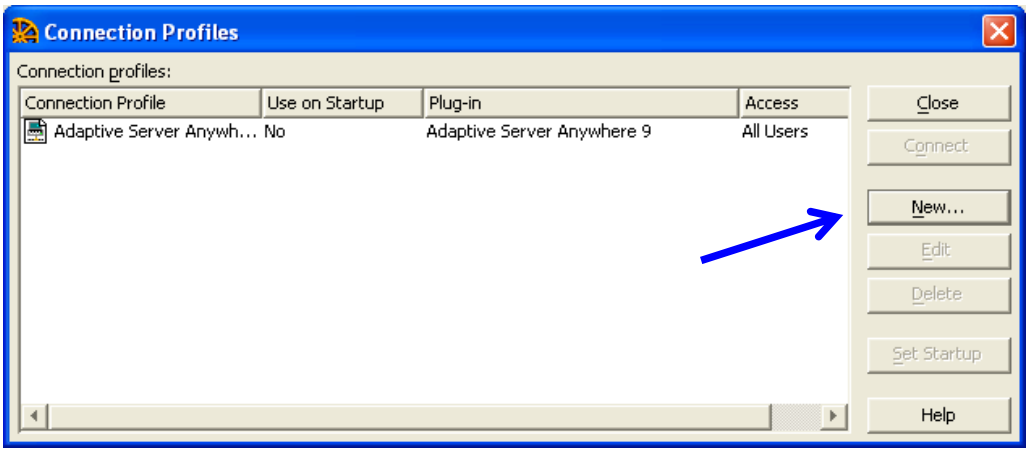


Figura 94: Conexión automática a la base de datos: Paso 2

3. Seleccionar un nombre para el perfil, en nuestro caso se seleccionó LabConDB

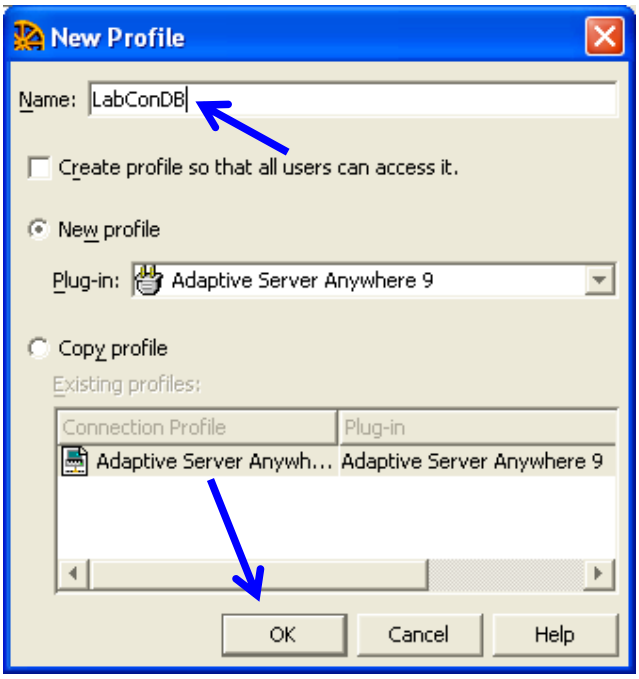


Figura 95: Conexión automática a la base de datos: Paso 3

4. A partir de este punto se debe repetir los pasos de la configuración manual desde el punto 1 hasta el punto 6.

5. Luego de seguir los pasos del 1 al 6 de la configuración manual seleccionar el perfil que se acaba de configurar y dar clic en set Startup.

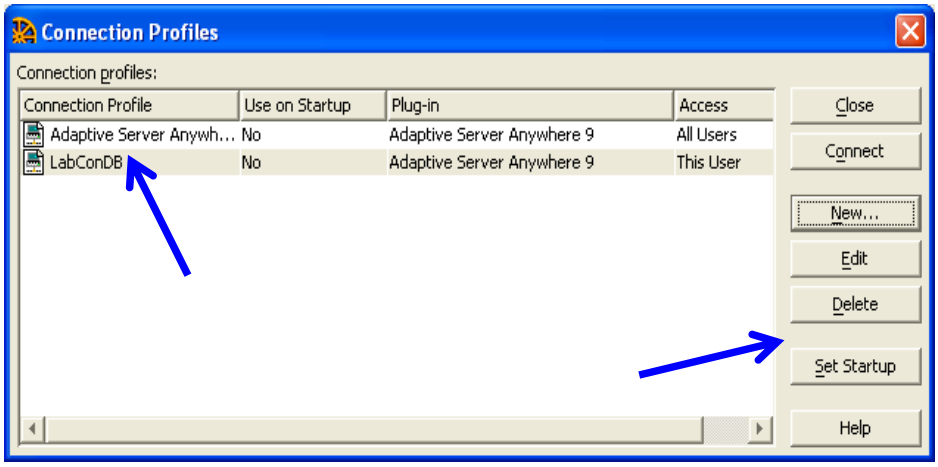


Figura 96: Conexión automática a la base de datos: Paso 5

6. Revisar que el campo Use on Startup del perfil creado se haya cambiado a Yes

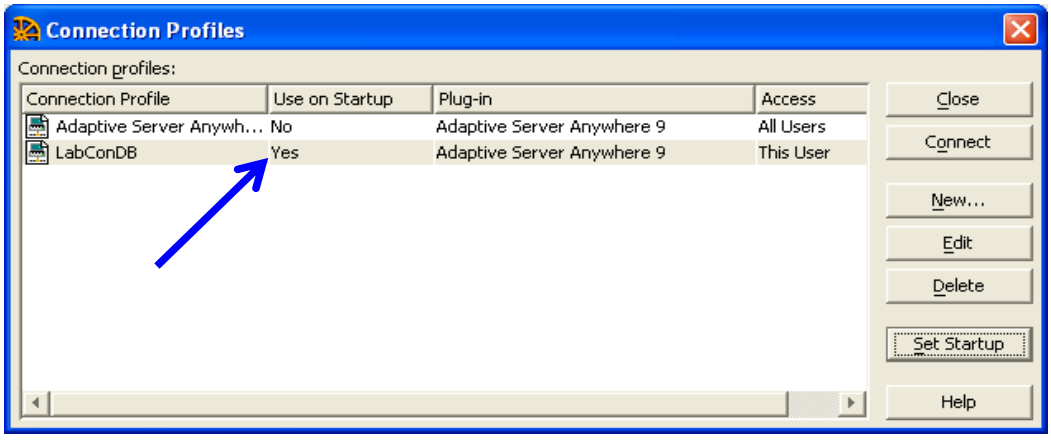


Figura 97: Conexión automática a la base de datos: Paso 6

7. Luego de estos pasos la base de datos quedara configurada para que se conecte automáticamente al abrir Sybase

Administración de datos del archivo XML de configuración (datos de grupos, bloques y parámetros del Sistema)

Estructura de datos del archivo XML

- El archivo XML está formado con la siguiente estructura:

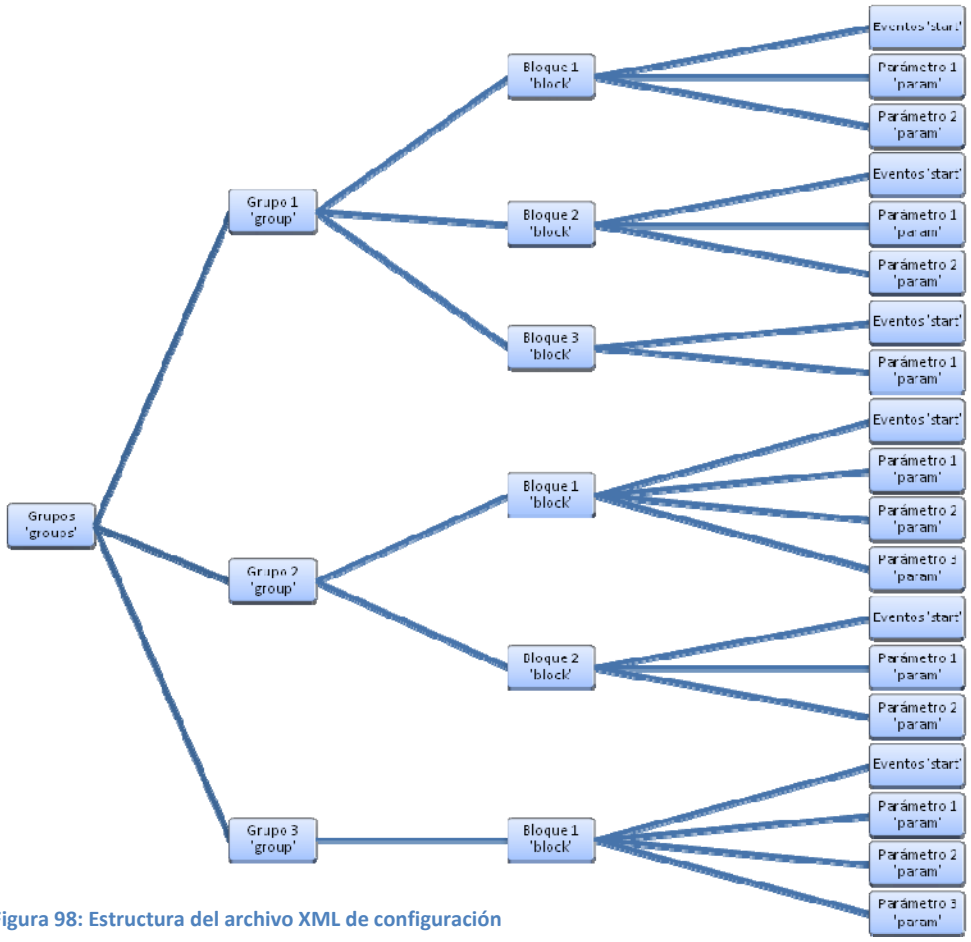


Figura 98: Estructura del archivo XML de configuración

Los datos están conformados por uno o varios grupos, los cuales contienen uno o varios bloques, que a su vez están formados por parámetros y acciones especiales que se realizan según el comportamiento del bloque.

Grupos en el archivo XML

<group name="NOMBRE_DEL_GRUPO">

BLOQUES

</group>

- name : Nombre del grupo

Ejemplos de uso:

<group name="Continuous">

<block name="Integrator" .....>

.....

</block>

<block name="State-Space" .....>

.....

</block>

<block name="Transfer Fcn" .....>

.....

</block>

<block name="Zero-Pole" .....>

.....

</block>

</group>

```
<group name="Practices Lab">
        <block  name="Tank" .....>
                .....
        </block>
        <block name="Speed Control" .....>
                .....
        </block>
</group>
```

Bloques de un grupo en el archivo XML

```
<block name="NOMBRE_DEL_BLOQUE"
npIN="NÚMERO_PUERTOS_ENTRADA"
npOUT="NÚMERO_PUERTOS_SALIDA"
sWIN="NÚMERO_FACTOR_ALTO_VENTANA"
desc="DESCRIPCIÓN_BLOQUE" enable="BLOQUE_HABILITADO">
        PARÁMETROS
</block>
```

- **name:** El nombre del bloque (que tiene que ser el mismo nombre del bloque agregado en el modelo de la librería)
- **npIN:** El número de puertos de entrada del bloque

- **npOUT:** El número de puertos de salida del bloque
- **sWIN:** Variable que sirve para definir el tamaño de la ventana de parámetros de un bloque proporcionalmente.
    - Ejemplos de uso: 0, 0.1, 0.3, 0.5, 1, 1.5

- **desc:** Descripción del bloque
    - En este campo se pueden utilizar códigos especiales para mejorar la presentación de la descripción del bloque en el sistema Flash
        - **[-BR-]:** Genera un salto de línea
        - **[-TAB1-]:** Genera una distancia de tabulación pequeña
        - **[-TAB2-]:** Genera una distancia de tabulación mayor

Ejemplo de uso: en el XML el código es:

desc="[-BR-]State-space model:[-TAB2-] dx/dt = Ax + Bu [-TAB2-] y = Cx + Du" y en el sistema al ver la información del bloque, se observa así:

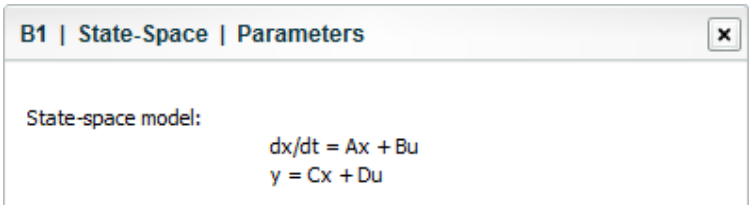


Figura 99: Información de un bloque del panel de dibujo

- **enable:** Indica si el bloque está activado o no al iniciar el sistema

Ejemplo de uso:

<block name="Integrator" npIN="1" npOUT="1" sWIN="1.8" desc="[-BR-]Continuous-time integration of the input Signal." enable="1">

<param name="TextInput" ..... />

<param name="TextInput" ..... />

</block>

<block name="State-Space" npIN="1" npOUT="1" sWIN="1.0" desc="[-BR-]State-space model:[-TAB2-] dx/dt = Ax + Bu [-TAB2-] y = Cx + Du" enable="1">

<param name="TextInput" ..... />

<param name="TextInput" ..... />

</block>

Parámetros de un bloque en el archivo XML

<paramname="**NOMBRE_DEL_BLOQUE_PADRE**"label="**ETIQUETA_VARIABLE**" varname="**NOMBRE_VARIABLE**"def="**VALOR_POR_DEFECTO**" validation="**VALIDACIÓN_EVENTO**"image="**NOMBRE_VARIABLE_IMAGEN**" ports="**NOMBRE_VARIABLE_PUERTO**"/>

- **name:** Nombre del tipo de parámetro a utilizar
    - Se tienen disponible los siguientes tipos de parámetros:
        - **ComboBox:** Listas de opciones seleccionables

