PROGRAMACIÓN GRAFICA EN LABVIEW

Profesor Luis Alberto Osorio
Universidad de Antioquia

INTRODUCCIÓN

Las técnicas de programación y los lenguajes de programación han recorrido un largo camino durante varias décadas y en su permanente evolución, rápidamente abandonaron el caos inicial de los años 60, para abordar esquemas estructurados durante la década de los 70 en su adolescencia e ingresar inexorablemente en la edad adulta durante la década de los 80 con la programación orientada a objetos en cuyo seno florece un cúmulo de herramientas de programación gráfica, paradigmas de productividad y eficacia en la generación de aplicaciones en todos los campos del conocimiento .

Si en la actualidad estamos clamando por esquemas pedagógicos revolucionarios que rompan con la complicidad en la rutina del discurso del profesor y la rutina de la repetición insulsa del estudiante, ajenos al verdadero método científico y al ejercicio de la creatividad, encontramos realizaciones ampliamente difundidas como conclusiones del último congreso nacional de profesores universitarios (1996). En la programación gráfica observamos claramente una gran alternativa en el mejoramiento del proceso enseñanza-aprendizaje y muy especialmente en crear conciencia de productividad intelectual que allane el camino hacia la verdadera investigación en la búsqueda de nuevos conocimientos .

LABVIEW es una herramienta de programación gráfica de propósito general por excelencia que ofrece muy amigablemente un ambiente propicio para poner a prueba la claridad del concepto y la maestría del conocimiento en el nivel de abstracción preferido por el físico, el matemático, el estadístico y en mayor grado, el ingeniero.

EL AMBIENTE LABVIEW

La herramienta de programación y simulación LABVIEW presenta el concepto VI (Virtual Instrument) como elemento central de su filosofía cuyo principal objetivo es el empleo de un P.C. de propósito general para generar por software instrumentos virtuales apoyados en el uso extensivo de su infraestructura : Teclado , mouse , memoria, CPU , disco e interfaces. El recurso más sobresaliente lo puede constituir una tarjeta de adquisición de datos multicanal de alto rendimiento con D.M.A., temporizado, disparo (Trigger) externo por hardware o por programa , sincronización y acople entre tarjetas múltiples cuando las hay .

Labview es para el ingeniero y el científico, lo que la herramienta hoja de cálculo (spreadsheet), es para la comunidad financiera o por lo menos esa ha sido la intención de sus creadores KODOSKY, TRUCHARD y McCRISKEN. Los orígenes del Labview se atribuyen a un desarrollo previo por Truchard y Kodosky en A.R.L. (Applied Research Laboratory) a finales de la década del 70 y que consiste en un sistema de pruebas para transductores de navíos de la marina en USA. La flexibilidad de tal herramienta permitía que un técnico efectuara procedimientos de chequeos preestablecidos.

En otro nivel de interacción, un ingeniero en acústica submarina podía diseñar sus procedimientos de prueba e incorporarlos a las librerías . Para el investigador brindaba un nivel de acceso total incluso hasta la programación del hardware para reconfigurarlo a sus necesidades . Aun cuando el tiempo de desarrollo de tal sistema fue de 18 hombres /año el producto final resultó poco amigable al usuario final quien debía familiarizarse con infinidad de nemónicos y extensiva manipulación de menús para lograr algún cambio.

El refinamiento de la filosofía del diseño condujo a Kodosky al concepto jerárquico de instrumentos virtuales, compuestos a su vez de otros sub-instrumentos que en el nivel más bajo habrían de conformar los bloques de software más elementales: los computacionales y las operaciones de I/O. A la interconexión y anidamiento de los diferentes niveles del software se les dio una estructuración homogénea de manera que los VI en cualquier punto de la jerarquía, siguen las mismas reglas en su construcción. Con lo anterior se logró impartir gran productividad al proceso

de diseño de aplicaciones, manteniendo la habilidad computacional de los lenguajes de programación y adicionando el paralelismo en ejecución de los programas concurrentes .

