Universidad del Azuay

Facultad de Ciencia y Tecnología

Escuela de Ingeniería Electrónica

Diseño de un sistema HMI/SCADA para una planta de clasificación con Visión Artificial.

Trabajo de graduación previo a la obtención del título de Ingeniero Electrónico.

Autores

Edgar Fernando Lojan Bermeo

Daniel Adrián Iñiguez Quesada

Director

Ing. Hugo Torres Salamea.

Cuenca, Ecuador
2009

Este trabajo monográfico es un complemento del curso de graduación de “Especialización en Control Automático y Automatización” que se realizó en la Universidad de Buenos Aires, Argentina, en la Facultad de Ingeniería de la Escuela de Graduados en Ingeniería Electrónica y Telecomunicaciones (EGRIET), cuyo inicio fue en septiembre del 2008 y finalizó en octubre del mismo año.

DEDICATORIA

Dedico este trabajo con todo el amor del mundo a mi querida hija Madeline y a mi mujer mis fuentes de amor e inspiración.

Edgar Lojan

AGRADECIMIENTOS

Agradezco en primer lugar a Dios por brindarme salud y sabiduría para alcanzar una meta más en mi vida.

Al Ing. Hugo Torres por su acertada dirección en el desarrollo de éste trabajo.

A mi Universidad, UDA, por brindarme una buena formación académica y profesional. A todos los profesores que han impartido y compartido sus conocimientos, por sus sabios consejos y sugerencias para ser personas de bien.

A mi familia por todo el apoyo y sacrificio brindado, en especial a mis padres quienes siempre me han incentivado a estudiar y superarme.

Un agradecimiento muy especial a mi Tío Darío por su apoyo económico y moral para lograr alcanzar una meta más en mi vida.

Edgar Lojan

Agradezco a Dios, a mis padres a la Universidad Del Azuay y a su personal docente, también de manera especial al Ing. Hugo Torres por su acertada dirección en la elaboración de éste trabajo.

Daniel Iñiguez

RESUMEN

La presente tesis monográfica trata acerca del "Diseño de un sistema HMI/SCADA para una planta de clasificación mediante Visión Artificial", en éste caso para dos tipos diferentes de figuras (cubo y cilindro). Para el desarrollo del software se utilizará el lenguaje de programación gráfico LabVIEW versión 8.6, además se usarán los módulos *LabVIEW DSC* y *NI VISION DEVELOPMENT* para realizar la aplicación SCADA y de Visión Artificial respectivamente. También se construirá una maqueta para simular el proceso de clasificación de las piezas. Para la adquisición de las imágenes se va a utilizar una WebCam.

ABSTRACT

The present monographic thesis deals about the "Design of a HMI/SCADA system for a plant of classification by means of Machine Vision" in this case for two types different from figures (cube and cylinder). For the development of the software will be used the graphical programming language LabVIEW version 8.6, also there will be used LabVIEW DSC module and NI Vision Development module to realize the application SCADA and Machine Vision respectively. Also a mock-up will be constructed to simulate the process of classification of the pieces. For the image acquisition a WebCam is going to be used.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

Lojan Bermeo Edgar Fernando.
Iñiguez Quesada Daniel Adrián.
Ing. Hugo Torres Salamea.
Octubre 2009.

DISEÑO DE UN SISTEMA HMI/SCADA PARA UNA PLANTA DE CLASIFICACIÓN CON VISIÓN ARTIFICIAL.

INTRODUCCIÓN

En la actualidad el escenario industrial tiene la tendencia de expandirse hacia un ambiente totalmente automatizado, a la vez que las aplicaciones de monitoreo y control son más necesarias y complejas. Aparece entonces la necesidad de implementar Softwares potentes y amigables, para realizar aplicaciones económicas y de gran utilidad para la industria, sobre todo en materia de control y gestión.

Dentro de un proceso automatizado también se hace necesaria la implementación de una interface visual aprovechando las potencialidades y las características del computador y software con que se trabaje; dichas interfaces se realizan con el propósito de actuar directa ó indirectamente sobre los diferentes procesos y tener una idea mediante animaciones de cómo está trabajando la planta, además de que proporcione la suficiente información del proceso en el momento en que se necesite, creándose así las llamadas interface Hombre Máquina ó simplemente HMI.

Además dicha interface debe ser capaz de interactuar con el sistema de control para iniciarlo, detenerlo, cambiar valores de base, setpoints, conteo de piezas, toma de decisiones, etc., a este sistema se le conoce como Control Supervisorio ó SCADA.

Hay que tomar en cuenta al momento de realizar el diseño de un sistema HMI/SCADA que software utilizar, debido a que existen una gran variedad de Softwares dedicados a éste fin, en éste caso se ha preferido trabajar con el software LabVIEW, que es un software de alto nivel, líder en medición y automatización basada en PC y que permite gracias a su entorno de programación gráfico desarrollar proyectos de mejor calidad y mayor eficacia.

LabVIEW permite integrar módulos adicionales como LabVIEW DSC para crear aplicaciones HMI/SCADA, "*NI VISION 8.6 DEVELOPMENT MODULE*" para crear aplicaciones de Visión Artificial, etc., permitiendo así integrar todo dentro de un solo software y con un mismo lenguaje de programación creando potentes aplicaciones.

Los Sistemas de Visión Artificial basados en PC, es un tema importante y cada vez de más uso dentro de la industria en aplicaciones de control, medición de rasgos, reconocimientos de objetos, identificación de partes, clasificación, verificación de presencia, conteo, etc.

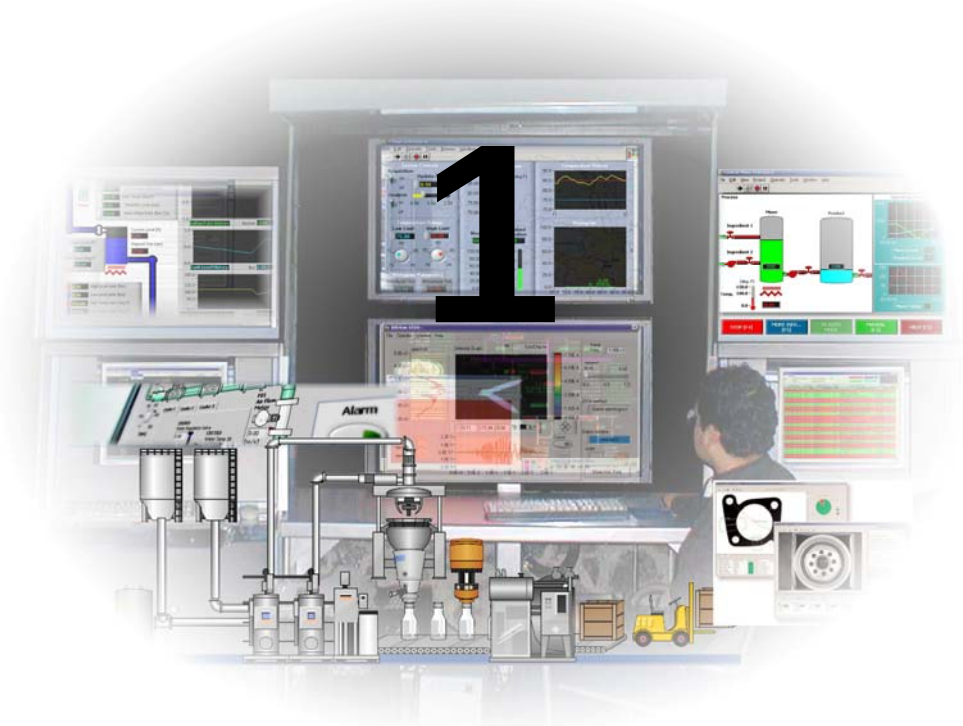
Lo que se plantea realizar en esta presente tesis monográfica es el Diseño de un sistema HMI/SCADA para un prototipo de planta de procesamiento genérico con clasificación mediante visión artificial de dos tipos diferentes de piezas genéricas utilizando el software LabVIEW de National Instruments Corporation. En este caso, la idea principal es construir una maqueta para simular el proceso antes mencionado, la misma que contará con una banda que va ha transportar las piezas. Las piezas deberán pasar por un horno para simular algún proceso en el que influya la temperatura, luego las piezas serán llevadas a donde se encuentra la Webcam para realizar el proceso de visión artificial, que en éste caso es el reconocimiento de patrones y finalmente dependiendo que tipo de pieza sea, será clasificada en la banda correspondiente y en caso de que no sea reconocida como ninguno de los dos modelos será tomada como pieza en mal estado esto para realizar un control de calidad de las piezas.

El software va ha ser desarrollado en LabVIEW aprovechando como se había mencionado en párrafos anteriores la ventaja que en un solo software se puede integrar aplicaciones de visión artificial, sistemas SCADA y aplicaciones de Control.

Para la adquisición y generación de señales se va ha usar el módulo DAQ USB 6008/9 de National Instruments.

Los capítulos 1 y 2 es la parte de conocimientos previos teóricos de que es un sistema HMI/SCADA, que es LabVIEW y que es Visión Artificial con aplicaciones de éste en LabVIEW, mientras los capítulos 3, 4 y 5 se hace referencia al diseño de la maqueta, diseño del software y las pruebas pertinentes del software y hardware, respectivamente.

CAPÍTULO



Sistemas HMI/SCADA

- Interfaz Hombre Máquina.
- Sistemas SCADA.

CAPÍTULO 1

SISTEMAS HMI/SCADA

1.1 Introducción

Antiguamente los procesos industriales se supervisaban manualmente, es decir, se iba anotando en un registro (papel) los productos, piezas, valores de temperatura, niveles de los tanques, etc.… que se daba en la fábrica y/o industria, produciéndose errores en la lectura de los datos, ya que, el operador tenia que ver en un indicador y luego anotar en su papel el valor, pero dicho valor puede cambiar en un segundo drásticamente, en algunos casos, provocando así errores en los registros de supervisión, además también el cansancio que sufría el operador, propios del estrés del trabajo, etc., luego se implementaron equipos para realizar tareas de supervisión y control, principalmente, pero estos equipos utilizaban gran espacio (físico), finalmente en los tiempos actuales con el avance de los sistemas informáticos, se ha superado, en la mayoría de los casos, este acontecimiento, con la implementación de Softwares que proporcionan la suficiente capacidad de recoger los datos automáticamente y guardarlos en un registro ó en una base de datos para su posterior análisis, éste suceso a permitido que las fábricas y/o industrias agilicen de forma significativa su proceso productivo, aprovechando la gran capacidad que tienen las computadoras para el procesamiento de datos y la ventaja de que se le pueda programar para que haga tareas por nosotros valiéndonos del Hardware y Software existente.

En la actualidad el escenario industrial tiene la tendencia de expandirse hacia un ambiente totalmente automatizado, a la vez que las aplicaciones de monitoreo y control son más necesarias y complejas. Aparece entonces la necesidad de implementar Softwares potentes y amigables, para realizar aplicaciones económicas pero de gran utilidad para la industria, sobre todo en materia de control y gestión.

En los subcapítulos siguientes se van ha estudiar conceptos asociados a la parte de visualización industrial, que implica la interfase o la conexión que debe existir entre el hombre y la maquina, también al monitoreo, supervisión, control, adquisición de datos, que son tan necesarios en el campo industrial para optimizar los procesos, aumentar la productividad, y reducir los costos.

1.2 Interfaz Hombre – Máquina (HMI)

En un proceso industrial, cualquier proceso que se desee controlar y automatizar tiene que ser monitorizado. Existen ocasiones en las que se debe sensar temperatura, presión, posición, flujo velocidad, proximidad, etc., estas actividades deben ser realizadas con elementos de medición los cuales se les denomina sensores. El estado de los sensores constituyen las entradas para el elemento controlador, como puede ser un PLC, PAC, DAQ, PC, etc., estos controladores, según la entrada del sensor que reciban van a tomar alguna acción que tengan previamente programada, mediante los actuadores. Los actuadores son dispositivos de control que básicamente interactúan con el proceso a controlar, como lo son relevadores, arrancadores, electroválvulas, etc.

Hasta este punto se tiene un sistema automatizado basado en controladores, actuadores y sensores. Pero dicho sistema está aislado del elemento humano, claro que en la mayoría de casos esto es lo que se necesita, pues, el controlador es un dispositivo con capacidades superiores a las de un ser humano a la hora de realizar tareas de mucha precisión, repetitivas y de manera ininterrumpida, pero es el hombre quien toma decisiones en situaciones que implican mayor flexibilidad, simple sentido común, ó simplemente el proceso es tal que antes de iniciarlo los operadores deben introducir información al sistema, como puede ser el material o sustancias que se van a usar en una etapa del proceso, la cantidad de producto final que es necesario elaborar, cual de los varios contenedores disponibles van a utilizarse, o bien detener el proceso por alguna razón. Además los operadores del proceso necesitan interactuar con el sistema de control de una manera intuitiva, fácil de conocer, y que les proporcione la suficiente información del proceso en el momento que se necesite. Por esta razón se crean las llamadas **Interfaz Hombre – Máquina (HMI).**

Un HMI, por sus siglas en inglés: “*Human Machine Interface*”, son interfaces gráficas, muy simples, que muestran información del proceso en tiempo real,

utilizando diagramas esquemáticos, algunos contornos y hasta animaciones en pantallas o paneles; entonces se puede decir que una HMI es el aparato que presenta los datos a un operador (humano) y a través del cual éste controla el proceso.

En la **figura 1.1** se ilustra un ejemplo de interfase de usuario (HMI) realizada en Software LabVIEW.