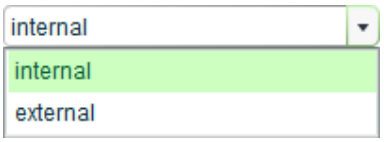


Figura 100: Parámetros de bloques: ComboBox

        - **TextInput:** Campos de entradas normales

Figura 101: Parámetros de bloques: TextInput

        - **CheckBox:** Cuadro de verificación (marcar)

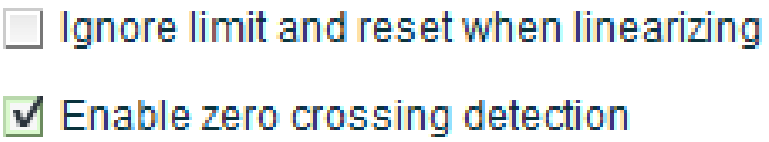


Figura 102: Parámetros de bloques: CheckBox

        - **FaderGain:** Control de ganancia (caso especial)

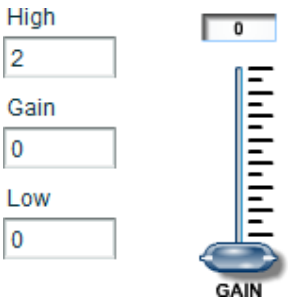


Figura 103: Parámetros de bloques: FaderGain

- **label:** Etiqueta que representa el nombre de la variable para su ingreso en la ventana de parámetros

- **varname:** Nombre interno que asigna MATLAB a la variable para ser reconocida en el modelo

- **def:** Valor por defecto con el que inicia el bloque, debe ser el mismo que se tenga configurado en libreria.mdl
    - Cuando se utiliza el tipo de parámetro ComboBox, es necesario indicar al sistema de dos o más elementos formarán parte de la lista.
    - Por ejemplo, si la lista de nombres de variables para el ComboBox es: internal, external. La forma correcta para indicar al sistema en el XML es: def="internal,|_|,external" y en el sistema al ver el parámetro del bloque, se observa así:

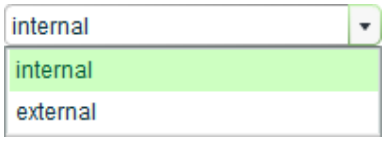


Figura 104: Configuración de opciones internas de un bloque

- **validation:** Indica acciones especiales a realizar según el tipo de parámetro utilizado

- Este campo solo es utilizado por los tipos de parámetros: ComboBox y CheckBox
- Las validaciones habilitadas son:
  - **visible:** Sirve para modificar si una o varias variable que se indiquen, son visibles o no
  - **editable:** Sirve para modificar una o varias variables que se indiquen, son editables o no
- Ejemplos de uso:
  - Un ComboBox tiene disponible las variables: internal y external
    - La validación: visible,|_|,internal,|_|,InitialCondition indica que si la variable tiene el valor 'internal', la variable de nombre 'InitialCondition' es visible, caso contrario, permanecerá oculta
  - Un CheckBox inicialmente no está marcado
    - La validación: editable,|_|,on,|_|,UpperSaturationLimit,|_|,LowerSaturationLimit indica que si la variable está marcada 'on', las variables 'UpperSaturationLimit' y 'LowerSaturationLimit' serán editables, caso contrario, permanecerán como no editables.

- **image:** Permite modificar la imagen del bloque, si las variables indicadas es seleccionada
    - Este campo solo es utilizado por los tipos de parámetros: ComboBox y CheckBox
    - Si la variable es activada o seleccionada, cambia la imagen del bloque por la asignada al nombre de la variable en cuestión
    - Ejemplos de uso:
        - Un ComboBox tiene disponible las variables: internal y external
            - La validación: external indica que si la variable seleccionada es 'external', la imagen del bloque cambia por la asignada a esta variable
        - Un ComboBox tiene disponible las variables: none ,rising, falling, either, level, levelhold
            - La validación: rising,|_|,falling,|_|,either,|_|,level,|_|,levelhold indica que si la variable seleccionada es 'rising' o 'falling' o 'either' o 'level' o 'levelhold', la imagen del bloque cambia por la asignada a esta variable
            - Cuando la imagen de un bloque es modificada al aplicar esta validación, en la ruta de imágenes del sistema, dentro del grupo donde se encuentra el

bloque, se debe crear una carpeta con el mismo nombre del bloque. Por ejemplo del bloque 'Integrator' del grupo 'Continuous'

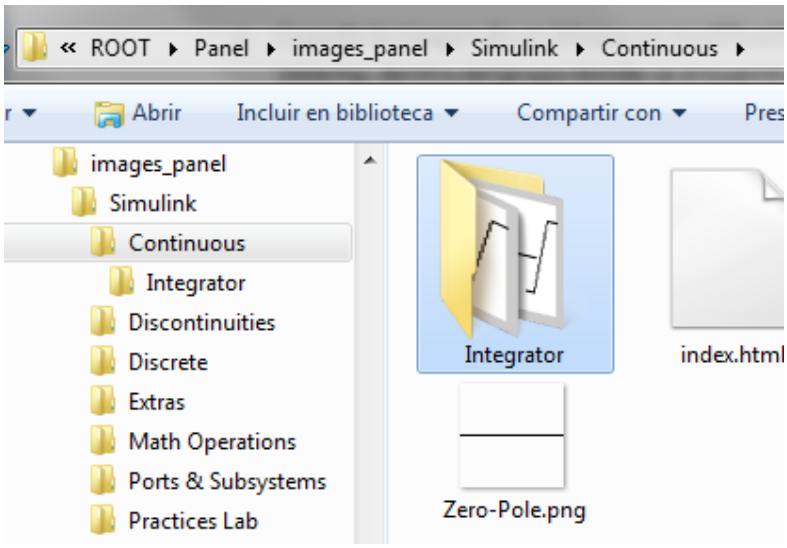


Figura 105: Directo de imágenes

o Dentro de la carpeta creada, se ubican las imágenes para el bloque que se utilizarán acorde los valores que se configuren en los parámetros que estén dentro de las validaciones de imágenes. Por ejemplo del bloque 'Integrator' del grupo 'Continuous'