El uso de la herramienta Labview también puede ubicarse en diferentes niveles de sofisticación el cual se empieza por el usuario que simplemente busca procesar unos datos y rápidamente obtener resultados sin los «detestables « enredos de la sintaxis y codificación de programas. Este tipo de usuario simplemente recurre a las bibliotecas (Libraries) de aplicaciones y escoge la que presumiblemente se adapta a sus requerimientos. En este esfuerzo por suministrar soluciones a usuarios, seguirán apareciendo ofertas de compañías independientes que desarrollan Labview.

El mayor beneficio para el ingeniero y para el usuario creativo se encuentra en la técnica de programación gráfica que enfatiza los bloques funcionales que interconectados en una relación uno a uno con la conceptualización de la solución a un diseño en la mente del ingeniero, le proporciona en muy corto tiempo un esquema, una técnica de diagramación que Labview en forma transparente se encarga de utilizar como programa fuente. Este código gráfico se somete luego a la acción mágica de un compilador gráfico, para finalmente generar un código ejecutable correctamente ligado a bibliotecas referenciadas. Entre las características de la modalidad de programación gráfica de Labview que rápidamente afloran, pueden enumerarse algunas como:

* Ejecución de eventos bajo la modalidad de diagramación de flujo de datos que condiciona la ejecución a la disponibilidad de la totalidad de los datos en cualquier punto del diagrama.

* Solución a la débil capacidad de computación interactiva y condicional de la diagramación tradicional de flujo de datos mediante la integración de estructuras gráficas de programación que incorporaron los ineludibles lazos repetitivos de los lenguajes de programación estructurada . Las estructuras gráficas WHILE, FOR, IF, CASE, SEQUENCE se incorporan al concepto de flujo de datos en la forma de rectángulos de variadas formas que abarcan gráficamente un conjunto de tareas asociadas con cada estructura. En los bordes de cada rectángulo se acomodan los diferentes elementos semánticos que caracterizan cada lazo. El resultado de estos diagramas estructurados de flujo de datos es un conjunto de eventos paralelos independientes o interactivos sujetos a las reglas de flujo de datos en su ejecución . Puesto que la dinámica del flujo de datos en la ejecución de cada estructura es importante, para poder mantener un control de ejecución secuencial de diferentes eventos, se debe resaltar la inclusión de la estructura SEQUENCE que con la similitud de los cuadros en películas proporcionan la funcionalidad deseada.

* Polimorfismo en operadores y funciones que deben procesar los tipos de datos y variables que en caso de conflicto se unifican automáticamente en representación de punto flotante. Los bloques computacionales y operativos invariablemente ajustan la naturaleza del cómputo al tipo de variables suministradas como operandos. En otras palabras, el bloque computacional se adapta a la naturaleza de enteros, complejos, arreglos de sus operandos de entrada, sujetos, claro está, a ciertas reglas cuando, por ejemplo, se plantea le suma de un entero con un arreglo. La regla será: la suma del entero a cada elemento del arreglo etc.

* La existencia de una interfaz de usuario en cada componente (V.I.) del instrumento virtual global. Cada interfaz de usuario es una parte integral del modelo de programación que permite verificar interactivamente la operatividad de cada módulo componente. Las herramientas gráficas de alto rendimiento como las funciones del software Quickdraw del Macintosh, permitían la inmediata interactividad gráfica con los objetos del panel de control del instrumento virtual y fueron el escogimiento inicial para materializar hacia 1985 la interfaz gráfica deseada. El procesador Motorola 68xxx con su terso espacio de direccionamiento directo, sin complicaciones de segmentación y en un ambiente de sistema operativo integral y ajeno a las restricciones en espacio de direccionamiento de DOS, ofrecía las condiciones ideales para la manipulación de grandes bloques de datos. El lanzamiento de Windows 3.0 para plataforma I.B.M P.C permitió en 1991 la aparición de la primera versión del Labview para P.C. 5 años después de la primera versión del Macintosh + con las mismas posibilidades gráficas de la interfaz de usuario.