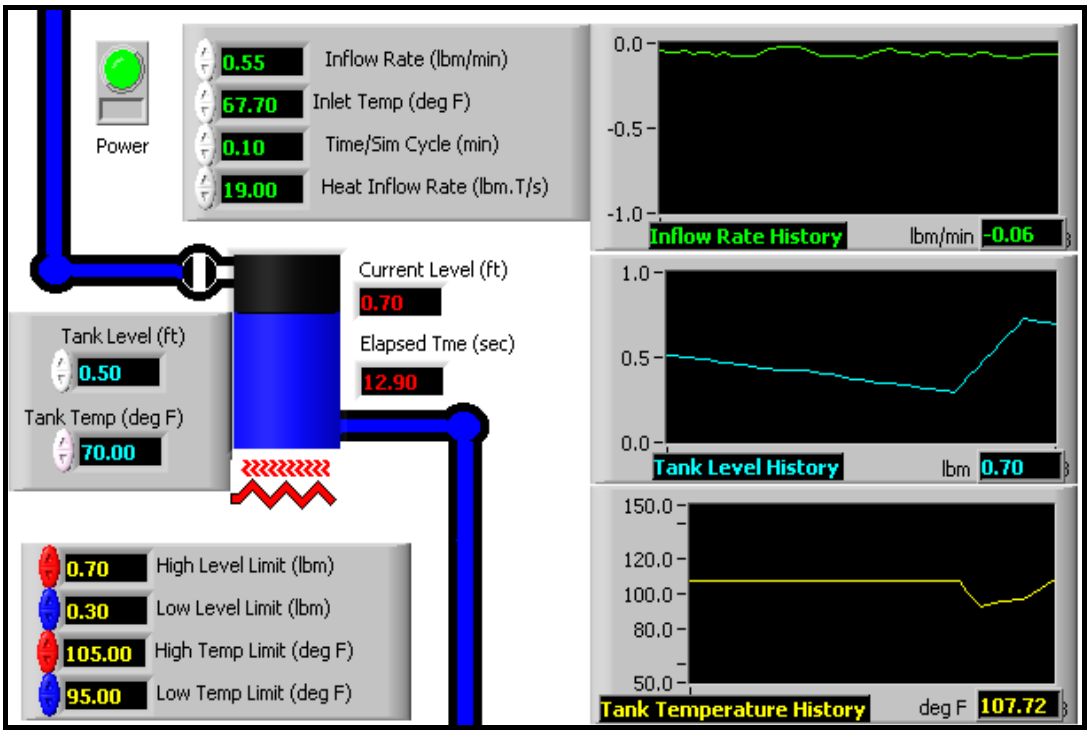


Figura 1.1 Interfaz HMI en LabVIEW.

(Tomado de los Ejemplos de LabVIEW)

Las interfaces básicas de usuario son aquellas que incluyen cosas como menús, ventanas, teclado, ratón y algunos otros sonidos que la computadora hace, en general, todos aquellos canales por los cuales se permite la comunicación entre el hombre y la computadora a través de una adecuada interfaz que le brinde tanto comodidad como eficiencia.

Se puede distinguir básicamente dos tipos de interfaces de usuario:

1.2.1 Según la forma de interactuar del usuario:

- **Interfaces Alfanuméricas**, son aquellas que solo presentan texto.

- **Interfaces gráficas de usuario**, son las que permiten comunicarse con el ordenador de una forma muy rápida e intuitiva representando gráficamente los elementos de control y medida.

- **Interfaces táctiles**, éstas representan gráficamente un "panel de control" en una pantalla sensible que permite interaccionar con el dedo de forma similar a si se accionara un control físico.

1.2.2 Según su construcción:

1.2.2.1 Interfaces hardware: Se trata de un conjunto de controles ó dispositivos que permiten la interacción hombre-máquina, de modo que permiten introducir o leer datos del equipo, mediante pulsadores, reguladores e instrumentos.

1.2.2.2 Interfaces software: Son programas o parte de ellos, que permiten expresar nuestros deseos al ordenador o visualizar su respuesta.

Las principales funciones de un interfaz de usuario son:

- Puesta en marcha y apagado.
- Control de las funciones manipulables del equipo.
- Manipulación de archivos y directorios.
- Herramientas de desarrollo de aplicaciones.
- Comunicación con otros sistemas.
- Información de estado.
- Configuración de la propia interfaz y entorno.
- Intercambio de datos entre aplicaciones.
- Control de acceso.
- Sistema de ayuda interactivo.

El diseño de una interfaz de usuario es crítico para el manejo de un equipo, hay algunas muy bien diseñadas que incorporan controles intuitivos y de fácil manejo,

en cambio existen otras que no se entienden bien y el usuario no acierta a manejarlas correctamente sin estudiar un manual ó recibir formación del experto.

El proceso general para diseñar la interfaz de usuario empieza con la creación de diferentes modelos de función del sistema:

- ✓ Se definen las tareas orientadas al hombre y a la maquina, requeridas para conseguir la función del sistema.

- ✓ Se consideran los aspectos de diseño aplicables a todos los diseños del sistema.

- ✓ Se consideran los aspectos del diseño aplicables a todos los diseños de interfaz.

- ✓ Se usan herramientas para crear el prototipo e implementar el modelo de diseño.

- ✓ Se evalúa la calidad del resultado.

Algunas de estas interfaces necesitan, además de informativas de manera gráfica y de tiempo real, tener la capacidad de visualizar Información Histórica, es decir, datos del proceso correspondientes a hace algunas horas o días, además, las interfaces deben ser capaces de interactuar con el sistema de control para iniciarlo, detenerlo, cambiar valores base, set-points, monitorear la disponibilidad y existencia de materia prima, la selección de las llamadas Recetas, que son la relación de sustancias componentes de una mezcla o compuesto, así como la secuencia de sub-procesos y los tiempos correspondientes a cada uno de ellos y quizás la más importante, la adquisición de datos para su posterior procesamiento y obtención de elementos propios para la toma de decisiones a nivel de planta; este tipo de sistemas recibe el nombre de Control Supervisorio ó SCADA.

1.3 Sistemas SCADA.

SCADA por sus siglas en inglés: "***Supervisory Control And Data Acquisition***", es decir: "**Control con Supervisión y Adquisición de Datos**". Se trata de una

aplicación formado por diferentes Softwares ó programas diseñada para funcionar sobre ordenadores en el control de producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos, autómatas programables, Tarjetas de adquisición de datos, etc.) y controlando el proceso de forma automática desde la pantalla del computador. Además, provee de toda la información que se genera en el proceso productivo a diversos usuarios, tanto del mismo nivel como de otros supervisores dentro de la empresa: control de calidad, supervisión, mantenimiento, etc.

En este tipo de sistemas usualmente existe un computador, que efectúa tareas de supervisión y gestión de alarmas, así como tratamiento de datos y control de procesos. Un sistema SCADA incluye un hardware de señal de entrada y salida, controladores, interfaz hombre-máquina, comunicaciones, redes, base de datos y software. En la **Figura 1.2** se ilustra un ejemplo del concepto SCADA, anteriormente mencionado.

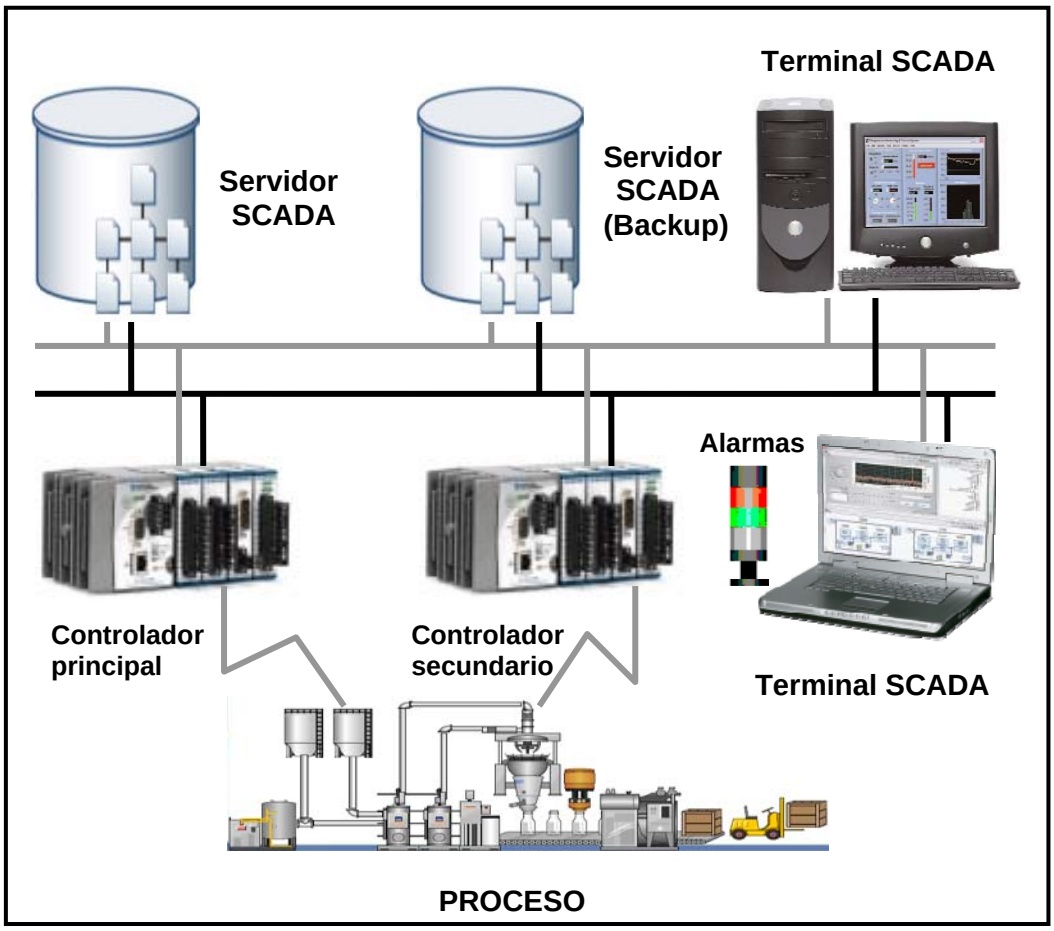


Figura 1.2. Concepto del Sistema SCADA. Arquitectura típica de un sistema HMI/SCADA.

La mayor parte del control del sitio es en realidad realizada automáticamente por una
Unidad Terminal Remota (RTU) o por un Controlador Lógico Programable (PLC), tarjetas de adquisición de datos (DAQs), Controlador Automático Programables (PAC), o por PCs.

Las funciones de control del servidor están casi siempre restringidas a reajustes básicos del sitio o capacidades de nivel de supervisión. Por ejemplo un PLC puede controlar el flujo de agua fría a través de un proceso, pero un sistema SCADA puede permitirle a un operador cambiar el punto de consigna (set point) de control para el flujo, y permitirá grabar y mostrar cualquier condición de alarma como la pérdida de un flujo o una alta temperatura, etc. La realimentación del lazo de control es cerrada a través del RTU o el PLC, PC, etc.; y lo que hace el sistema SCADA es monitorear el desempeño general de dicho lazo.

1.3.1 Componentes de un Sistema SCADA.

Los componentes principales de un sistema SCADA son tres:

- ✓ Múltiples Unidades de Terminal Remota (también conocida como RTU o Estaciones Externas).
- ✓ Estación Maestra y Computador con HMI.
- ✓ Infraestructura de Comunicación.

En la **figura 1.3** se muestra los componentes de un sistema SCADA.

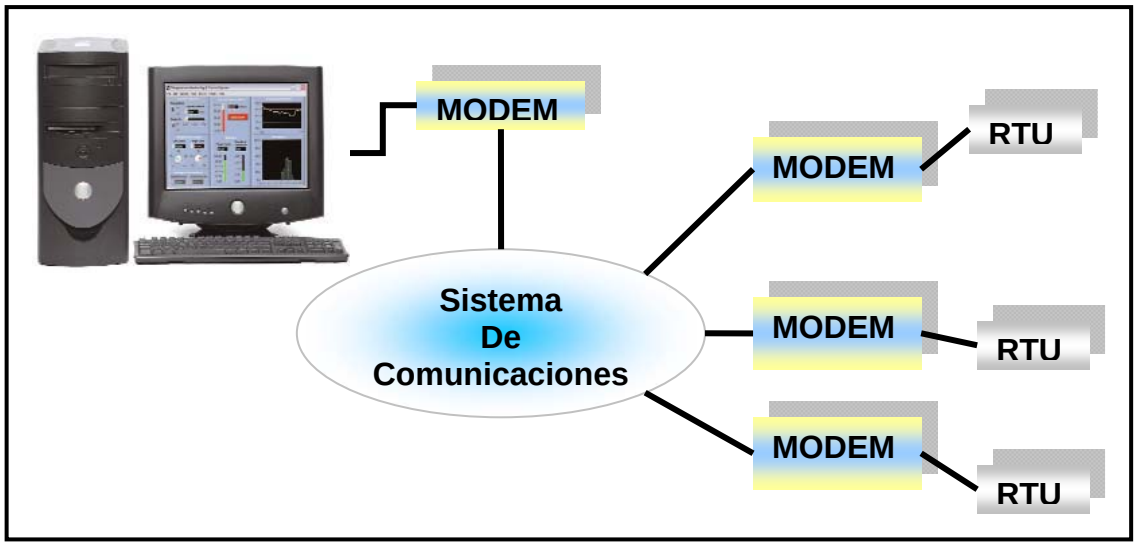


Figura 1.3 Componentes de un sistema SCADA.

1.3.1.1 Unidad de Terminal Remota (RTU)

La RTU se conecta al equipo físicamente y lee los datos de estado como los cambios abierto/cerrado desde una válvula o un intercambiador, lee las medidas como presión, flujo, voltaje o corriente. Por el equipo el RTU puede enviar señales que pueden controlarlo: abrirlo, cerrarlo, intercambiar la válvula o configurar la velocidad de la bomba, etc. La RTU puede leer el estado de los datos digitales o medidas de datos analógicos y envía comandos digitales de salida o puntos de ajuste analógicos.

Una de las partes más importantes de la implementación de un sistema HMI/SCADA son las alarmas.

Conceptualmente una alarma es un punto de estado digital, un aviso que se ha producido una desviación de un parámetro normal, por ende los sistemas de monitoreo desarrollan un esquema para facilitar el manejo de las mismas.

Se definen 3 estados en los cuales se puede encontrar una alarma cuyos niveles dependerá del tipo de variable a establecer (analógicas o digitales):

- Sin alarma.
- Alarma nueva sin reconocer (unack alarm).
- Existente y reconocida (ack alarm).

En las alarmas analógicas se distinguen 4 niveles, (**figura 1.4**):

- Muy Bajo: Cuando se encuentra muy por debajo de un valor de referencia LoLo.
- Bajo: cuando se encuentra por debajo de un valor de referencia Lo.
- Alto: cuando se encuentra por arriba de un valor de referencia Hi.
- Muy Alto: cuando se encuentra muy arriba de un valor de referencia HiHi.