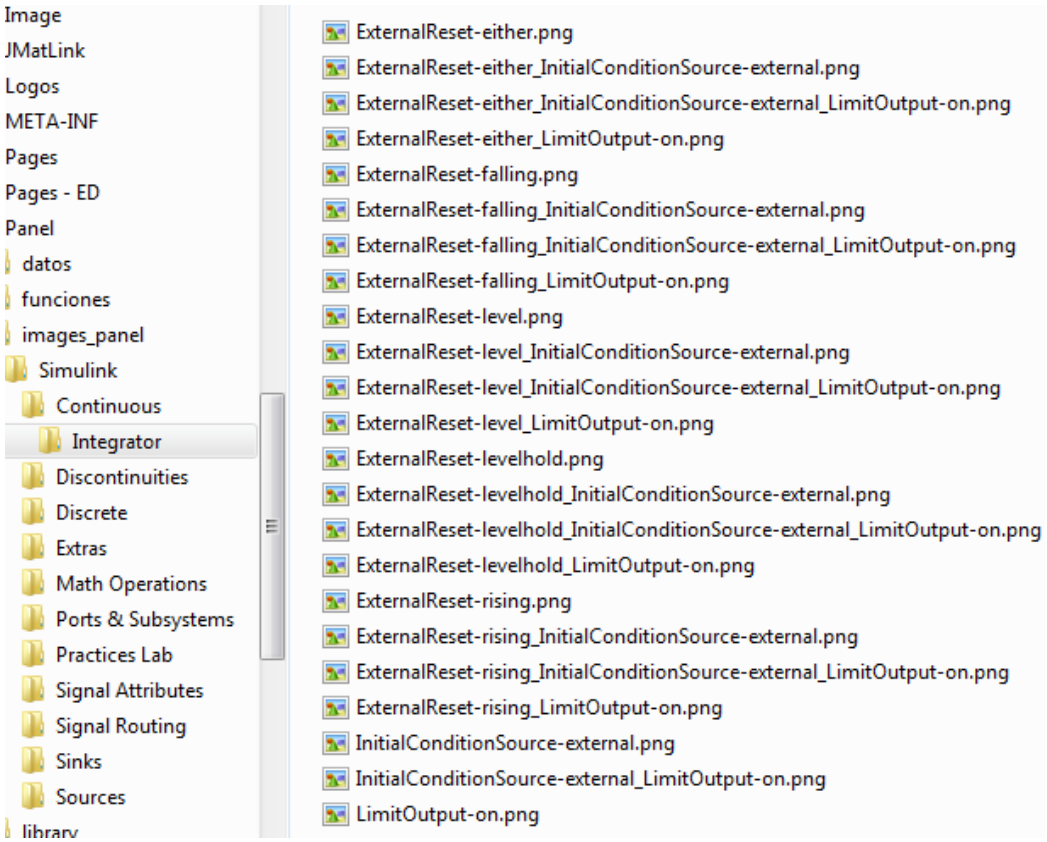


Figura 106: Carpeta de imágenes del bloque Integrator

- **ExternalReset-either:** Se utiliza esta imagen cuando en la variable 'ExternalReset' el valor seleccionado es 'either'

- **ExternalReset-falling:** Se utiliza esta imagen cuando en la variable 'ExternalReset' el valor seleccionado es 'falling'

- **InitialConditionSource-external:** Se utiliza esta imagen cuando en la variable 'InitialConditionSource-external' el valor seleccionado es 'external'

- **ExternalReset-falling_InitialConditionSource-external:** Se utiliza esta imagen cuando en la variable 'ExternalReset' el valor seleccionado es 'falling' y a la vez, en la variable 'InitialConditionSource-external' el valor seleccionado es 'external'

- **ExternalReset-levelhold_InitialConditionSource-external_LimitOutput-on:** Se utiliza esta imagen cuando en la variable 'ExternalReset' el valor seleccionado es 'levelhold' y a la vez, en la variable 'InitialConditionSource-external' el valor seleccionado es 'external' y al mismo tiempo en la variable 'LimitOutput' el valor seleccionado es 'on'

Figura 107: Imágenes del bloque Integrator

- **ports:** Permite agregar puertos de entrada o de salida, si la variable indicada es seleccionada
    - Este campo solo es utilizado por los tipos de parámetros: ComboBox y CheckBox
    - Las validaciones habilitadas son:
        - **in:** Sirve para aumentar un puerto de entrada en el bloque
        - **out:** Sirve para aumentar un puerto de salida en el bloque.

- Si la variable es activada o seleccionada, cambia la imagen del bloque por la asignada al nombre de la variable en cuestión
- Ejemplos de uso:
  - Un ComboBox tiene disponible las variables: internal y external
    - La validación: in,|_|,external indica que si la variable seleccionada es 'external', el bloque aumenta un puerto de entrada al bloque
  - Un CheckBox inicialmente no está marcado
    - La validación: out,|_|,on indica que si la variable está marcada 'on', el bloque aumenta un puerto de salida al bloque

Ejemplos de uso:

<param name="TextInput" label="A:" varname="A" def="1"/>

<param name="TextInput" label="Poles:" varname="Poles" def="[0 -1]"/>

<param name="ComboBox" label="Units:" varname="Units" def="Hertz,|_|,rad/sec" validation="" image="" ports=""/>

<param name="CheckBox" label="Enable zero crossing detection" varname="ZeroCross" def="on" validation="" image="" ports=""/>

Eventos especiales que se aplican en el bloque en el XML

<start name="**EVENTO_ESPECIAL**" values="**VALORES_EVENTO**"/>

- **name:** Nombre del tipo de evento especial a realizar
    - Se tienen disponible los siguientes tipos de eventos:
        - **stNameBlock:** Al agregar este evento como parte de un bloque, habilita la opción de que se le pueda asignar un nombre personalizado al bloque en la creación del modelo
            - Se utiliza sin asignar nada en 'values', ejemplo: <start name="stNameBlock" values=""/>
        - **stAccor:** Se utiliza para asignar valores iniciales en distintas posiciones de la imagen de un bloque para ser mostrados en la librería de bloques del sistema
            - Para representara textos dentro de un bloque, se dispone de las siguientes ubicaciones:
                - v, vA, vA2, vB, vN

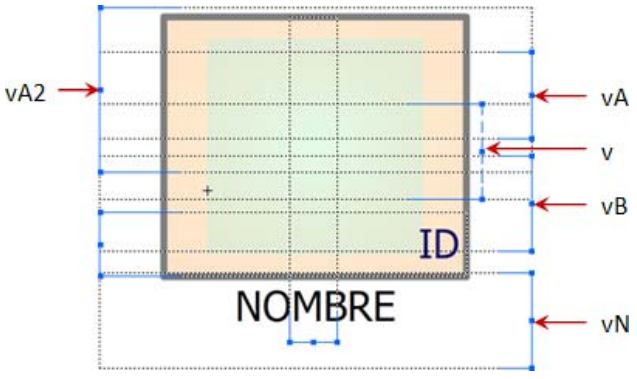


Figura 108: Rotulación de un bloque

- Por ejemplo, del bloque 'Transfer Fcn' del grupo 'Continuous'

  <startname="stAccor"values="vA,|_|,1,|_|,vB,|_|,s+1"/>

  que indica que a la ubicación 'vA' asigna valor '1' y a la ubicación 'vB' asigna el valor 's+1'. Se obtiene en el sistema:

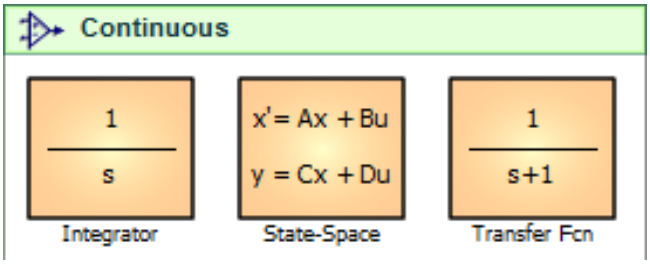


Figura 109: Vista de bloques en el sistema

- **stPanel:** Se utiliza para reflejar visualmente en la imagen del bloque, los valores que se configuren en las propiedades del parámetro, cuando el bloque ya ha sido agregado al modelo creado, y se modifiquen los valores predeterminados
  - Para representara textos dentro de un bloque, se dispone de las siguientes ubicaciones:
    - v, vA, vA2, vB, vN