OTROS ANTECEDENTES

Hasta la aparición del P.C en 1980, la totalidad de laboratorios de física e ingeniería empleaban ya diversidad de instrumentos interconectados para registrar y controlar sus sistemas bajo prueba. Los puertos de comunicación y control de cada instrumento se ajustan a la norma IEEE-488 también conocida como GPIB (General Purpose Interface Bus) . Entre 1980-83 la Cia. National Instrument estableció un sólido liderazgo como proveedora de interfaces GPIB para P.C's y minicomputadoras para el control y monitoreo remoto de instrumentos de laboratorio .

La programación BASIC fue la base inicial para el software de control de la instrumentación que requería en cada montaje de medición y prueba de idóneos ingenieros programadores , conocedores de la instrumentación y con sólida fundamentación en el proceso o experimento para generar un programa de prueba exitoso. Con la experiencia acumulada en el desarrollo de programas para el control de instrumentación remota con interfaces GPIB, aunada a la expectativa de madurar una herramienta de software que trascendiera la lograda en A.R.L. , los creadores del Labview abordaron la monumental tarea pretendiendo crear para el ingeniero y el científico un producto para simulación y control que tuviera el mismo impacto que la hoja de cálculo en aplicaciones financieras .

La versión Labview aparece en 1990 luego de un total diseño de la versión 1.2 con técnicas de programación OOP (Object-Oriented Programming). En el momento del inicio del diseño en 1988 las herramientas OOP del lenguaje C eran tan primitivas que la N.I tuvo que recurrir al diseño mismo de herramientas de desarrollo apoyadas en la funcionalidad de hojas de cálculo. A partir de 1990, Labview adquiere la jerarquía mundial de herramienta de simulación , control y monitoreo de procesos. El elemento más significativo en las nuevas versiones 2.5, 3.0, 3.1, es el compilador gráfico incorporado al programa de manera tan transparente al usuario que, la compilación y encadenamiento tradicionales, se efectúan imperceptiblemente a partir de los diagramas de bloques.

En el terreno de generación de códigos en forma entendible por Labview y compilado externamente, la NI reporta únicamente la existencia del compilador WATCOM-C que brinda la compatibilidad requerida cuando de generar bloques de código CIN se trate. La intención, en todo caso, no es la de fomentar la proliferación de otros paquetes y herramientas, sino lo contrario: la de tener una infraestructura autosuficiente que en efecto se logra siempre mediante el manejo de hardware por los CIN integrados con muchas funciones básicas del Labview.

Las formidables exigencias que caracterizan la instrumentación para medición y control de experimentos y fenómenos físicos, en términos de potencia operativa y seguimiento en tiempo real, fácilmente permiten descartar la P.C. como el instrumento mismo que pretenda remplazar con software los numerosos instrumentos que pueden eventualmente requerirse en un arreglo experimental. La eficiencia y diseño especializado de instrumentos digitales optimizados, con su particular integración hardware-softtware hacia tareas muy específicas, los hace insustituibles en la mayoría de los casos. Labview tampoco pretende reemplazarlos y menos descargar toda esa responsabilidad en la C.P.U. del P.C, puesto que existe una conciencia total de la carga que tan solo el manejo de gráficos en el panel frontal impone al desempeño del paquete .

La filosofía del paquete como era de esperarse, se orientó al monitoreo y control de instrumentos remotos de muchos fabricantes a través de VI's denominadas Instrument Drivers , muchas veces desarrolladas por ellos mismos . Estos Drivers han sido ampliamente difundidos por N.I de manera gratuita a todos los usuarios clientes de Labview y entre ellos se incluyen los que interactúan con tarjetas de red para operar integralmente en ambientes de control distribuido o ambientes de menor jerarquía pero que requieren de las bondades de la conectividad en el manejo de información . En este contexto pueden existir varios sistemas Labview conectados en red , con la carga de trabajo en tiempo real y actividades interactivas con el usuario distribuidas entre ellos .