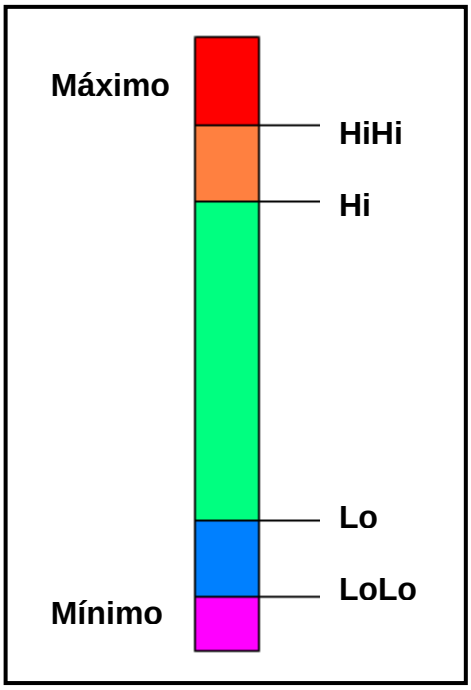


Figura 1.4 Niveles de Alarma Analógica

En algunos caso las alarmas pueden asignarse no solo al cambio de un valor sino a la velocidad de cambio de una variable, a este tipo de alarmas se las denomina “ROC alarms” (Rate of Change Alarm).

La alarma se puede crear en cada paso que los requerimientos lo necesiten, y el operador del sistema HMI/SCADA pone atención a la parte del sistema que lo requiera, por la alarma, y según esto, pueden enviarse por correo electrónico o mensajes de texto con la activación de una alarma, alertando al administrador o incluso al operador de SCADA, que es lo que esta pasando en la fábrica.

1.3.1.2 Estación Maestra

El término "Estación Maestra" se refiere a los servidores y el software responsable para comunicarse con el equipo de campo (RTUs, PLCs, PACs etc.). En la estación maestra se encuentra el software HMI corriendo para las estaciones de trabajo en el cuarto de control o en cualquier otro lado, la estación maestra puede estar en un solo computador (Sistema SCADA pequeño) o puede incluir muchos servidores, aplicaciones de software distribuido, y sitios de recuperación de desastres (Sistema SCADA a gran escala).

Como se había mencionado al principio de éste capitulo, el Sistema HMI/SCADA usualmente presenta la información al personal operativo gráficamente, esto significa que el operador puede ver un esquema que representa la planta que está siendo controlada, de forma que pueda ser intuitiva para el operario.

Los diagramas de representación pueden consistir en gráficos de líneas y símbolos esquemáticos para representar los elementos del proceso, o pueden consistir en fotografías digitales de los equipos sobre los cuales se animan las secuencias. El paquete HMI para el sistema SCADA incluye un programa de dibujo con el cual los operadores usan para cambiar la manera que estos puntos son representados en la interfaz. En la **figura 1.5** se ilustra el concepto de una estación maestra.

Los sistemas HMI/SCADA poseen varios componentes que permiten algunos tipos de funciones; además de visualizar los datos, establecer alarmas, visualizar tendencias de las variables medidas, comunicarse con los dispositivos de campos, también permiten generar datos históricos, etc. En la **figura 1.6** puede observarse algunos de los principales módulos de un software SCADA.

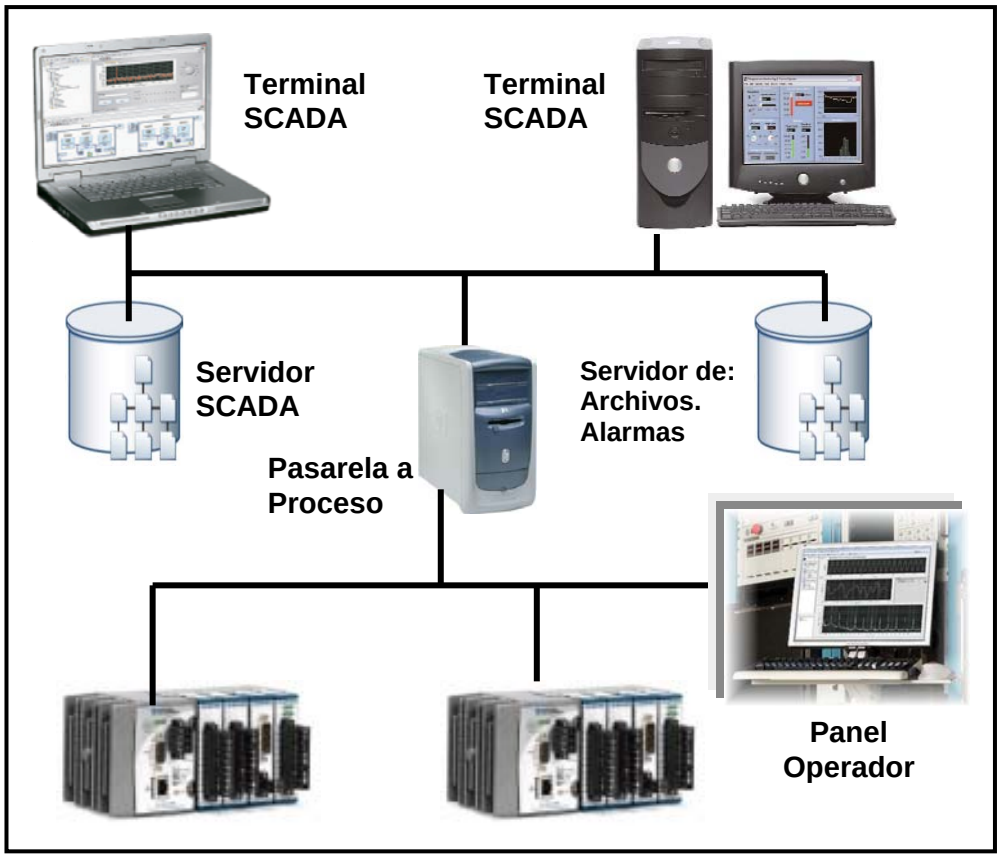


Figura 1.5 Ejemplo de una estación maestra.

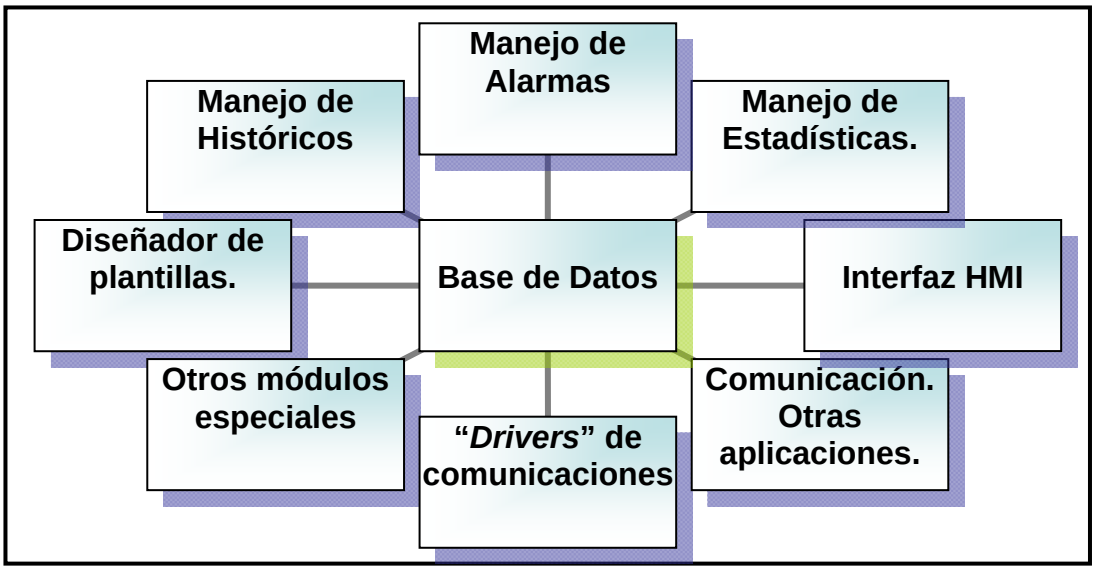


Figura 1.6 Principales módulos de un Sistema Scada

1.3.1.2.1 Módulos de un sistema SCADA

Los módulos o bloques software que permiten las actividades de adquisición, supervisión y control son los siguientes:

- **Configuración**: permite al usuario definir el entorno de trabajo de su SCADA, adaptándolo a la aplicación particular que se desea desarrollar.
- **Interfaz gráfico del operador**: proporciona al operador las funciones de control y supervisión de la planta. El proceso se representa mediante sinópticos gráficos almacenados en el ordenador de proceso y generados desde el editor incorporado en el SCADA o importados desde otra aplicación durante la configuración del paquete.
- **Módulo de proceso**: ejecuta las acciones de mando preprogramadas a partir de los valores actuales de variables leídas.
- **Gestión y archivo de datos**: se encarga del almacenamiento y procesado ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos.
- **Comunicaciones**: se encarga de la transferencia de información entre la planta y la arquitectura hardware que soporta el SCADA, y entre ésta y el resto de elementos informáticos de gestión.

1.3.1.2.2 Filosofía Operacional.

En vez de confiar en la intervención del operador o en la automatización de la estación maestra los RTUs pueden ahora ser requeridos para operar ellos mismos, realizando su propio control sobre todo por temas de seguridad. El software de la estación maestra requiere hacer más análisis de datos antes de ser presentados a los operadores, incluyendo análisis históricos y análisis asociados con los requerimientos de la industria particular. Los requerimientos de seguridad están siendo aplicados en los sistemas como un todo e incluso el software de la estación maestra debe implementar los estándares más fuertes de seguridad en ciertos mercados.

Para algunas instalaciones, los costos que pueden derivar de los fallos de un sistema de control son extremadamente altos, es posible incluso que haya riesgo de herir a las personas. El Hardware del sistema SCADA es generalmente lo suficientemente robusto para resistir condiciones de temperatura, humedad,

vibración y voltajes extremos pero en estas instalaciones es común aumentar la fiabilidad mediante hardware redundante y varios canales de comunicación. Una parte que falla puede ser fácilmente identificada y su funcionalidad puede ser automáticamente desarrollada por un hardware de backup. Una parte que falle puede ser reemplazada sin interrumpir el proceso. La confianza en cada sistema puede ser calculado estadísticamente y este estado es el significado de tiempo medio entre fallos, el cual es una variable que acumula tiempos entre fallas. El resultado calculado significa que el tiempo medio entre fallos de sistemas de alta fiabilidad puede ser de siglos.

1.3.1.3 Infraestructura y Métodos de Comunicación.

Los sistemas SCADA tienen tradicionalmente una combinación de radios y señales directas seriales o conexiones de módem para conocer los requerimientos de comunicaciones, incluso Ethernet e IP sobre SONET (fibra óptica) es también frecuentemente usada en sitios muy grandes como ferrocarriles y estaciones de energía eléctrica. Es más, los métodos de conexión entre sistemas pueden incluso que sea a través de Wireless (por ejemplo si queremos enviar la señal a una PDA, a un teléfono móvil,...) y así no tener que emplear cables. Para que la instalación de un SCADA sea perfectamente aprovechada, debe de cumplir varios objetivos:

- Deben ser sistemas de arquitectura abierta (capaces de adaptarse según las necesidades de la empresa).
- Deben comunicarse con facilidad al usuario con el equipo de planta y resto de la empresa (redes locales y de gestión).
- Deben ser programas sencillos de instalar, sin excesivas exigencias de hardware. También tienen que ser de utilización fácil

1.3.1.3.1 Transmisión de datos

Los escenarios tradicionales en un Sistema SCADA a nivel de transmisión de datos son básicamente tres:

- ✓ A través de una interfaz serial RS-232 y distintos protocolos asociados.
- ✓ A través de una red con protocolo propietario.
- ✓ A través de una red con protocolo TPC/IP

1.3.1.3.1.1 Interfaz Serial RS-232

La comunicación por RS-232 ha sido la forma más común de comunicación de un Sistema SCADA con un PLC; las primeras implantaciones generaban mensajes bajo un código numérico para solicitar que el PLC ejecutará una acción o proporcionará información, posteriormente se cuentan con implantaciones en las que su programación se realizaba con lenguajes de tercera generación (BASIC, C, etc.); esta forma de comunicación se constituye en una conexión primitiva punto a punto, sin flexibilidad de comunicarse con otros equipos.

1.3.1.3.1.2 Comunicación en red con protocolo propietario

Dada la necesidad de conexiones multipunto entre equipos de cómputo soportando Sistemas SCADA y PLCs los fabricantes se vieron en la necesidad de crear arquitecturas y topologías de redes para conectarlos, de esta forma una aplicación SCADA se puede comunicar con más de un PLC en red; y un PLC con más de una aplicación SCADA.

Algunos de estos protocolos se convirtieron en estándares de facto; tal ha sido el caso de PROFIBUS de Siemens, muchos fabricantes de PLCs ofrecen compatibilidad para este protocolo, sin embargo, tal esquema de propiedad implica un alto costo.

1.3.1.3.1.2.1 Profibus

Es un bus de alto nivel que ya está normalizado y completamente integrado en Europa y en todo el mundo, está basado en el modelo OSI de ISO, del cual utiliza solo el nivel físico, el nivel de enlace y el nivel de aplicación; destinado a aplicaciones de fabricación discreta; el funcionamiento de profibus es del tipo maestro/esclavo, donde el bus puede poseer distintos maestros, los maestros pueden ocupar el bus en un tiempo y hora determinados, lo que determina la ocupación del bus es el permiso denominado Token que circula entre los maestros del bus siguiendo una orientación lógica determinada por las direcciones lógicas de las estaciones maestras; la combinación maestro/esclavo y el acceso por Token establecen un acceso al medio llamado híbrido, el atributo más destacado del profibus es la flexibilidad.

1.3.1.3.1.2.2 TCP/IP

La creciente demanda de la utilización de estándares abiertos de comunicación, llevó al diseño basado en el protocolo TCP/IP. La utilización de TCP/IP soportando aplicaciones SCADA y PLCs amplió una gama de posibilidades hacia otro tipo de equipos y esquemas tipo IPC (InterProcess Communication) tales como sockets, RPCs (Remote Procedure Calls) y Middleware (esquemas de comunicación entre objetos) con la capacidad de generar aplicaciones distribuidas.

La utilización de TCP/IP, y en general, la adopción del modelo de referencia ISO/OSI, ha implicado considerar una mayor importancia del papel que desempeñan los sistemas operativos que soportan estos protocolos y esquemas de ventaja asociados, siendo importante no olvidar las características de tiempo real que deben soportar éstos. Los sistemas operativos que se han incorporando a los ambientes productivos son: QNx, Linux Real Time y Microsoft Windows NT.