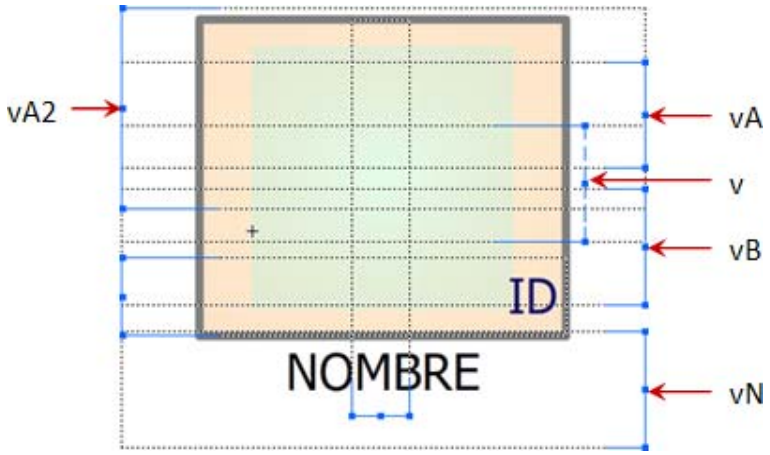


Figura 110: Rotulación de bloques

- Por ejemplo, del bloque 'Transfer Fcn' del grupo 'Continuous'

  <startname="stPanel"values="vA,|_|,Numerator,|_|,vB,|_|,Denominator"/>

  que indica que a la ubicación 'vA' asigna valor que se configure en la variable 'Numerator' y a la ubicación 'vB' el valor que se configure en la variable 'Denominator' de los parámetros del bloque. Se obtiene en el sistema:

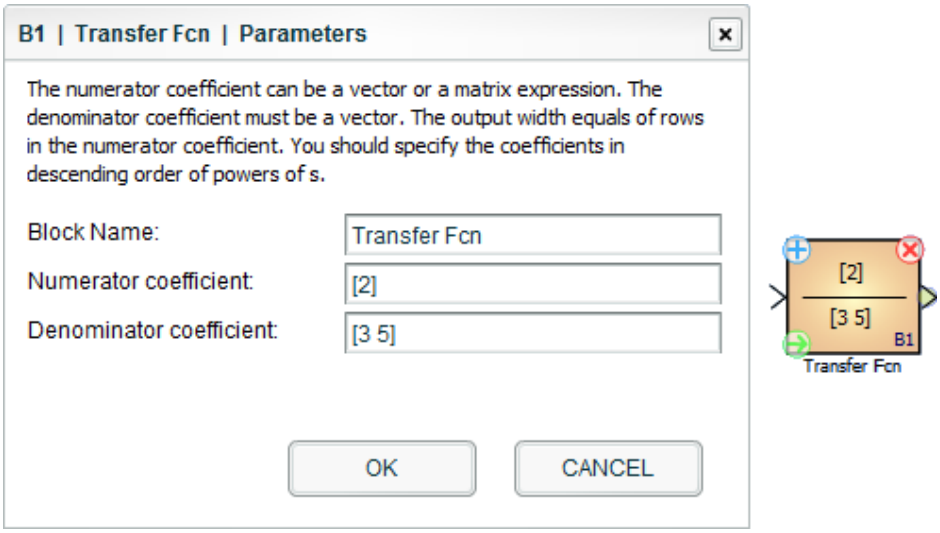


Figura 111: Vista de la configuración interna de un bloque

- **label:** Etiqueta que representa el nombre de la variable para su ingreso en la ventana de parámetros

Ejemplos de uso:

<start name="stNameBlock" values=""/>

<start name="stAccor" values="vA,|_|,1,|_|,vB,|_|,s+1"/>

<start name="stPanel" values="vA,|_|,Numerator,|_|,vB,|_|,Denominator"/>

<start name="stAccor" values="vA,|_|,(s-1),|_|,vB,|_|,s(s+1)"/>

<start name="stPanel" values="vA,|_|,Zeros,|_|,vB,|_|,Poles"/>

<start name="stAccor" values="v,|_|,1"/>

<start name="stPanel" values="v,|_|,Gain"/>

Modificación y actualización del XML

Revisión del bloque en parte del MATLAB

- Abrir el archivo de la librería ubicado en:

  ROOT -> Code -> Library -> **libreria.mdl**

- Abrir las propiedades del bloque a actualizar y verificar sus valores iniciales, los cuales deben ser los mismos que se coloquen el archivo **grupos_bloques.xml**

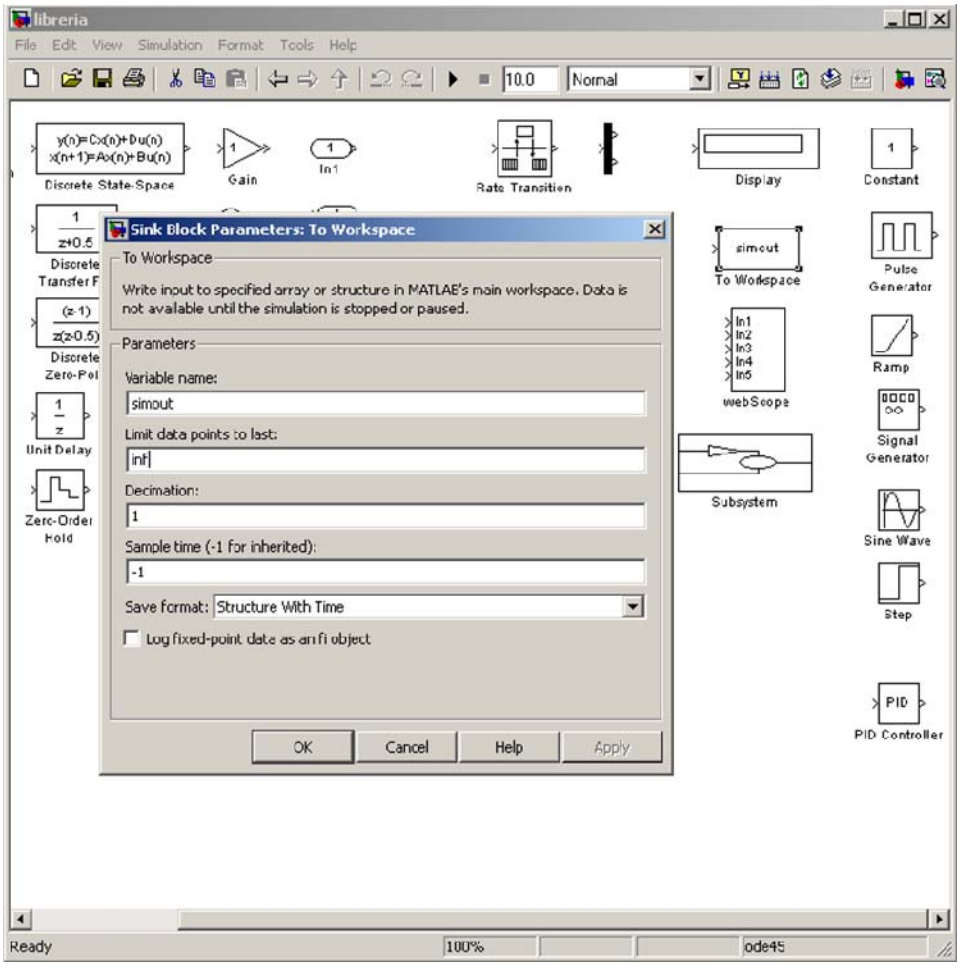


Figura 112: Revisión de un bloque de la librería de componentes de Matlab en el servidor