Con el sólido soporte en red del Labview se puede tener una configuración donde una máquina P.C. efectúa el control de tiempo real con adquisición de datos y atención a canales I/O. Otra máquina se destina a comunicarse periódicamente con la primera tanto para descargarla del cúmulo de datos adquiridos en tiempo real y procesarlos debidamente, como para entregarle

parámetros y nuevos requerimientos exigidos por el operario sin temor de interrumpir delicadas tareas mientras la C.P.U de esta segunda máquina, interactúa con el operario. El software de la máquina dedicada a operar en tiempo real es el encargado de llevar a cabo las tareas críticas y decidir cuándo habilita el canal de comunicación con la máquina supervisora para aceptar nuevos requerimientos operativos.

El uso extensivo del P.I.C's o controladores inteligentes en el ambiente de mayor proximidad al proceso mismo, sigue siendo una mejor alternativa por la confiabilidad, rapidez de respuesta en tiempo real y recientemente por la conectividad a estaciones que administran la interfaz operario-máquina en ambiente Labview, para brindar en conjunto sistemas de control con resolución de 1 mseg.

Es importante resaltar las limitaciones de velocidad de respuesta Labview bajo cualquier sistema operativo que lo soporte , aceptado que la complejidad de los mismos los ubica no precisamente como ambientes óptimos para aplicaciones en tiempo real con tiempos de muestreo y ciclos de programa muy pequeños. Este campo de aplicación nos obliga a definir más exactamente los rangos de tiempo de ciclo para diferentes procesos.

En un sistema de control de temperatura centralizado en un hogar promedio, el tiempo de ciclo es del orden de 1 minuto. En un sistema de regulación de presión el tiempo de ciclo no debe ser mayor a 1 seg. El tiempo de muestreo en el control de vuelo de un avión de combate es del orden de los 10 mseg. Solo de esta manera se materializa la exigencia o marco de referencia para la operación en tiempo real.

El soporte en hardware para adquisición de datos como se verá, permite operar confiablemente a frecuencias de varias decenas de Kilohertz, controlando por hardware o software el inicio y terminación del proceso de adquisición. Algo similar ocurre con la operación de temporizados, en la medición de eventos externos por unidades de tiempo, medición del período de un evento etc. La limitante de tiempo de ejecución de una aplicación Labview se distribuye entre manipulaciones de gráficas, manipulaciones de estructuras de datos especialmente de tipo string [cadena de caracteres], conversión entre tipos de datos, que obligan al escogimiento de funciones para manipular arreglos que no requieran intervención permanente del manejador de memoria, etc.

Con el empleo de Labview en el entorno de supervisor, la disponibilidad de drivers para todo tipo de interfaces y protocolos propietarios es asombrosa. La conectividad a redes de área local especiales, también ha sido abordada por fabricantes de módulos VXI que transmiten y reciben datos bajo protocolos diversos (ARINC en la red DITS en la industria de aeronaves comerciales, MILSTD-1533A/B en la red de aviación militar etc.) al tiempo que se conectan a la red de monitoreo LABVIEW, una de las cuales es la GPIBENET que permite el control de instrumentos en un enlace ETHERNET bajo protocolo standard TPC/IP . Para equipos con interface SCSI existen igualmente módulos de conversión a GPIB. Es conveniente aclarar sin embargo que el protocolo TCP/IP soportado por LABVIEW únicamente se ofrecía en plataformas SUN-UNIX hace muy poco tiempo.