Los Sistemas Operativos incorporan el middleware, en el que básicamente existen dos estándares:

- DCOM (Distributed Component Object Model, de Microsoft).
- CORBA (Common Object Request Broker Architecture).

Esto permite que objetos creados bajo el mismo estándar se haya facilitado la creación de aplicaciones SCADA que se comunican con otros componentes y con aplicaciones generales.

1.3.1.3.2 El Estándar OPC

En 1994, un grupo de Proveedores de un importante segmento industrial le dio forma a la fundación OPC, la meta era desarrollar una especificación de cliente/servidor que permita a cualquier proveedor compartir datos de una manera rápida y robusta tendiendo a eliminar la necesidad de desarrollar un sistema de drivers de comunicaciones en aplicaciones cliente.

Los componentes OPC se pueden clasificar en clientes o servidores:

Servidor OPC *(OPC Server*): Es una aplicación que realiza la recopilación de datos de los diversos elementos de campo de un sistema automatizado y permite el acceso libre a estos elementos desde otras aplicaciones que los soliciten (clientes OPC).

Para muchos proveedores el esfuerzo requerido para desarrollar drivers de comunicaciones sobrepasa en esfuerzo de desarrollar las aplicaciones completas; con la adopción de la tecnología OPC un proveedor puede ahora centrar sus esfuerzos casi exclusivamente en el desarrollo de aplicaciones cliente.

En la **Figura 1.7** se ilustra como un cliente OPC se puede conectar a servidores OPC proporcionados por más de un vendedor.

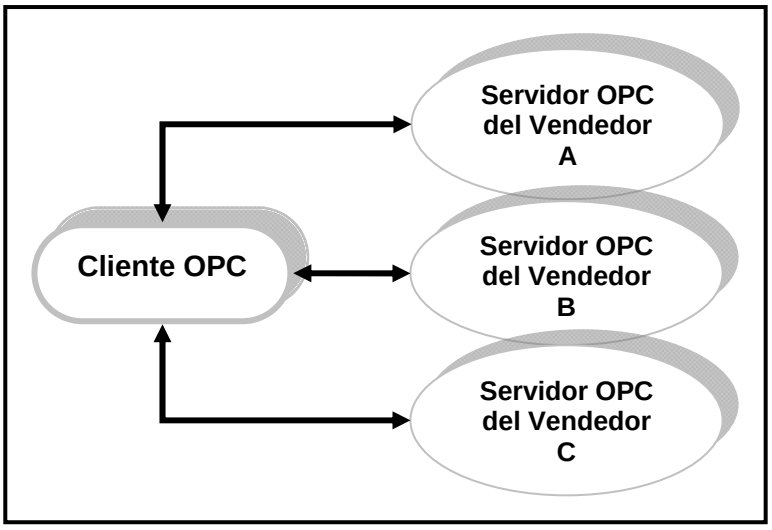


Figura 1.7. El Estándar OPC (cliente-servidor)

OPC puede utilizarse en varios subsistemas, consolas de gestión, aplicaciones de monitoreo y dispositivos de entrada/salida con sus respectivos drivers.

1.3.2 Niveles de un Sistema Scada

Un sistema SCADA tiene 4 niveles principales:

- ✓ Gestión Intercambio de información para la toma de decisión estratégica.
- ✓ Operación Supervisión, mando y adquisición de datos del proceso.
- ✓ Control Dispositivos de Control Distribuido
- ✓ Sensores y Actuadores Dispositivos de campo e instrumentación.

En la **figura 1.8** se distinguen los diferentes niveles de un Sistema SCADA dentro de cada cual según su jerarquía se aprecian sus distintos instrumentos de medición, dispositivos de campo, alarmas, etc.

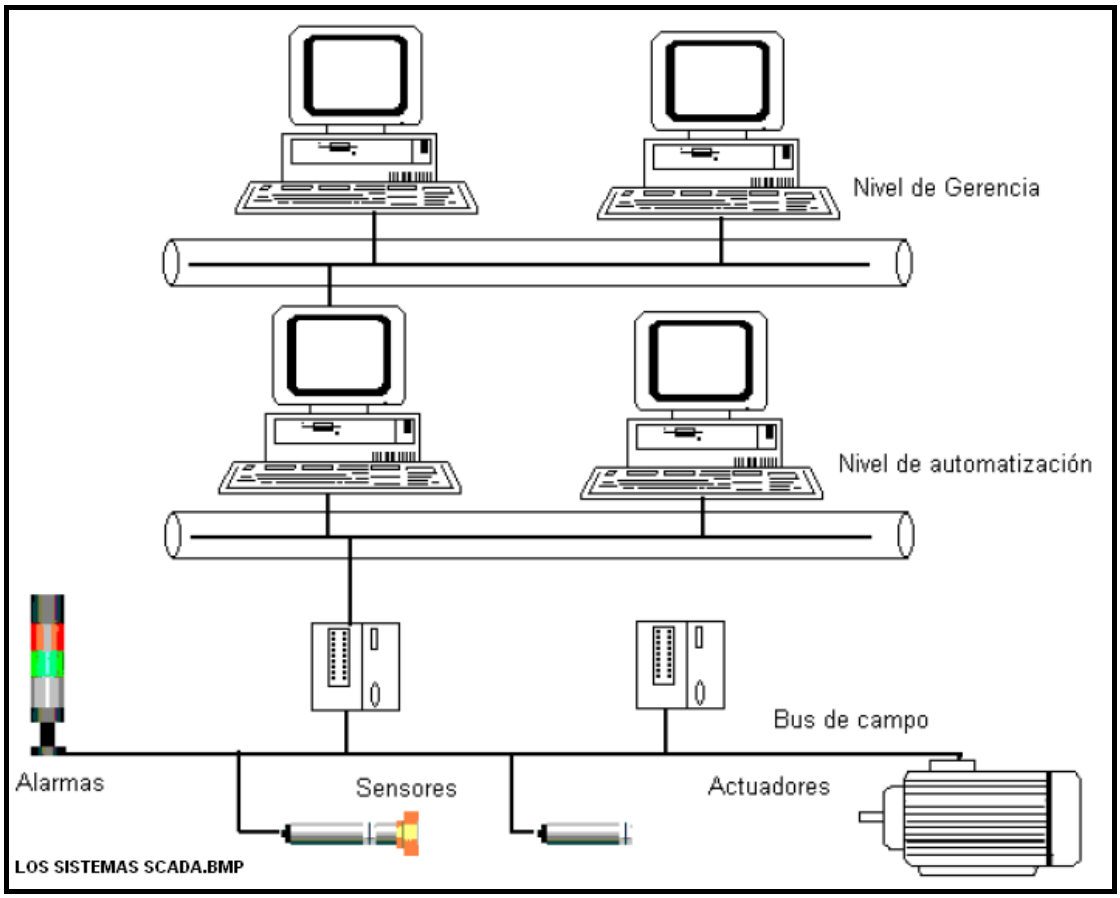


Figura 1.8 Niveles de un Sistema SCADA.
(HERRERA, Lecciones sobre Sistemas SCADA, 2008, Pág. 11.)

La comunicación se realiza mediante buses especiales o redes LAN, la ejecución normalmente es en tiempo real y están diseñados para dar al operador de planta la posibilidad de supervisar y controlar varios procesos. Los SCADA actuales pueden almacenar datos en una base de datos local o remota. Generan reportes de producción diarios, mantienen un sistema de alarma que se puede propagar a través de la red LAN o la Intranet, en Algunos casos muy sofisticados, pueden comunicarse directamente a través de Internet.

1.3.3 Características de un sistema HMI/SCADA

Las características principales de un sistema HMI/SCADA se enumeran a continuación.

- ✓ Alto número de canales (y diversidad de dispositivos).
- ✓ Registro en base de datos.
- ✓ Gestión de Alarmas y Eventos.
- ✓ Seguridad.
- ✓ Trabajo en red.

1.3.4 Prestaciones de un paquete SCADA.

Un paquete SCADA debe estar en disposición de ofrecer las siguientes prestaciones:

- Posibilidad de crear paneles de alarma, que exigen la presencia del operador para reconocer una parada o situación de alarma, con registro de incidencias.
- Generación de históricos de señal de planta, que pueden ser volcados para su proceso sobre una hoja de cálculo.
- Ejecución de programas, que modifican la ley de control, o incluso anular o modificar las tareas asociadas al autómata, bajo ciertas condiciones.
- Posibilidad de programación numérica, que permite realizar cálculos aritméticos de elevada resolución sobre la CPU del ordenador.

Con ellas, se pueden desarrollar aplicaciones para ordenadores (tipo PC, por ejemplo) con captura de datos, análisis de señales, presentaciones en pantalla, envío de resultados a disco e impresora, etc.

Además, todas estas acciones se llevan a cabo mediante un paquete de funciones que incluye zonas de programación en un lenguaje de uso general (como LabVIEW, C, Pascal, o Basic), lo cual confiere una potencia muy elevada y una gran versatilidad. Algunos SCADA ofrecen librerías de funciones para lenguajes de uso general que permiten personalizar de manera muy amplia la aplicación que desee realizarse con dicho SCADA.

1.3.5 Soluciones de Hardware

La solución de hardware de un Sistema SCADA frecuentemente posee componentes de Sistemas de Control Distribuido (DCS). El uso de RTUs o PLCs y últimamente PACs sin involucrar computadoras maestras está aumentando, los cuales son autónomos ejecutando procesos de lógica simple. A menudo se usa un lenguaje de programación funcional para crear software's que corran en estos RTUs y PLCs; la complejidad y la naturaleza de este tipo de programación hace que los programadores necesiten cierta especialización y conocimiento sobre los actuadores que van a programar. Aunque la programación de estos elementos es ligeramente distinta a la programación tradicional, también se usan lenguajes que establecen procedimientos, como pueden ser Fortran o C que les permite a los ingenieros de sistemas SCADA implementar programas para ser ejecutados en RTUs o en PLCs. Por su parte, National Instruments Corporation, mediante el software LabVIEW permiten implementar SCADAs y la programación de PACs y PLCs para lograr controlar y supervisar los procesos, en un ambiente de programación gráfico.

1.3.5.1 Sistemas de Control Distribuido, DCS (Distributed Control System)

Un sistema de control distribuido se puede construir según dos enfoques:

a) Desarrollar únicamente los protocolos de comunicación entre los distintos dispositivos; este enfoque no ofrece una visión única y coherente del sistema, o sea las características de cada dispositivo son visibles y deben tomarse en cuenta al implementar las aplicaciones.
b) Consiste en usar una capa adicional de software sobre los protocolos de comunicaciones; esta capa provee a las aplicaciones una visión uniforme y coherente de los elementos heterogéneos subyacentes (máquinas, sistemas operativos, sistemas de comunicación, etc.)

Las principales ventajas de este tipo de software son:

* Interoperabilidad.
* Transparencia.
* Confiabilidad.
* Disponibilidad

- Escalabilidad
- Abstracción.

En la **tabla 1.1** se muestra un cuadro comparativo de las principales características de los sistemas SCADA y los sistemas de Control Distribuido (DCS).

ASPECTO	SCADA	DCS
TIPO DE ARQUITECTURA	CENTRALIZADA	DISTRIBUIDA
TIPO DE CONTROL PREDOMINANTE	SUPERVISORIO: Lazos de control cerrados por el operador. Adicionalmente: control secuencial y regulatorio.	REGULATORIO: Lazos de control cerrados automáticamente por el sistema. Adicionalmente: control secuencial, batch, algoritmos avanzados, etc.
TIPO DE VARIABLES	DESACOPLADAS	ACOPLADAS
AREA DE ACCION	Áreas geográficamente distribuidas.	Área de la planta
UNIDADES DE ADQUISION DE DATOS Y CONTROL	Remotas, PLCs.	Controladores de lazo, PLCs.
MEDIOS DE COMUNICACION	Radio, satélite, líneas telefónicas, conexión directa, LAN, WAN	Redes de área local, conexión directa
BASE DE DATOS	CENTRALIZADA	DISTRIBUIDA

Tabla 1.1 Algunas diferencias típicas entre sistemas SCADA y DCS.

(HERRERA, Lecciones sobre Sistemas SCADA, 2008, Pág. 33.)

1.3.6 Paradigma Cliente/Servidor

El Paradigma Cliente Servidor se constituye en la práctica como una filosofía de diseño de soluciones, empíricamente suele asociarse el concepto de servidor a un equipo de cómputo corporativo que ofrece una serie de servicios a equipos cliente conectados en red; sin embargo, la concepción de la arquitectura cliente/servidor va más allá de esta percepción.

En términos conceptuales, el cliente es aquella entidad en la que se formula un requerimiento y valida los datos indispensables para solicitarlo al servidor. El servidor es una entidad que recibe requerimientos por parte del cliente, los procesa, genera los resultados y los envía al cliente. El cliente recibe los resultados del servidor y los utiliza para mostrarlos al usuario para que éste disponga de ellos.

El usuario puede ser una persona u otra aplicación; cuando el Paradigma Cliente Servidor describe aspectos estructurales, suele expresarse como Arquitectura Cliente/Servidor (ACS), de tal forma que la ACS aplicado a redes (computadora servidor y computadoras cliente) es un caso particular de esta situación. La ACS también se puede aplicar a procesos (proceso servidor, proceso cliente), a objetos, a manejadores de bases de datos, a sistemas SCADA.

1.3.6.1 Arquitectura Cliente/Servidor en Sistemas SCADA

Es una arquitectura basada en capas; estas capas proporcionan los servicios de datos, negocio (algoritmos centrales) y presentación al usuario (persona o aplicación). Los elementos principales que interactúan con estas aplicaciones SCADA son equipos de control automático, generalmente PLCs y aplicaciones en general, las cuales suelen ser de tipo científico o administrativos con acceso a bases de datos.

1.3.7 Softwares SCADA

En la actualidad existen muchísimos Softwares que permiten realizar aplicaciones SCADA, el uso de uno u otro software para realizar las diferentes aplicaciones de SCADA dependen en gran medida del fabricante del PLC, PAC, DAQ, etc., ya que cada fabricante tiene su propio protocolo de comunicación, claro que esto no es un imposible para establecer una comunicación ya que se puede usar la arquitectura OPC (*Ole for Process Control*), como ya se había visto en un subtitulo anterior.

Algunos de los Softwares más importantes y usados dentro del entorno industrial por sus prestaciones, facilidad de uso, etc., se mencionan a continuación:

- ✓ **PROVEEDOR**: USDATA (http://www.usdata.com/)
  - **Producto**: Factory Link 7.