Revisión del bloque para conocer nombres internos de las variables a ser agregadas al documento XML

- Crear un modelo nuevo en MATLAB y agregar solo el bloque a conocer el nombre interno que maneja MATLAB para sus variables
- Cambiar los valores por defecto de las variables que se desea por valores que sean fácilmente reconocibles

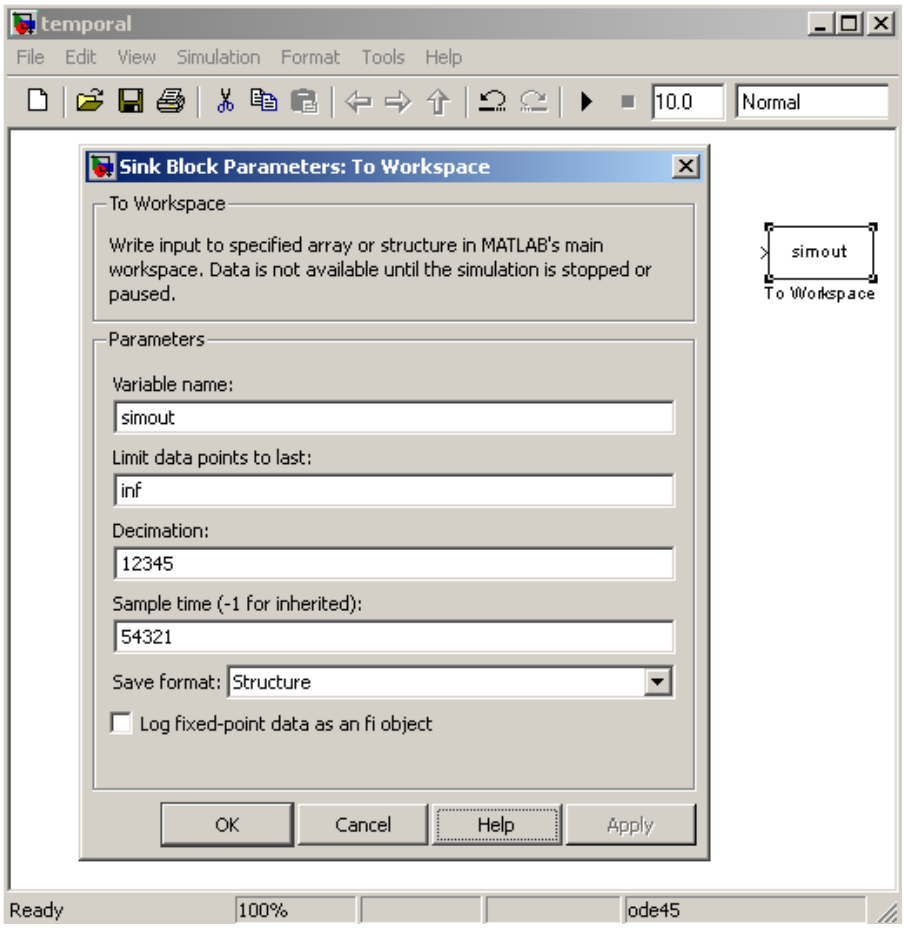


Figura 113: Ingreso de valores característicos para conocer el nombre de las variables internas

- Guardar este nuevo modelo en algún lugar de fácil acceso
- Abrir este modelo de forma manual con algún editor de texto, en este caso se va mostrar con el Bloc de notas de Windows
- Revisar el texto buscando los valores introducidos para las variables, que generalmente se encuentran al final del archivo

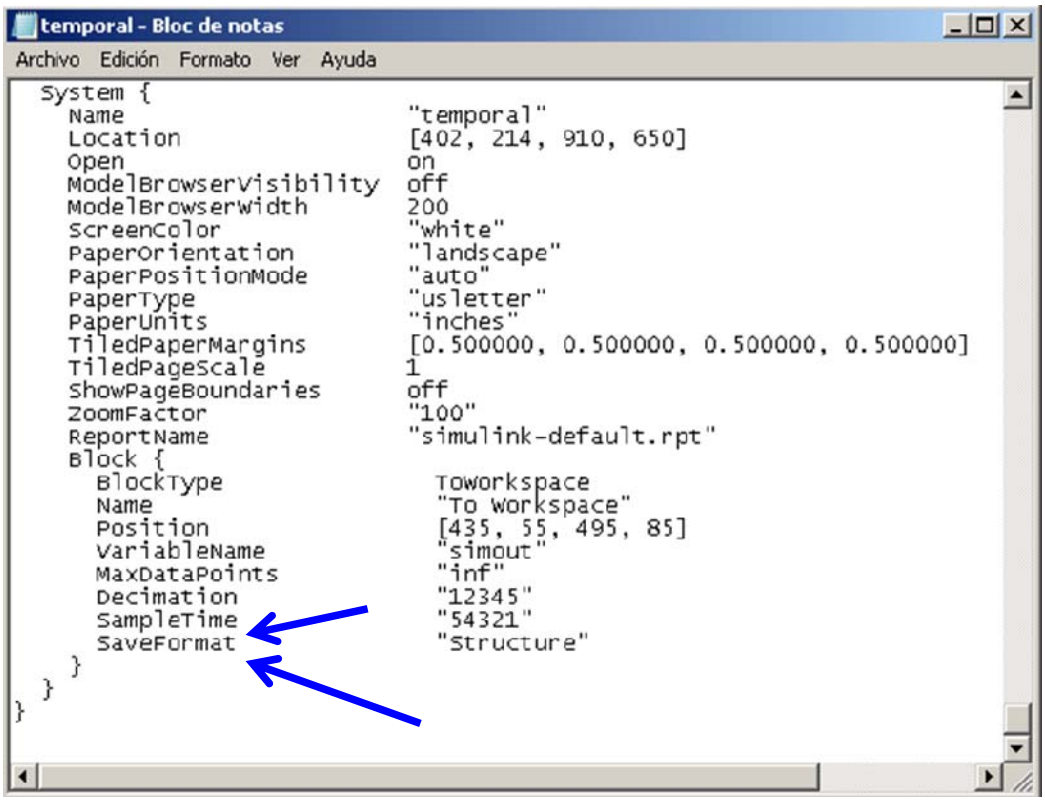


Figura 114: Vista de un archivo mdl en el block de notas de Windows

- En la columna izquierda, de la variable con los valores ingresados, está el nombre interno de la variable con el que MATLAB gestiona por comandos su gestión