REQUERIMIENTOS DE MAQUINA

La versión LABVIEW 3.1 debe correrse preferiblemente en una P.C bajo WINDOW 3.X , procesador 486DX-50 Mhz (coprocesador indispensable), memoria RAM mínima de 8 MBytes (Preferiblemente 16 M.Bites), Monitor SVGA .28. En aplicaciones tan comunes como adquisición de datos, la representación de los mismos exige como mínimo 4 bytes por muestra y hasta 16 bytes dependiendo de la precisión en la representación del dato. Una imagen con resolución SVGA (800 x 600) requiere un byte por cada pixel en formato de 256 colores, de 1/2 millón de bytes, cantidad que puede cuadruplicarse si se recurre a conversiones de tipo de datos para efectos computacionales. Una aplicación LABVIEW está compuesta de una cantidad apreciable de pantallazos gráficos según el anidamiento de SUBVI's empleadas. Esto nos da una idea del por qué de los requerimientos de memoria RAM.

El paquete de software Labview exige alrededor de 50 Mega-bytes de espacio en disco para su completa instalación en una P.C. (64 MB en una estación Sun). Los requerimientos adicionales de disco son fácilmente estimables según el tipo de datos que se vayan a almacenar y el número total de puntos o muestras.

Las cantidades de punto flotante almacenadas en formato binario en precisión sencilla ocupan 4 bytes por cada punto mientras que si se convierten en formato ASCII pasarán a ocupar hasta 12 bytes por muestra, de manera que la demanda de espacio en disco es perfectamente

predecible por cada usuario. Adicionalmente siempre está disponible el disco flexible de 1.44 M.B. para descongestionar periódicamente el espacio de disco duro si fuere necesario.

COMO INICIARSE EN LABVIEW.

La primera actividad estará seguramente orientada a explorar las librerías de LABVIEW, simplemente yendo de directorio en directorio y de rama en rama al acostumbrado estilo Windows, tratando de localizar directorios con extensiones LLB que al entrar a ellos presentan una variedad de archivos especiales con extensión .VI. Con la selección de algunos de estos archivos culmina la búsqueda que se ha iniciado con un comando OPEN del menú FILE que, como en toda aplicación Windows pertenece a la tradicional barra de comandos. Las posibilidades de explorar están ordenadas temáticamente en las categorías siguientes:

* ANALISIS: DSP, estadística, generación de señales, mediciones.
* Aplicaciones: Adquisición de datos y técnicas en control.
* DAQ: I/O análogo y digital, temporizadores.
* Archivos (files): ASCII, Binarios, Datalog, Hoja de cálculo.
* General: Estructuras de programación, Gráficos.
* Imágenes: Paquete Pictures .VI.

El path: \EXAMPLE\GENERAL es una buena selección para llevar a cabo éste tipo de exploración . La ejecución de una aplicación particular exige tan sólo presionar el botón RUN de la barra de control manipulando luego los controles en el panel frontal del instrumento en cuestión, hasta terminar muy seguramente presionando algún botón programado como STOP en el mismo panel frontal.

El segundo nivel de actividad está encaminado a mostrar la filosofía operativa y la composición de una aplicación, las herramientas de monitoreo y depuración existentes y en general los pasos por seguir en la creación de una aplicación (VI)que sea la fiel materialización de la simulación de algún evento que pueda representarse algorítmicamente en el lenguaje visual del LABVIEW. En este proceso es irremplazable la guía escrita por N.I. y denominada LABVIEW TUTORIAL MANUAL, la cual propone una serie de ejercicios, que no solo exponen la mecánica del lenguaje gráfico sino que también confeccionan paso a paso aplicaciones sencillas ejecutables por el usuario.

El proceso de creación, una vez superada la mecánica operativa con el tutorial, conduce necesariamente a consultas permanentes de otros tres (3) manuales como son:

* LABVIEW for Windows User Manual.
* LABVIEW Analysis VI Reference Manual .
* LABVIEW for Windows Data Adquisition Reference Manual .