- ✓ **PROVEEDOR**: Advantech (http://www.advantech.com/)
  - **Producto**: Paradym-31
- ✓ **PROVEEDOR**: AlterSys Inc.
  - **Producto**: Virgo 2000.
- ✓ **PROVEEDOR**: eMation (http://www.emation.com/)
  - **Producto**: WizFactory.
- ✓ **PROVEEDOR**: GE Fanuc (http://www.gefanuc.com/)
  - **Producto**: Cimplicity.
- ✓ **PROVEEDOR**: Iconics (http://www.iconics.com/)
  - **Producto**: Genesis32.
- ✓ **PROVEEDOR**: Intellution (http://www.intellution.com/)
  - **Producto**: Intellution Dynamics.
- ✓ **PROVEEDOR**: National Instruments (http://www.ni.com/)
  - **Producto**: LabVIEW.
  - **Producto**: Lookout
- ✓ **PROVEEDOR**: Nematron (www.nematron.com/)
  - **Producto**: HMI/SCADA Paragon
- ✓ **PROVEEDOR**: Opto 22 (http://www.opto22.com/)
  - **Producto**: FactoryFloor Software
- ✓ **PROVEEDOR**: Rockwell Automation (http://www.software.rockwell.com/)
  - **Producto**: RSView32
  - **Producto**: RSBatch
- ✓ **PROVEEDOR**: Siemens
  - **Producto**: HYBREX (Hybrid Expert System)
  - **Producto:** WinCC HMI
  - **Producto:** Web Control Center (webCC)
  - **Producto:** SIMATIC WinAC ODK (Open Developer Kit)
  - **Producto:** SIMATIC WinAC (Windows Automation Center)
  - **Producto:** SIMATIC PLCSim
  - **Producto:** SIMATIC Protool
- ✓ **PROVEEDOR:** TA-Engineering Products (www.ta-eng.com/home.htm)
  - **Producto:** Aimax
- ✓ **PROVEEDOR:** Wonderware (http://www.wonderware.com/)

- **Producto:** FactorySuite 2000
  - **Intouch**.
  - **IndustrialSQL Server**

✓ **Etc.…**

Como se puede observar existe una gran cantidad de Softwares dedicados a realizar tareas SCADAs, de ahí la decisión de saber que software utilizar, que en gran medida debe estar ajustado a nuestras condiciones y exigencias, y por supuesto de acuerdo al proyecto que se esté realizando, también dependerá de la facilidad de manejo de dicho software, es decir la programación del SCADA, y obviamente del precio o coste del software SCADA. Generalmente los Softwares SCADA son amigables, de hecho esa es una característica de los sistemas SCADA.

En esta presente tesis, en particular, se decidió trabajar con el software LabVIEW de National Instruments Corporation, porque es un software de programación gráfico liberando a los programadores de la rigidez de las arquitecturas basadas en texto. Al ser un lenguaje de programación gráfico, no solamente va ha servir para realizar tareas de SCADA, sino también implementar programas adicionales, como una transformada rápida de Fourier, integrales, derivadas, etc.… según sea el caso, es decir herramientas matemáticas, facilitando el monitoreo y/o control del proceso, además de otras herramientas adicionales que tiene LabVIEW. Además LabVIEW tiene un gran número de aplicaciones tanto dentro de la industria, la medicina, la manufactura, automatización (robótica y visión artificial), adquisición y generación de señales, etc. Aprovechando esta gran variedad de recursos que tiene LabVIEW y todo dentro de un mismo ambiente de programación, se hizo uso para realizar tareas de visión artificial basada en PC. Para más información, en el **capítulo 2** se hace una introducción al entorno de programación en LabVIEW y a la Visión artificial basada en PC con LabVIEW.

1.3.7.1 SCADA en LabVIEW

LabVIEW de National Instruments es la plataforma de software líder en la industria para sistemas de Control, Pruebas y Diseño. Gracias a este software ingenieros y técnicos pueden aumentar su productividad y reducir sus costos, confiando en el desarrollo gráfico de LabVIEW para desarrollar sus proyectos a lo largo de todo el

ciclo de creación de un producto, obteniendo así una mejor calidad y una mayor eficacia en la ingeniería y manufactura.

En el pasado, los ingenieros que desarrollaban sistemas industriales tuvieron que aprender instrumentos de software diferentes para programar a los controladores y las aplicaciones HMI/SCADA comprando siempre todo del mismo vendedor, para evitarse los problemas de incompatibilidad, etc... Hoy en día, con el software de programación gráfica de LabVIEW de National Instruments Corporation, se puede programar tanto la Interfase Hombre Maquina (HMI), y en el caso de que se tengan los PACs (Controlador Automático Programable) de National Instruments, programarlo en el mismo ambiente.

El término SCADA en LabVIEW es equivalente a **DSC**, por sus siglas en inglés "***Datalogging and Supervisory Control***", es decir: Control Supervisorio y Almacenamiento de Datos, no confundir con el término DCS (Sistemas de Control Distribuido). Entonces, en LabVIEW para realizar o crear aplicaciones SCADA se debe adquirir el módulo "***NI LabVIEW 8.6 Datalogging and Supervisory Control Module***", (disponible en modo de evaluación en: www.ni.com/downloads).

Con el Módulo LabVIEW DSC se puede desarrollar sistemas HMI/SCADA flexibles, a través de una herramienta de fácil uso, el programador puede simplemente diseñar la interfase humano-máquina (HMI) de su preferencia (colocando los objetos gráficos en el panel frontal de LabVIEW), configurar las señales de E/S, etc. Las aplicaciones de hoy requieren HMI/SCADAs abiertos y que tengan conectividad OPC (Ole for Process Control) y Modbus y la flexibilidad de un lenguaje de programación, entonces con todo esto se puede añadir medidas rápidas y análisis a sistemas existentes. El módulo de LabVIEW DSC proporciona configuraciones basadas en registros de datos, alarmas, y seguridad desarrollados para los sistemas HMI/SCADAs basados en Windows XP, por ejemplo, se puede usar el módulo de LabVIEW DSC para crear al instante miles de etiquetas, unirlos a servidores de entrada – salida y redes de targets de LabVIEW de Tiempo real y dispositivos OPC.

Según National Instruments (NI), en LabVIEW 8, el Módulo LabVIEW DSC fue totalmente integrado en el ambiente LabVIEW, ofreciendo un realce de interpretación significativo y mejorando la facilidad del uso. En LabVIEW 8.20 o superior, se puede usar el Módulo LabVIEW DSC para leer y escribir de la base de

datos en tiempo real incorporada en 11000 actualizaciones por segundo para 20000 etiquetas, mejora de interpretación en 2 veces del registro de datos. También se puede escalar una aplicación en el tiempo de ejecución usando nuevos instrumentos de programación, como la creación programática de un grupo de variables compartidas, en las cuales se puede encadenar a servidores de entrada - salida.

Esta plataforma de desarrollo integrada permite que los ingenieros construyan rápidamente sistemas de control y los modifiquen fácilmente a medida que cambian los requerimientos del sistema.

1.3.7.1.1 Características para HMI/SCADA de LabVIEW DSC

En la **figura 1.9** se muestra algunas de las principales características del módulo DSC de LabVIEW. Entre las principales características de dicho módulo se tienen:

- Soporte OPC Client.
  - Conectarse a cualquier PLC.
- Modbus I/O Server.
  - Conexión a nuevo hardware Industrial
- I/O Server Personalizado
  - Cualquier VI en servicio
- Gráficos Industriales.
  - Desarrollo de Paneles.

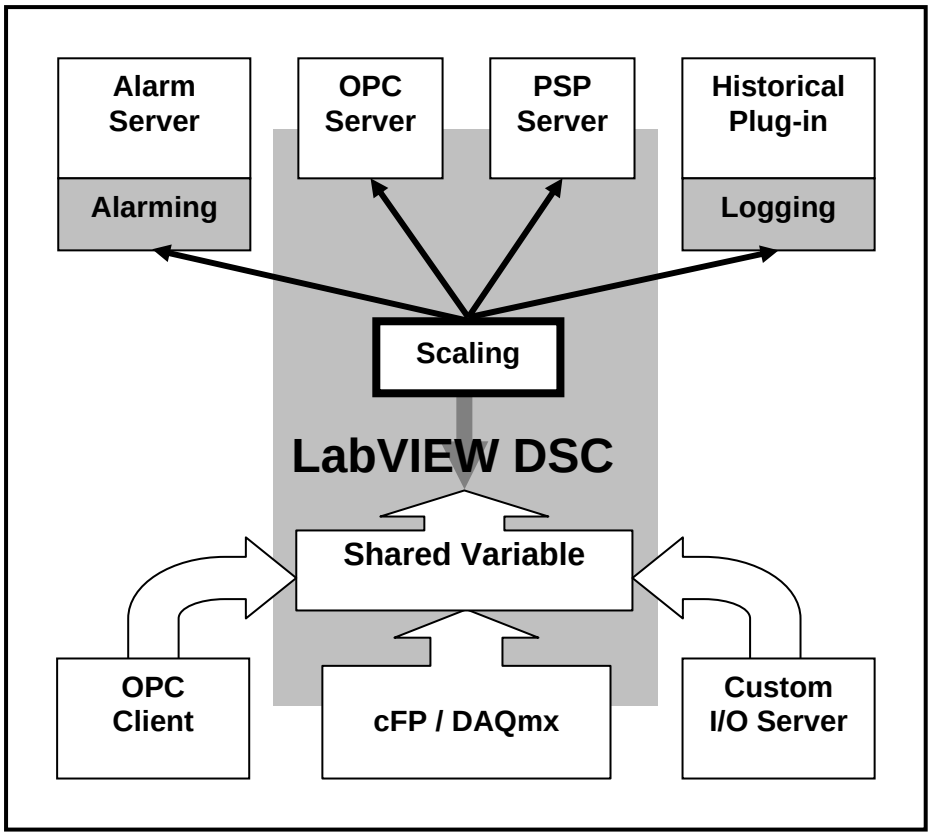


Figura 1.9 Algunas características del LabVIEW DSC.

El desarrollo de un sistema de medición y control dentro de un entorno de desarrollo fuertemente integrado, como LabVIEW, según la información obtenida de una publicación de la empresa Highlights (www.highlights.com.ec), permite obtener numerosos beneficios, tales como:

- Fuerte incremento de la productividad en los procesos de desarrollo, instalación, mantenimiento y modificación mediante una aproximación de desarrollo intuitiva que está optimizada para aplicaciones de medición y control.
- Mayor rendimiento para sistemas de monitoreo en tiempo real y control.
- Mayor integración, lo que permite interconectar instrumentos de medición y control de muy diversos fabricantes dentro de sistemas de más alto nivel que se pueden conectar fácilmente a otros procesos dentro de la organización.
- Menores costos durante la vida útil de los productos.

En el módulo de LabVIEW DSC viene la aplicación ***NI Image Navigator***, **figura 1.10**, que es un catalogo de más de 4000 símbolos industriales, en el cual se incluyen bombas, tubería, válvulas, tanques, mezcladoras, motores, ductos,

símbolos eléctricos, Sensores, transmisores, símbolos ISA y más, permitiendo así la personalización del HMI/SCADA dentro de LabVIEW.

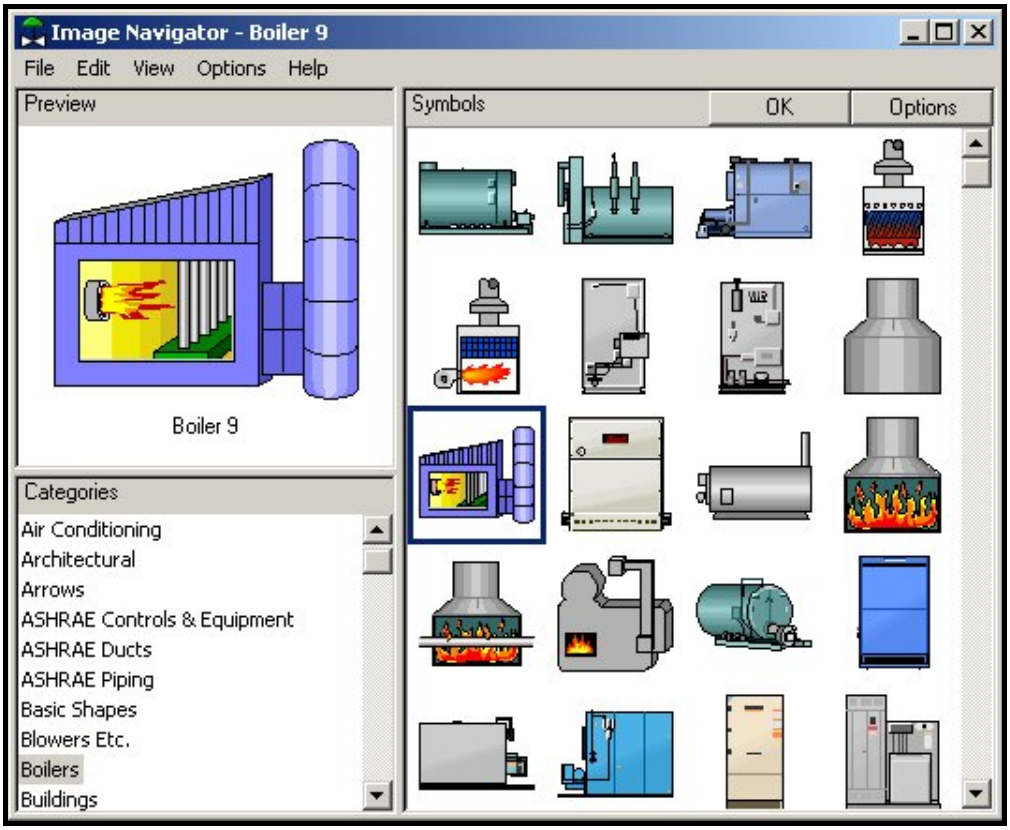


Figura 1.10 "*NI Image Navigator*".

En la **figura 1.11** se muestra un ejemplo de una aplicación de un HMI/SCADA realizado con LabVIEW y el módulo LabVIEW DSC, para un intercambiador de calor.