Revisión del bloque en parte de modificación del XML

- Ubicar el archivo:

  ROOT -> Panel -> datos -> **grupos_bloques.xml**

- Abrirlo con algún editor de texto, de preferencia con una herramienta de desarrollo web, para edición HTML como Adobe Dreamweaver o herramientas parecidas

```
240
241      <group name="Sinks">
242
243          <block name="To Workspace" npIN="1" npOUT="0" sWIN="0.6" desc="[-BR-]Write input to speci
       or structure in MAILAB's main workspace. Data is not avaible until the simulation is stopped or
      enable="1">
244
245              <start name="stNameBlock" values=""/>
246              <start name="stAccor" values="v,|_|,simout"/>
247              <start name="stPanel" values="v,|_|,VariableName"/>
248
249              <param name="TextInput" label="Variable name:" varname="VariableName" def="simout"/>
250              <param name="TextInput" label="Decimation:" varname="Decimation" def="1"/>
251              <param name="ComboBox" label="Save format:" varname="SaveFormat" def="Structure With
      Time,|_|,Structure,|_|,Array" validation="" image="" ports=""/>
252
253          </block>
254
255          <block name="webScope" npIN="5" npOUT="0" sWIN="0" desc="" enable="1">
256
257          </block>
258
259      </group>
260
```

Figura 115: Vista del archivo mdl en Dreamweaver

```
<group name="Sinks">
        <block name="To Workspace" npIN="1" npOUT="0" sWIN="0.6" desc="[-BR-]Write input to specified
                <start name="stNameBlock" values=""/>
                <start name="stAccor" values="v,|_|,simout"/>
                <start name="stPanel" values="v,|_|,variableName"/>

                <param name="TextInput" label="Variable name:" varname="VariableName" def="simout"/>
                <param name="TextInput" label="Decimation:" varname="Decimation" def="1"/>
                <param name="ComboBox" label="Save format:" varname="SaveFormat" def="Structure With
        </block>
        <block name="webScope" npIN="5" npOUT="0" sWIN="0" desc="" enable="1">

        </block>
</group>
```

Figura 116: Vista del archivo mdl en el bloc de notas de Windows

Modificar y agregar un campo a determinado bloque en el XML

- Al presente ejemplo, se va agregar un campo más al bloque de 'To Workspace', la variable de 'Sample Time'
  - Según se puede apreciar en MATLAB, ese tipo de variables es solo ingreso de texto, para lo cual en el XML, teniendo como referencia la base presentada, se utilizará el siguiente ejemplo: <param name="TextInput" label="**Decimation:**" varname="**Decimation**" def="**1**"/>
  - Para agregar la variable de 'Sample Time' con los datos que se obtuvieron del modelo utilizado, nos queda de la siguiente forma: <param name="TextInput" label="**Sample time (-1 for inherited):**" varname="**Sample Time**" def="**-1**"/>
- Se procede a pegar el texto del nuevo parámetro en el XML y guardar
- Al cargar nuevamente el Flash Panel, debe aparecer la opción habilitada para su configuración

```
241     <group name="Sinks">
242
243         <block name="To Workspace" npIN="1" npOUT="0" sWIN="0.6" desc="[-BR-]Write input to specified array
or structure in MATLAB's main workspace. Data is not avaible until the simulation is stopped or paused."
enable="1">
244
245             <start name="stNameBlock" values=""/>
246             <start name="stAccor" values="v,|_|,simout"/>
247             <start name="stPanel" values="v,|_|,VariableName"/>
248
249             <param name="TextInput" label="Variable name:" varname="VariableName" def="simout"/>
250             <param name="TextInput" label="Decimation:" varname="Decimation" def="1"/>
251             <param name="TextInput" label="Sample time (-1 for inherited):" varname="Sample Time" def="-1"/>
252             <param name="ComboBox" label="Save format:" varname="SaveFormat" def="Structure With
Time,|_|,Structure,|_|,Array" validation="" image="" ports=""/>
253
254         </block>
255
256         <block name="webScope" npIN="5" npOUT="0" sWIN="0" desc="" enable="1">
257
258         </block>
259
260     </group>
```

Figura 117: Añadiendo campos al archivo ML de configuración

Modificar y agregar un nuevo bloque en un grupo ya creado en el XML

- Al presente ejemplo, se va agregar un nuevo bloque "Bloque Ejemplo"
- El nuevo bloque a agregar tiene características similares al bloque "To Workspace", el cual utilizaremos como base para generar el nuevo bloque
    - Para lo cual en el editor del XML seleccionamos todo el texto estructurado del bloque "To Workspace" y se procede a copiar