En cuanto al procedimiento mismo de empezar a escribir programas en el agradable ambiente gráfico de LABVIEW es indispensable primero analizar exhaustivamente varias aplicaciones, entendiendo sin omitir detalle, la operación pormenorizada de cada bloque, de cada lazo, el uso de variables locales y variables globales dando particular atención a la concatenación tan especial de las variables tipo SHIFT REGISTER cuando existe anidamiento de lazos (Procedimientos). El orden de ejecución de los lazos anidados continúa siendo de afuera hacia dentro de tal manera que el lazo externo podrá iterar una vez más, solo cuando la totalidad de operaciones en lazos internos se haya completado. El control de ejecución de un lazo WHILE puede dejarse sin conexión alguna, lo cual implica la ejecución del lazo una sola vez. En la mayoría de los casos algún control manual del tipo booleano en el panel de control estará alambrado en el diagrama de la aplicación al control de ejecución del lazo. En otros casos se genera por programa una condición booleana que alambrada al control de ejecución determinará la finalización de la ejecución del lazo mencionado.

El control de ejecución de un lazo WHILE es verificado en paralelo por LABVIEW como una tarea más en el esquema de flujo de datos ya aludido, para determinar si el índice de iteración se incrementa una vez más. Por ello cuando aparentemente el programa no responde al accionamiento de un interruptor (control) de parada (stop) en el panel de control, se debe a que el lazo aún tiene tareas por ejecutar antes de verificar el estado del control booleano, o después de haberlo verificado.

Para resaltar en la programación gráfica como novedad, está la facilidad de crear las referidas variables SHIFT-

REGISTER que almacenan valores computados en iteraciones anteriores. La naturaleza de las variables por ellas representadas es dependiente de los objetos y estructuras de datos que el programador invoque durante la implementación algorítmica de cada procedimiento o lazo donde están definidas las SHIFT REGISTER.

La ubicación gráfica de estos elementos se circunscribe al contorno del lazo, sea éste de tipo WHILE o de tipo FOR; dicho contorno o delimitación gráfica de tareas pertinentes también va a contener túneles (que se generan automáticamente al paso de una conexión o «alambre» hacia o desde el interior del lazo), arreglos (que se generan automáticamente por cómputo iterativo de sus elementos dentro de un lazo FOR), condicionales (para la selección automática de procedimientos alternos dentro de una estructura). CASE (IF si la decisión es controlada booleanamente), o simplemente MARCAS para relacionar lazos y establecer de esta manera una dependencia de flujo de datos que condiciona la secuencia de ejecución como se había anticipado y como alternativa para el empleo de la estructura SEQUENCE.

En la presentación de ejemplos sobre programación gráfica será fácil apreciar el frecuente empleo de lazos while que ejecutan una sola vez cuando son invocados pero con la particular característica de retención (mediante el uso de una variable Shift Register) del valor generado en la anterior invocación y que incidirán, dependiendo del algoritmo, en el cómputo del nuevo valor, motivo de la invocación. La única condición para lograr una correcta operación es la NO inicialización de la variable S.R. desde el exterior del lazo de ejecución única.

La apariencia del panel frontal de la aplicación LABVIEW simula, la de un instrumento físico con sus respectivos botones, perillas, selectores y pantalla de gráficas para monitoreo de variables, configurando así 2 categorías de elementos llamados CONTROLES e INDICADORES. El programa fuente de la aplicación es el diagrama esquemático mismo que de alambrarse (interconectarse) apropiadamente equivaldrá a un planteamiento sintácticamente correcto . Como en todo esquema de programación la aprobación de la prueba sintáctica es efectuada por LABVIEW de manera automática. De ningún modo garantiza planteamientos correctos de la lógica de la solución. La persistencia del ícono de ejecución como una flecha quebrada (==//==>) es el indicativo de un error de interconexión entre bloques o la ausencia de una conexión obligatoria.