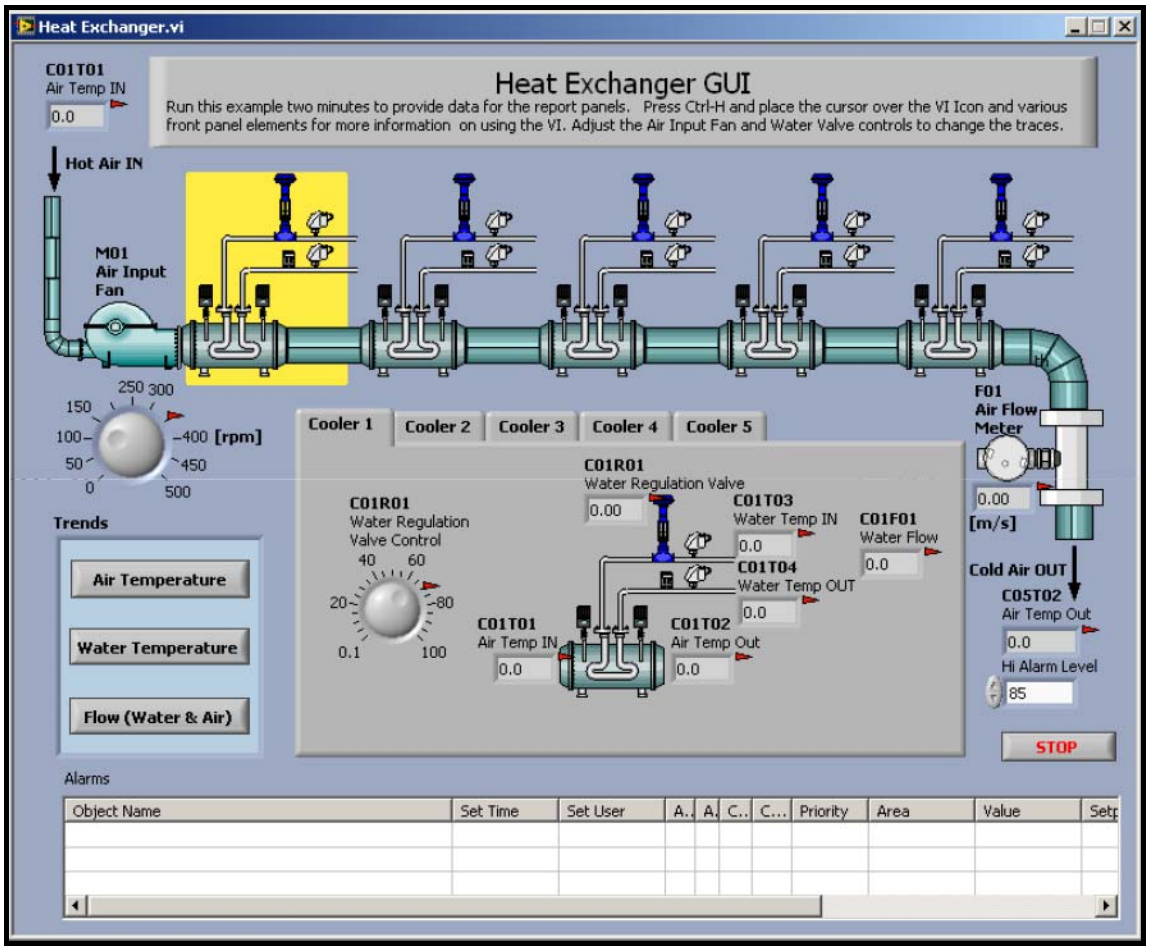


Figura 1.11 Ejemplo de un sistema HMI/SCADA implementado con LabVIEW.

(Tomado de los Ejemplos de LabVIEW)

En el **capitulo 4**, se volverá a retomar este tema explicando ya un poco más con ejemplos sobre el uso de LabVIEW DSC para aplicaciones SCADA, más concretamente para la aplicación de la presente tesis.

CAPÍTULO 2

LABVIEW Y VISIÓN ARTIFICIAL BASADA EN PC CON LABVIEW

- **Introducción al entorno del Software LabVIEW ™.**
- **Visión Artificial utilizando LabVIEW.**

CAPÍTULO 2

LabVIEW Y VISIÓN ARTIFICIAL BASADA EN PC CON LabVIEW

2.1 Introducción al entorno del software LabVIEW.

LabVIEW ™ (*Laboratory Virtual Instrument Engineering Workbench*) es un lenguaje de propósito general, como lo es el Lenguaje C ó Basic, pero con la característica principal que es totalmente gráfico y que usa iconos en lugar de líneas de texto para crear aplicaciones. A diferencia de los lenguajes de programación basados en texto, donde las instrucciones determinan la ejecución del programa, LabVIEW usa programación de "**FLUJO DE DATOS**", donde el flujo de los datos va ha determinar el orden de la ejecución del programa.

Al ser un lenguaje gráfico ó lenguaje "G" simplifica notablemente el desarrollo de aplicaciones, minimizándose el tiempo de programación y facilita el entendimiento y manejo de dicho lenguaje para el diseñador y programador.

LabVIEW es una herramienta diseñada especialmente para monitorizar, controlar, automatizar y realizar cálculos complejos de señales analógicas y digitales capturadas a través de tarjetas de adquisición de datos, puertos serie y GPIBs (Buses de Intercambio de Propósito General), etc.

Los programas desarrollados en LabVIEW son llamados Instrumentos Virtuales (VIs).

Un Instrumento Virtual (VI) es un módulo de software que simula el Panel Frontal de un instrumento común y apoyándose en elementos de hardware accesibles por el PC (tarjetas de adquisición, tarjetas DSP, Hardwares con Interfaces USB, instrumentos accesibles vía GPIB, VXI, RS-232, etc.) realiza una serie de medidas como si se tratase de un instrumento real.

En éste capítulo se tratará de una manera general los conceptos asociados con LabVIEW y su entorno de programación.

La empresa National Instruments (www.ni.com) oferta varios Paquetes de Aplicación que permiten crear Instrumentos Virtuales, que resultan muy completas y brindan importantes facilidades a Diseñadores dedicados a la Especialidad de Instrumentación y Control.

LabVIEW™ (en todas sus versiones), es un Conjunto de Paquetes de Aplicación desarrollado por National Instruments Corporation.

LabVIEW también se distingue por las siguientes facilidades:

- Posee un compilador intérprete (en línea).
- Es una herramienta de Programación Visual que contiene un conjunto muy variado de objetos con apariencias muy cercanas a Controles de un Panel en instrumentos reales como por ejemplo osciloscopios y multímetros, etc., lo que facilita enormemente la personalización de los Instrumentos Virtuales creados y ahorra tiempo al diseñador.
- Es capaz de manipular los tipos de datos y estructuras clásicas que se manejan en los Lenguajes de Programación convencionales.
- Posee una extensa Biblioteca de Funciones prefabricadas que permite cubrir la mayoría de las necesidades del diseñador en Adquisición, Procesamiento, Almacenamiento, Control y/o presentación de datos.
- A pesar de la existencia de gran variedad de Funciones Prefabricadas, LabVIEW también permite la Programación Libre de Aplicaciones, tal y como si se trabajara con un Lenguaje de Programación Convencional.
- National Instruments Corporation, oferta diferentes alcances de una misma versión del Paquete LabVIEW; desde una versión de Evaluación (válida por 30 días) pasando por una versión ilimitada para estudiantes; hasta el Paquete Profesional ó completo que incluye absolutamente todos los módulos auxiliares (“toolkits”) y el “Application Builder”, que permite generar aplicaciones ejecutables sobre cualquier Sistema Operativo (Windows, Linux, Solaris, etc.) independientes de la Plataforma LabVIEW.

En la **figura 2.1** se muestra como abrir y ejecutar LabVIEW, ya sea desde un VI en blanco, ó iniciar con un ejemplo.

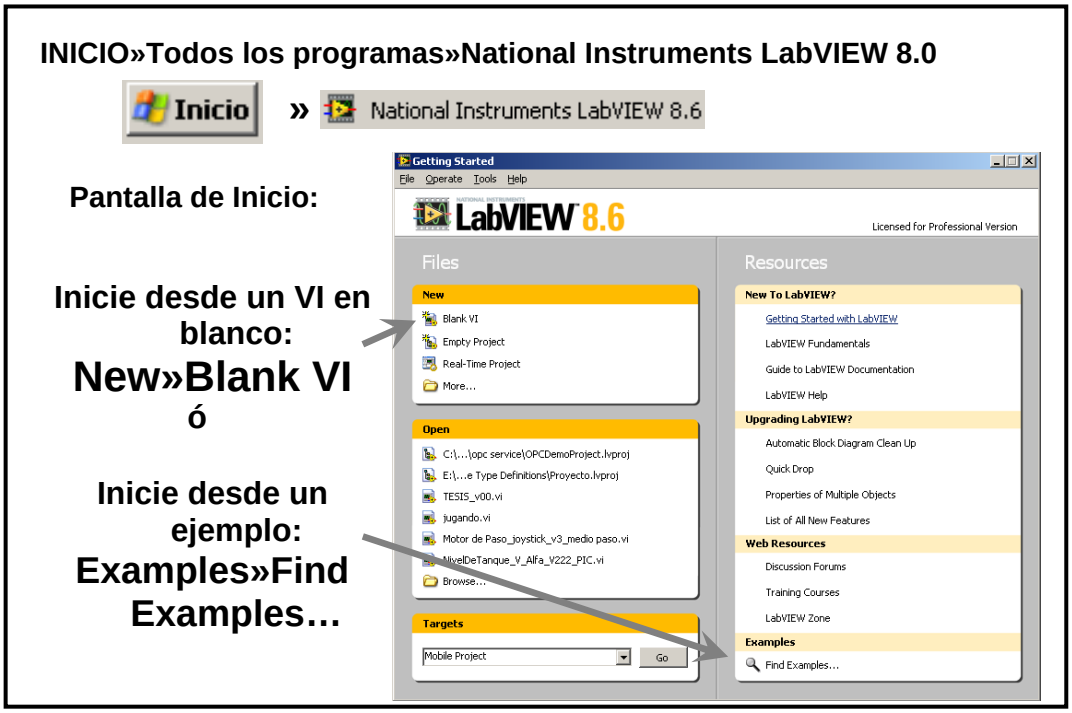


Figura 2.1 Abrir y Ejecutar LabVIEW.

2.1.1 Las partes principales de un VI.

Cada VI contiene tres partes principales:

- El “Panel Frontal” ó “***Front Panel***”: es aquí donde el Usuario interactúa con el VI, la Interfaz Hombre Máquina ó HMI.
- El “Diagrama de Bloques” ó “***Block Diagrams***”: Es el código que controla el programa.
- El “Icono/Conector” ó “***Icon/Connector***”: sirve para la conexión de un VI a otros VIs.

En LabVIEW el “Front Panel” y el “Block Diagram”, son dos ventanas separadas, pero están relacionadas entre sí. En la **figura 2.2** se ilustra las partes de un VI.

En LabVIEW, se puede construir la Interfaz de Usuario ó Interfaz Hombre Máquina (HMI) usando un conjunto de herramientas y objetos. La Interfaz Usuario es conocido como el “Front Panel”. Luego se puede agregar el código usando una

representación grafica de funciones para controlar los objetos del Panel Frontal. El diagrama de bloques contiene este código. De alguna forma, el Diagrama de Bloques reensambla el Diagrama de flujo.

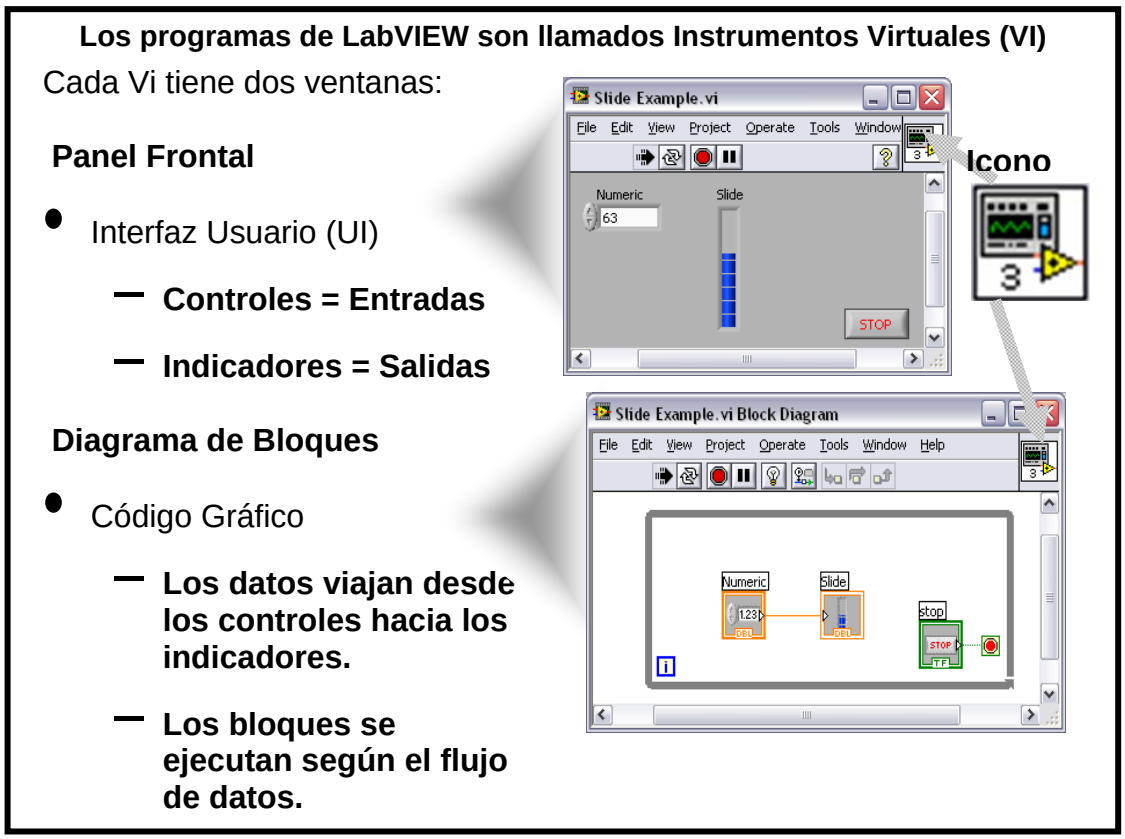


Figura 2.2 Las partes principales de un VI.