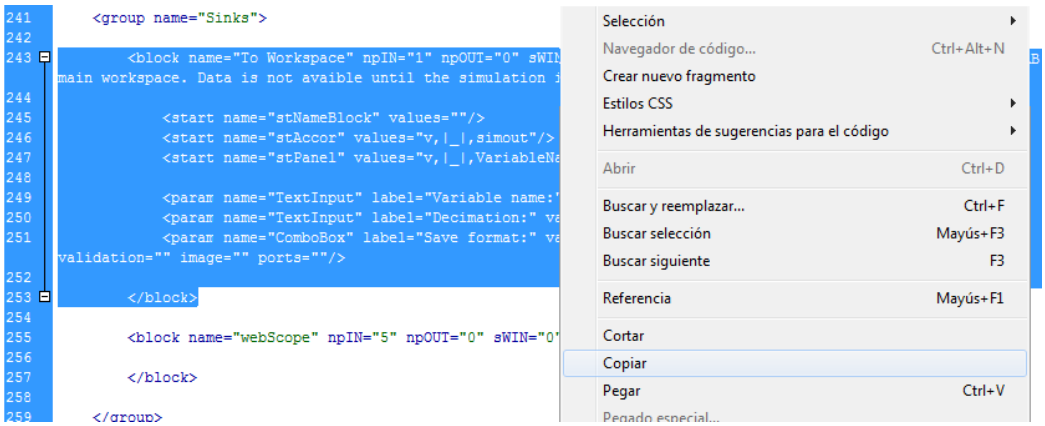


Figura 118: Copiando estructura de un bloque del archivo XML de configuración

- Ubicándose en la posición donde se desee se muestre el nuevo bloque y siguiendo la estructuración planteada del XML, se procede a pegar el texto copiado
    - Se editan los parámetros y configuraciones del bloque respectivamente acorde al bloque a agregar

```
243         <block name="To Workspace" npIN="1" npOUT="0" sWIN="0.6" desc="[-BR-]Write input to specified
    main workspace. Data is not avaible until the simulation is stopped or paused." enable="1">
244
245             <start name="stNameBlock" values=""/>
246             <start name="stAccor" values="v,|_|,simout"/>
247             <start name="stPanel" values="v,|_|,VariableName"/>
248
249             <param name="TextInput" label="Variable name:" varname="VariableName" def="simout"/>
250             <param name="TextInput" label="Decimation:" varname="Decimation" def="1"/>
251             <param name="ComboBox" label="Save format:" varname="SaveFormat" def="Structure With Time
    validation="" image="" ports=""/>
252
253         </block>
254
255         <block name="Bloque Ejemplo" npIN="1" npOUT="0" sWIN="0.6" desc="[-BR-]Write input to specifi
    main workspace. Data is not avaible until the simulation is stopped or paused." enable="1">
256
257             <start name="stNameBlock" values=""/>
258             <start name="stAccor" values="v,|_|,simout"/>
259             <start name="stPanel" values="v,|_|,VariableName"/>
260
261             <param name="TextInput" label="Variable name:" varname="VariableName" def="simout"/>
262             <param name="TextInput" label="Decimation:" varname="Decimation" def="1"/>
263             <param name="ComboBox" label="Save format:" varname="SaveFormat" def="Structure With Time
    validation="" image="" ports=""/>
264
265         </block>
```

Figura 119: Edición de los valores de un bloque en el documento XML

- Para configurar el nuevo bloque, se tiene la siguiente información: <block **name**="Bloque Ejemplo" **npIN**="1" **npOUT**="0" **sWIN**="0.6" **desc**="[-BR-]Write input to specified array or structure in MATLAB's main workspace. Data is not avaible until the simulation is stopped or paused." **enable**="1">
- Cuando se agrega un bloque o grupo al XML, es necesario crear carpetas y asignar imágenes con los respectivos nombres en el sistema de archivos del servidor
    - Estas imágenes, para su mejor visualización, deben ser modificadas con alguna herramienta de diseño gráfico para generar las transparencias del fondo

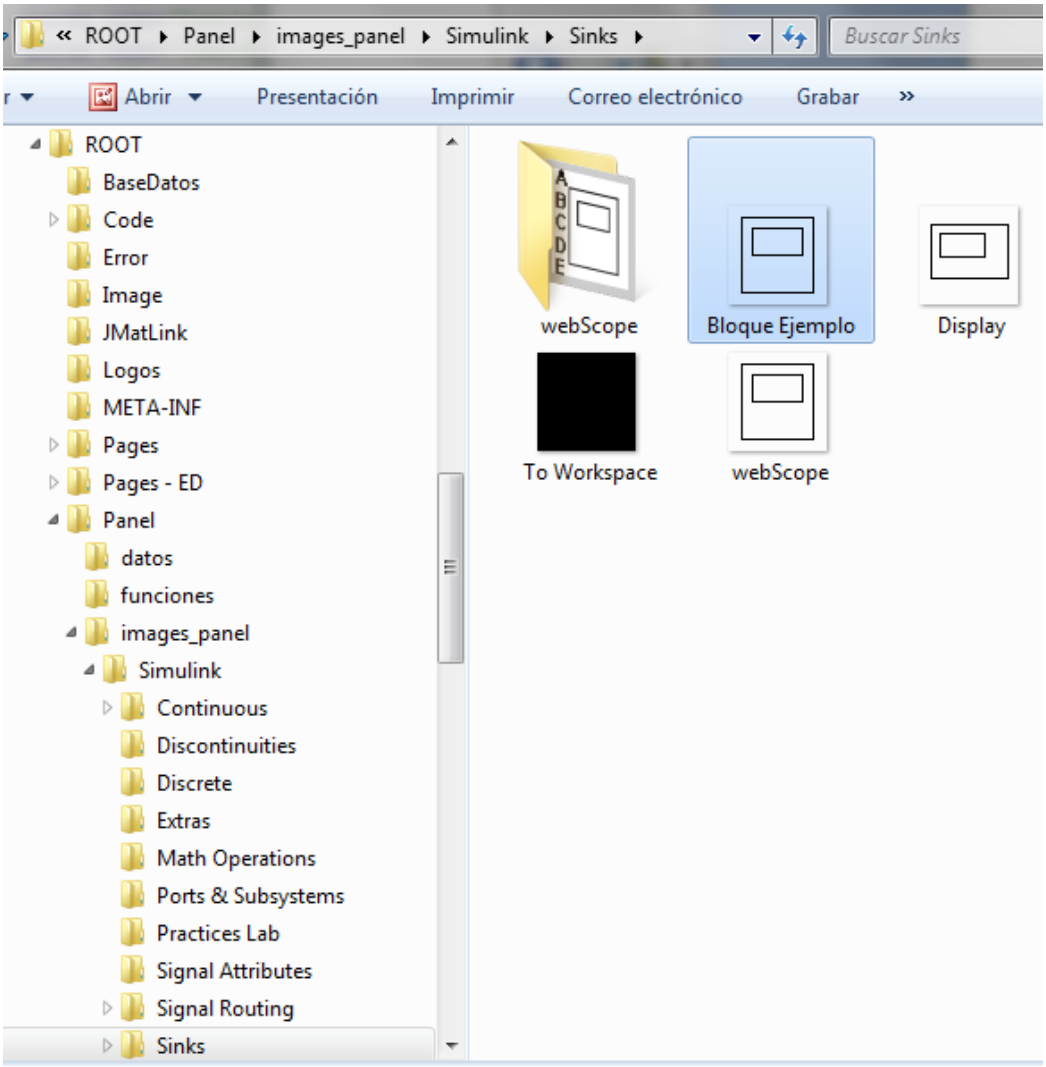


Figura 120: Ejemplo de imágenes configuradas para presentación en bloques

Nota: Existen bloques que son 'casos especiales' los cuales su funcionamiento está desarrollado en la programación del sistema Flash y no puede ser modificado en el XML de datos. Para modificar estos bloques o generar nuevos bloques con comportamientos especiales, se debe modificar el archivo fuente del sistema Flash.

Bibliografía

1.- Hilbert, Martin R. “From Industrial economics to digital economics”. CEPAL. United Nations Publication, Santiago, Chile. 2001. Pág 103

2.- M. Gonzáles, J. Adiego. L. F. Sanz, R. García, C. Hermo. “Can the WWW Help to Reduce the Digital Divide? An Example of Cost Effectiveness in Teaching Laboratory Development”. Proceedings of Technology for Education in Developing Countries TEDC2006 July 2006 Iringa Tanzania. Ed. H.H. Lund et al, IEEE Computer Society, Los Alamitos 2006. ISBN-13:978-0-7695-26331.

3.- Colaboradores Wikipedia. Adobe Flash.
http://es.wikipedia.org/wiki/Adobe_Flash (2010)

4.- Colaboradores Wikipedia. Adobe Flash Player.
http://es.wikipedia.org/wiki/Flash_Player (2010)

5.- Colaboradores Wikipedia. Extensible Markup Language.
http://es.wikipedia.org/wiki/Extensible_Markup_Language (2010)