Está permitido sin embargo, el dejar controles o indicadores sin conectar y aún poder ejecutar la aplicación sin incurrir en errores de sintaxis. El concepto de modularidad y jerarquización de aplicaciones es el tercer ingrediente en un programa LABVIEW. Al estilo de subrutinas y procedimientos en otros lenguajes , toda V.I. puede reducirse a un Icono/conector para convertirse en una lista de parámetros gráficos a través de los cuales otra aplicación de mayor jerarquía pueda invocarla y pasarle parámetros y en consecuencia retornar valores, funciones o simplemente realizar una tarea. El editor requerido par crear el ícono de la V.I. también está incorporado en el modo de edición y basta con invocar (presionando y seleccionando con el botón derecho del mouse) la opción apropiada para edición del ícono o para la asignación de terminales del conector a elementos del panel de control.

La inclusión de una SUB V.I en la librería (Biblioteca) VI. Lib permitirá, sin mayores rodeos, el poder invocarla posteriormente desde el programa fuente de otra aplicación en desarrollo , con la simple selección de su ícono a partir de un POP-UP del menú de funciones , opción V.I..., (un Pop Up es el resultado de oprimir el botón derecho del mouse bien sobre un espacio libre o en el panel frontal o en el diagrama de bloques , o bien sobre algún elemento ya instalado en uno de los dos ambientes).

Para finalizar esta corta introducción no puede pasarse por alto la importantísima presencia de la "Ayuda en Línea" que puede activarse con la opción Window de la barra de comandos y que la incluye en su respectivo menú. La combinación de teclas CNTRL-H es una alternativa rápida para activar y desactivar la ventana de ayuda especialmente en la modalidad de OPERACIÓN cuyo ícono es la "mano" . La ventana de ayuda en línea es útil tanto para análisis de un diagrama de una VI, como para el proceso de alambrado de una aplicación. Cada bloque funcional de un esquemático está plenamente documentado en dicha ventana, lo cual permite en muchos casos no tener que recurrir a los manuales, con solo ubicar la flecha del mouse sobre el bloque elegido.

Si estamos conscientes de que adicionalmente, con un doble clic sobre cualquier elemento (sub VI) del diagrama se nos proporciona la información total (panel de control y diagrama respectivo) sobre el elemento, podemos con la ventana de ayuda activada escudriñar cada detalle de una aplicación. El límite de este procedimiento, rápidamente se concluye, está en la visualización de funciones básicas y bloques CIN.

Cuando se enfrenta una aplicación que presumimos puede sernos de utilidad inmediata en la solución a una necesidad , nada más oportuno que disponer de la documentación que señale los alcances y limitaciones de la aplicación . Para esto generalmente en la opción GETINFO del menú FILE se nos presenta una ventana con la información deseada. Se recomienda por lo tanto que a toda aplicación que se desarrolle como VI, se le adicione la documentación precisa sobre el propósito y alcance de la misma apelando al cuadro para descripciones de la opción GET INFO.

El polimorfismo en las funciones de LABVIEW se manifiesta admirablemente en los gráficos para despliegue de datos; simplemente juntando arreglos unidimensionales para conformar arreglos bidimensionales podemos desplegar 2,3, o más curvas simultáneamente. Basta con alimentar la función BUILD ARRAY con los N arreglos unidimensionales (1xN) a la misma función polimórfica denominada GRAPH (o CHART), la que se ajusta automáticamente, a la dimensión del arreglo suministrado.

La herramienta de programación gráfica que aporta LABVIEW es tan flexible que puede emplearse en muchas disciplinas académicas como simulación estadística, experimentos en física, simulación de sistemas de control etc. En otro artículo se detallará la aplicación de la programación LABVIEW en la simulación y práctica del control automático. Es muy importante alertar al usuario que por primera vez intenta hacer modificaciones en alguna VI en disco, que LABVIEW tiene un nivel de seguridad casi nulo, lo que infortunadamente permite ocasionar serios daños a una aplicación. Por esto es importante, al abandonar una exploración de alguna VI en memoria, manifestar al LABVIEW que NO se desea guardar la versión modificada en ella; las alternativas de trabajo apuntan hacia la generación copias con otros nombres que puedan cargarse luego y experimentar con ellas al gusto del usuario.