Los usuarios interactúan con el Panel Frontal cuando el programa se está ejecutando. Los usuarios pueden controlar el programa, cambiar las entradas, y ver los cambios de los datos en tiempo real. Los controles son usados para ingresar o modificar algo, tal como: ajustar un slide para establecer el valor de una alarma, cambiar de estado a un Switch (On u Off) ó parar el programa. Los indicadores son usados como salidas. Termómetros, leds, y otros indicadores muestran los valores de salida del programa, esto puede ser datos, estado del programa u otra información.

Cada Control e Indicador en el Panel Frontal tiene una Terminal correspondiente en el Diagrama de Bloques. Cuando un VI está corriendo ó ejecutándose, los valores de los controles fluyen a través del Diagrama de Bloques, donde los valores de dichos controles son usados en diversas funciones dentro del diagrama y el resultado se pasa a otras funciones ó a los Indicadores a través de los “cables”.

2.1.2 Paleta de Control y Paleta de Funciones.

2.1.2.1 Paleta de Control

La Paleta de Control (**figura 2.3**) se usa para colocar los Controles e Indicadores dentro del Panel Frontal. La Paleta de Controles está disponible únicamente en el Panel Frontal. Para ver dicha paleta se va a la barra del menú principal y se selecciona "***Window »Show Controls Palette***". Pero también se puede mostrar la paleta haciendo clic derecho sobre un área desocupada del Panel Frontal. Al abrir la paleta de control se puede ver una imagen de una tachuela en la parte de arriba al lado Izquierdo, al pulsar sobre ella hace que la paleta se quede presente en el Panel Frontal.

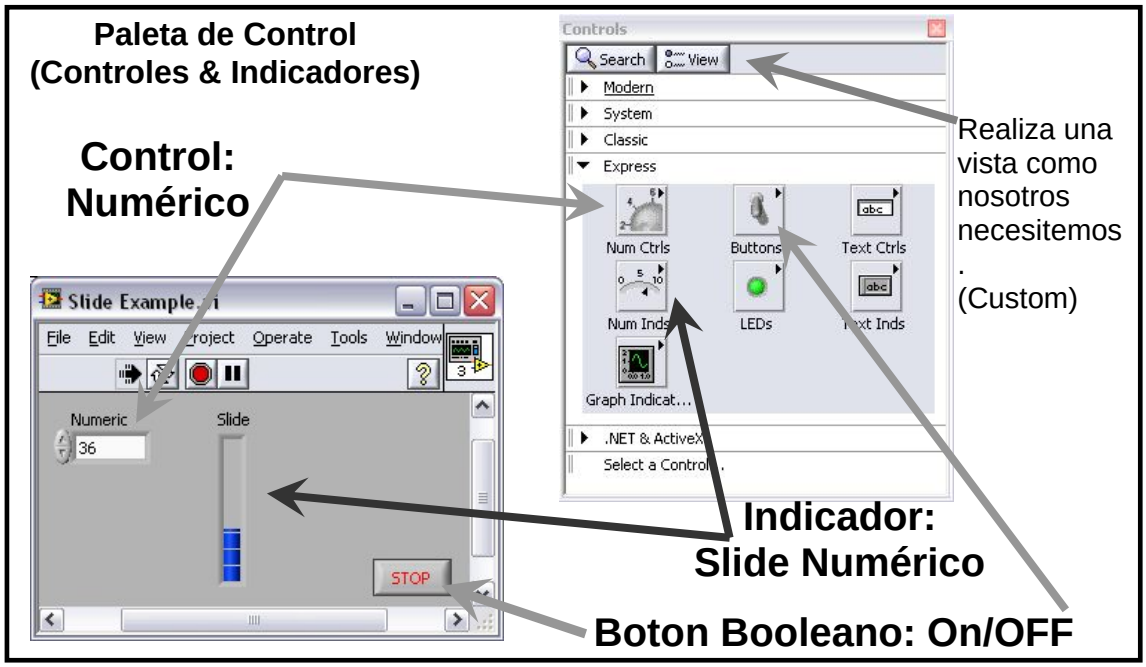


Figura 2.3 Paleta de Control.

2.1.2.2 Paleta de Funciones (y estructuras)

La paleta de Funciones se usa para construir un diagrama de bloques.

La paleta de Funciones (**figura 2.4**) está disponible únicamente en el Diagrama de Bloques. Para ver la paleta de funciones, seleccione: "***Window » Show Functions Palette***", también se puede mostrar dicha paleta haciendo un clic derecho sobre una parte libre en el Diagrama de Bloques. Se puede dejar estática esta paleta haciendo un clic sobre la tachuela.

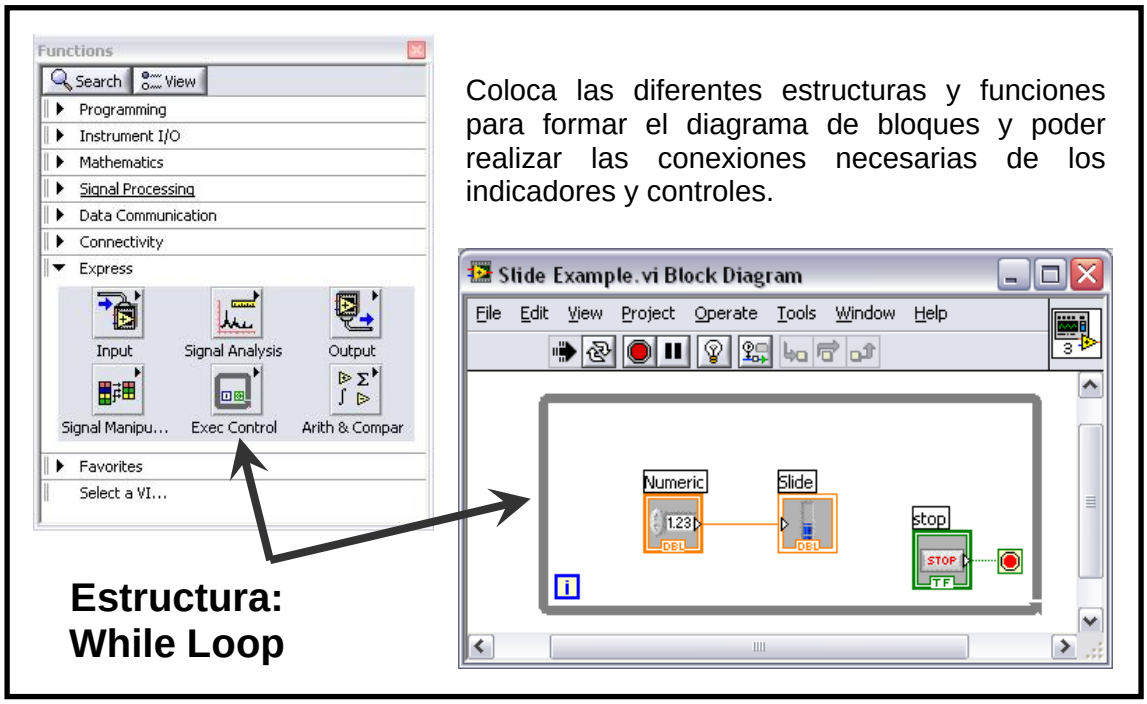


Figura 2.4 Paleta de Funciones.

2.1.3 Paleta de Herramientas.

En esta paleta se concentran todas las herramientas necesarias para la edición tanto en el Panel Frontal como en el Diagrama de Bloques. En la **figura 2.5** se encuentra dicha paleta, en esta figura se menciona la herramienta de selección automática. Si se encuentra activada la paleta de funciones en modo automático, y si se mueve el cursor del ratón sobre los objetos en el diagrama de Bloques ó en el Panel Frontal, LabVIEW automáticamente selecciona la herramienta correspondiente de la paleta de herramientas.

En la **tabla 2.1 (a)** y **(b)** se resume lo que hace cada herramienta dentro de la paleta de Herramientas. Para sacar la paleta de herramientas: "*View ›› Tools Palette*", ó pulsando Shift + clic derecho.

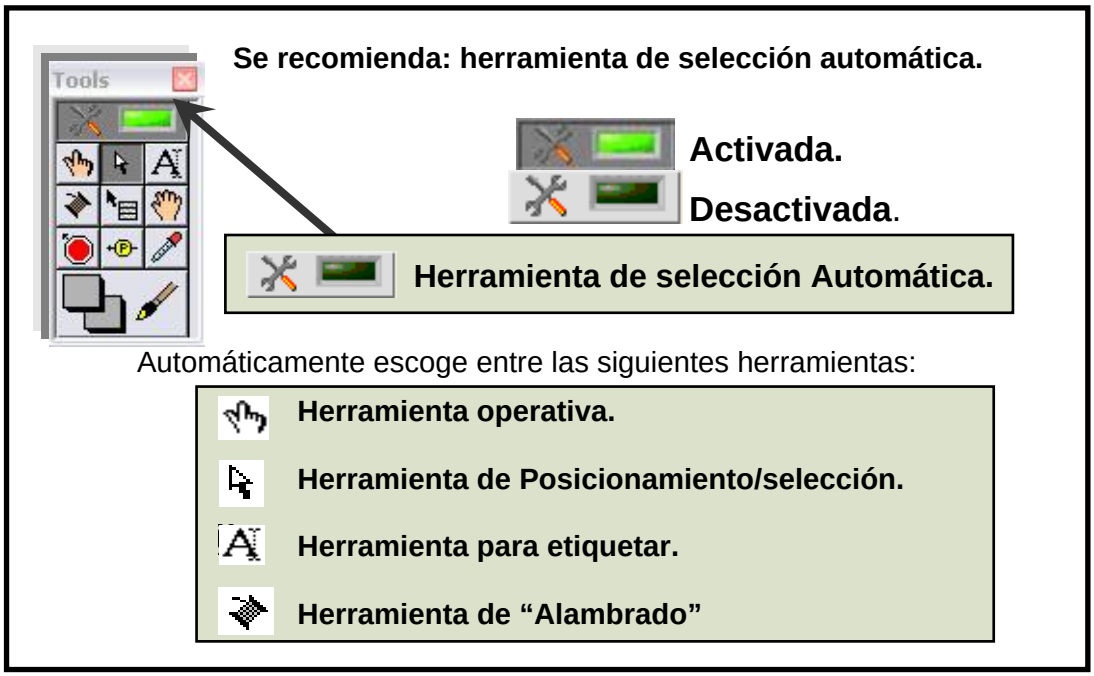


Figura 2.5 Paleta de Herramientas.

Herramienta:	Función:
	Herramienta de selección automática. Cuando se encuentra activado permite la selección automática de las siguientes herramientas:
	Herramienta operativa. Permite operar controles e indicadores y/o modificar sus valores. La mayor parte de las funciones están asociadas al Panel. En el diagrama si se da clic sobre el terminal directamente se "salta" al panel y se selecciona automáticamente (con líneas de puntos), el terminal asociado.
	Sirve para seleccionar y/o mover un objeto dado. El objeto seleccionado se rodea de una línea discontinua
	Sirve para poner textos, o editar alguno ya puesto.
	Herramienta de "alambrado". Sirve para unir terminales en el Diagrama de Bloques. Cada unión o "wire", guía el flujo de los datos. Cuando la unión es inválida, el compilador en lugar de un "wire" coloca una línea discontinua. Esta herramienta también se utiliza para definir los conectores del VI en el icono de la esquina superior derecha del panel.

Tabla 2.1 (a).- Resumen de la Paleta de Herramientas y sus diversas funciones, asociadas a la herramienta de selección automática.

Herramienta:	Función:
	OTRAS HERRAMIENTAS IMPORTANTES, estas herramientas ya no forman parte de la selección automática.
	Herramienta de "*scrolling*" ó desplazamiento: Sirve para realizar un paneo de la pantalla.
	Sirve para colocar puntos de ruptura en el código para facilitar la simulación.
	Sirve para colocar puntos de prueba ó visualización de datos en el diagrama. Los puntos aparecen también en el panel.
	Sirve para capturar el color de algún objeto, con el objetivo de colorear otro con el mismo color.
	Sirve para colorear objetos. Al ser seleccionada, con clic derecho se visualiza una paleta de colores dinámica que hace cambiar de color al objeto a medida que el pincel se desplaza por encima de él.
	Herramienta para colocar accesos directos al menú.

Tabla 2.1 (b).- Resumen de la Paleta de Herramientas y sus funciones adicionales.

2.1.4 Barra de Herramientas de Estado.

La barra de herramientas de Estado permite realizar algunas funciones con respecto al curso que toma el Instrumento Virtual, como por ejemplo ejecutar el VI, permite ver como se comporta el VI paso a paso para comprobar el estado de la programación del mismo, alinear los iconos, etc. En la **figura 2.6** se muestra la Barra de Herramientas de Estado de LabVIEW.

En la **tabla 2.2 (a)** se resumen los botones y sus funciones, para la Barra de Herramientas de Estado. Mientras en la **tabla 2.2 (b)** se muestran los botones adicionales que existen en la barra de herramientas de Estado en el diagrama de Bloques.

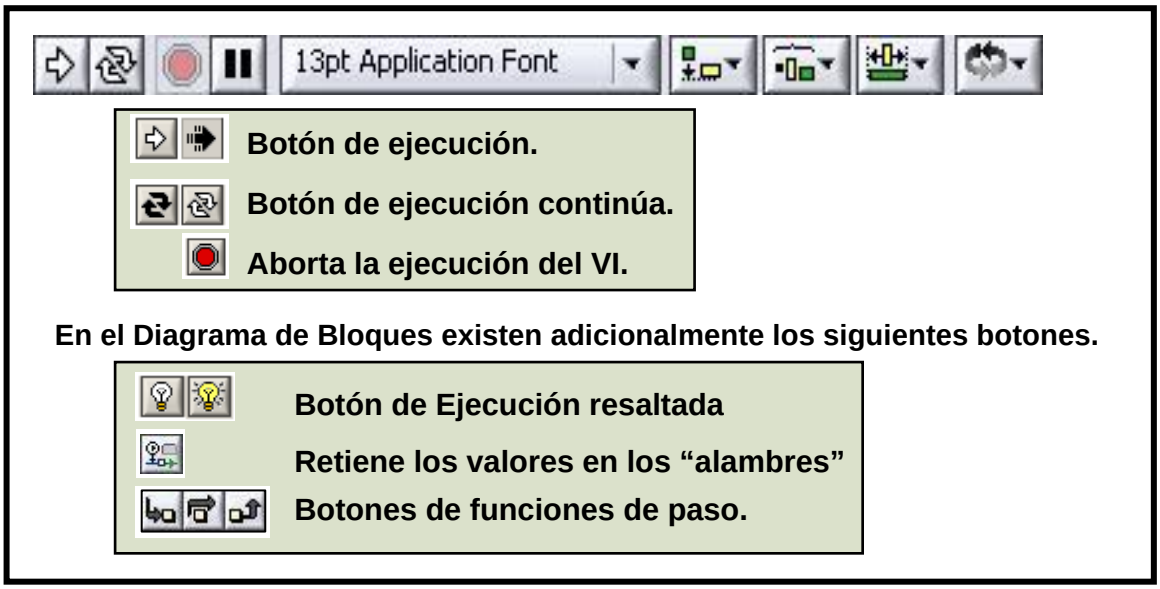


Figura 2.6 Barra de Herramienta de Estado.

2.1.5 Creación de un Instrumento Virtual (VI).

Cuando se crea un objeto en el Panel Frontal se crea al mismo tiempo una terminal en el Diagrama de Bloques; estas terminales tienen acceso a los objetos del panel Frontal con el código creado en el Diagrama de Bloques. En la **figura 2.7** se ilustra la creación de un VI.

Cada terminal contiene información útil referente al objeto al cual corresponde en el panel frontal. Por ejemplo, el color y el símbolo proporcionan la información acerca del tipo de dato, por ejemplo, Números de punto flotante y de doble-precisión, son representados con terminales anaranjadas y las letras DBL. Las terminales booleanas son verdes y son representadas por las letras TF.

En general, las terminales anaranjadas deben unirse (cablearse) con las terminales anaranjadas, verdes con verdes, y así sucesivamente. Claro que ésta no es una regla que no se puede romper; por ejemplo LabVIEW permitirá al usuario conectar una terminal azul (valor entero) a una terminal anaranjada (valor fraccional). Pero en la mayoría de casos, se debe buscar una igualdad en colores.

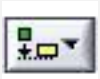
Botón:	Función:
	 **Botón para ejecutar el VI**.  Indica que el VI esta ejecutándose.
	**Botón de ejecución continúa**:  Botón para iniciar la ejecución continúa.  Indica el VI esta en modo de ejecución continua. Para salir de este modo solamente se debe dar un clic de nuevo en éste botón.
	**Botón para abortar la ejecución** del VI, aparece activo cuando el VI esta ejecutándose. ***Nota***.- Se debe evitar el uso del botón “Abort Execution” para detener el VI. Se debe por lo menos permitir que el programa termine con la ejecución de todos sus flujos de datos ó también se puede diseñar un método que permita al programa detenerse automáticamente. Esto se hace para que los VI estén o tengan en un estado conocido. Por ejemplo se puede colocar en el Panel Frontal un botón que **pare** el VI cuando se pulse sobre él.
	**Botón de Pausa/Continuación**: Detiene momentáneamente la ejecución de un VI. Cuando se da un clic en el botón de pausa LabVIEW revisa en el Diagrama de Bloques la localización de en donde se detuvo la ejecución. Si se da un clic en el botón de “Pause” de nuevo el programa vuelve a correr.
	**Configuración de Textos**: Se despliega hacia abajo un menú, en los que aparecen opciones para cambiar el tamaño de las letras, su forma, color, etc.
	**Alineamiento de Objetos**: se despliega un menú de herramientas que permiten alinear los objetos en los dos ejes (x , y), por ejemplo a la izquierda, derecha a lo largo, etc.
	**Distribución de Objetos**: se despliega un menú, que contiene herramientas para distribuir los objetos, espaciándolos sea verticalmente u Horizontalmente.
	**Redimensionamiento de Objetos**: se despliega un menú en el que nos da herramientas para manipular el tamaño de los objetos. Por ejemplo al seleccionar objetos (solo del Panel Frontal) de varios tamaños y grosores se puede hacer que todos tengan el mismo alto ó la misma altura, o sean tan pequeños como el más pequeño, etc.
	**Redimensionamiento de objetos**: se despliega un menú en el que podemos a los objetos agruparlos, enviarlos al fondo, traerlos al frente, etc.

Tabla 2.2 (a) Resumen de la barra de herramientas.

Botón:	Función:
	**Botón de ejecución resaltada**: se pueden observar como fluyen los datos en el Diagrama de Bloques. Para desactivar ésta herramienta se vuelve a dar un clic sobre si misma.
	**Retener valores de los alambres**: para guardar los valores que han circulado por el alambre en cada punto durante el flujo de la ejecución. Es como poner una punta de prueba sobre alguna parte del alambre y obtener así el valor más reciente de los datos.
	**Botón de entrada al ciclo (Step Into)**: para correr el programa a un paso a la vez, ingresar a un loop, a un SubVI, etc. Se simula paso a paso a través de los VI y de nodo a nodo. Cada nodo sobresale para denotar cuando está listo para ejecutarse.
	**Botón de salto (Step Over)**: como se puede observar en su símbolo, sirve para saltar nodos, loops, SubVIs, etc.
	**Botón de salida (Step Out)**: sale de un loop, SubVI, etc. Para salir de un nodo se debe cumplir primero la simulación paso a paso por cada nodo.

Tabla 2.2 (b) Resumen de los botones adicionales de la Barra de Herramienta de Estado en el Diagrama de Bloques.

Las terminales correspondientes de los controles en el Diagrama de Bloques tienen una flecha al lado derecho y tienen además un borde grueso (), mientras los indicadores () tienen una flecha en el lado izquierdo y un borde fino.

 Una regla lógica puede ser aplicada al momento de alambrar o cablear en LabVIEW: **cada cable debe tener una (pero solo una) fuente (ó control) y cada cable puede tener uno ó varios destinos (ó indicadores)**.

El VI que se ha creado en éste caso es para adquirir una señal cualesquiera, esta vez se ésta adquiriendo una señal de onda cuadrada, de 10Vp-p. Cabe recalcar que además de las terminales del Panel Frontal, el Diagrama de Bloques contiene obviamente sus propias funciones que le son necesarias para programar el VI, como puede ser el “DAQ Assistant”, funciones para sumar, restar, multiplicar, sacar coeficientes, realizar transformadas rápidas de Fourier, integrar, etc., etc.… Cada

una de estas funciones puede tener varias terminales de entrada y de salida. La conexión entre estas terminales es una parte muy importante dentro de la programación en LabVIEW.

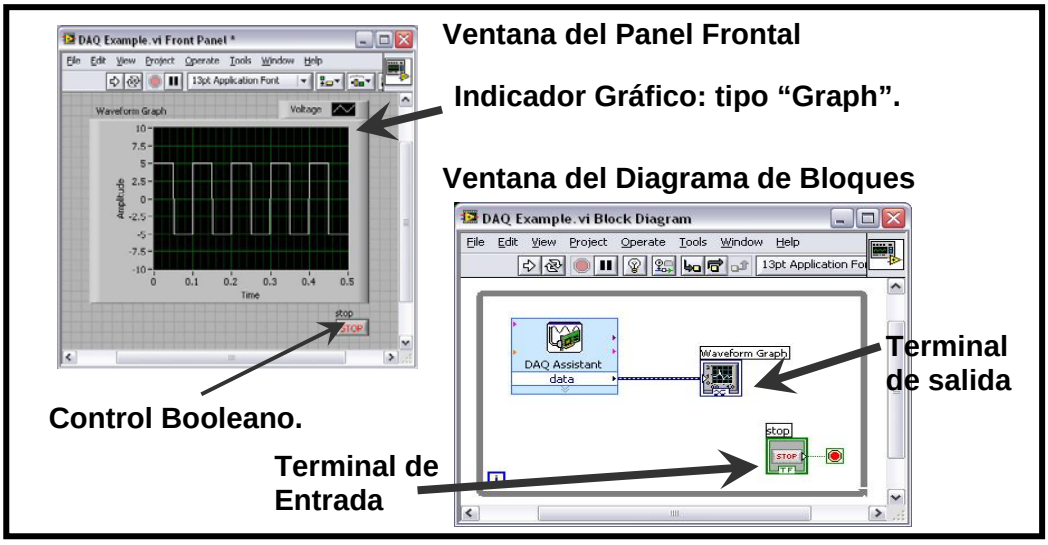


Figura 2.7 Demostración de la creación de un Instrumento Virtual (VI).

Una vez que se tenga un poco de experiencia programando en LabVIEW, el cableado se hace fácil. A continuación se dan algunas recomendaciones para comenzar con el cableado:

- La herramienta para conectar o de cableado es utilizada para conectarse a los nodos de las funciones. Cuando se apunte con la herramienta de cableado, se debe apuntar con el extremo del cable que cuelga del carrete. Aquí es donde el cable será colocado. **Figura 2.8 (a)**

- Mientras se mueve la herramienta de cableado sobre las funciones, observe la viñeta amarilla que aparece, dice el nombre de la terminal al que se esta conectando. **Figura 2.8 (b)**.

- Mientras se mueva la herramienta de cableado encima de una terminal, esta va a mostrar información. Esto ayuda a identificar donde se va a unir el cable. **Figura 2.8 (c)**.

- Para más ayuda con los terminales, haga clic derecho en la función y seleccione Visible Items u Objetos Visibles >> Terminals ó Terminales, se

mostrara un dibujo de la función que va ha revelar las terminales de la conexión. Note los colores, estos corresponden a los tipos de datos utilizados por los terminales del panel frontal. **Figura 2.8 (d)**.

- Para ayuda adicional, seleccione "*Help >> Show Context Help*", o presione CTRL+H. Esto mostrara la ventana de ayuda en contexto. A medida que uno mueva el ratón (Mouse) sobre la función, esta ventana le mostrara la función, terminales, y una breve descripción. Utilice esto junto con otras herramientas para ayudarse mientras conecta los cables. **Figura 2.8 (e)**.

- Si el cableado no se mira muy bien, haga clic derecho en el cable que se desea arreglar y escoja la opción de Clean Up Wire ó Limpieza del Cable para que automáticamente el cable haga su ruta de nuevo. **Figura 2.8 (f)**.

Por lo general al momento de cablear no se debe preocupar por el color de los cables, ya que, LabVIEW seleccionará automáticamente el cable correcto y adecuado para cada situación.

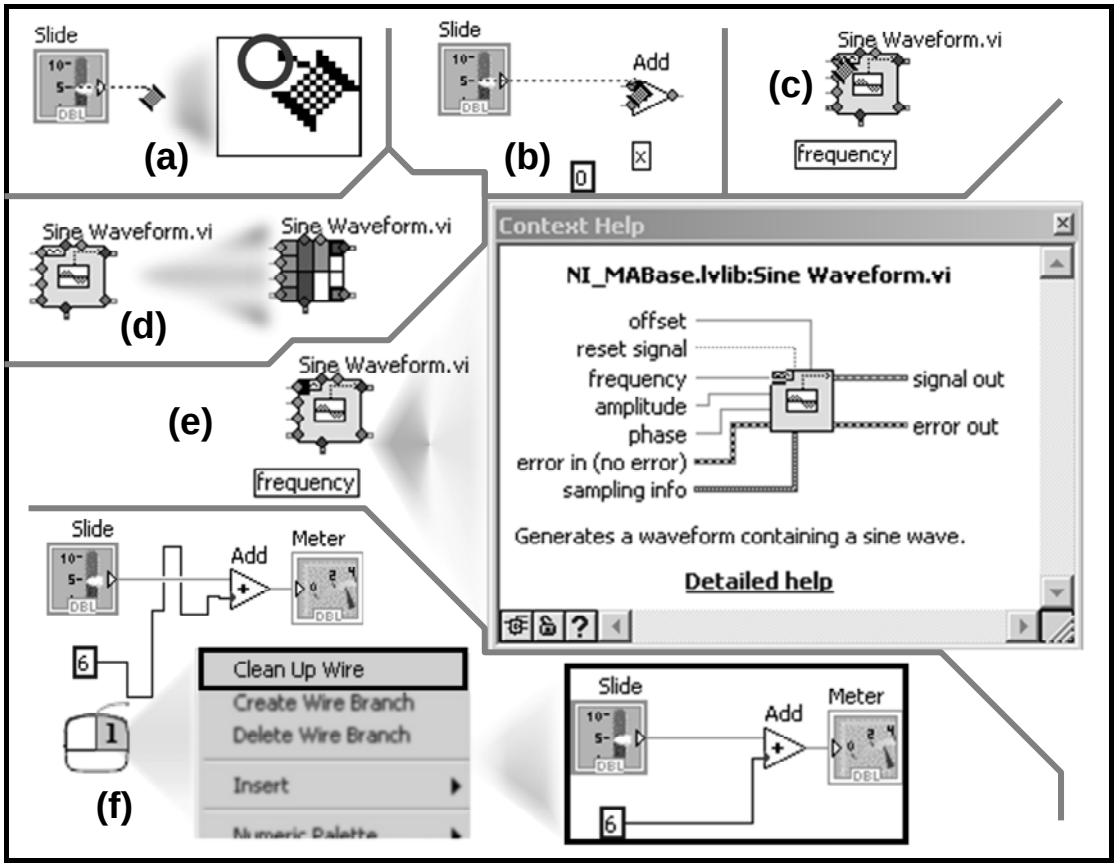


Figura 2.8 Recomendaciones para el cableado.

2.1.6 Programando el Flujo de Datos.

LabVIEW sigue un modelo de flujo de datos para ejecutar los VIs. Un nodo del diagrama de bloque se ejecuta cuando todas sus entradas están disponibles. Cuando un nodo completa la ejecución, suministra datos a sus terminales de salida y pasa los datos de salida al siguiente nodo en la trayectoria del flujo de datos. Visual Basic, C++, JAVA y otros lenguajes de programación basados en texto, siguen un modelo de control de flujo de la ejecución de un programa. En flujo de control, el orden secuencial de los elementos del programa determina el orden de ejecución de un programa.

Considérese el ejemplo de la **figura 2.9 (a)**, se suman dos números y luego se multiplica el resultado de la adición por 2; en éste caso, el diagrama de bloque se ejecuta de izquierda a derecha, no porque los objetos están puestos en ese orden, sino porque una de las entradas de la función de multiplicación no es valida hasta que la función de suma haya terminado su ejecución y pasado los datos a la función de multiplicar. Cabe recordar que: un nodo se ejecuta solamente cuando tiene datos disponibles en todas sus terminales de entrada y suministra datos a sus terminales de salidas solamente cuando termina su ejecución. En la segunda parte del código (**Figura 2.9 b**), el VI "*Simulate Signal Express*" recibe los datos desde los controles y muestra el resultado en el graficador ("*Waveform Graph*").

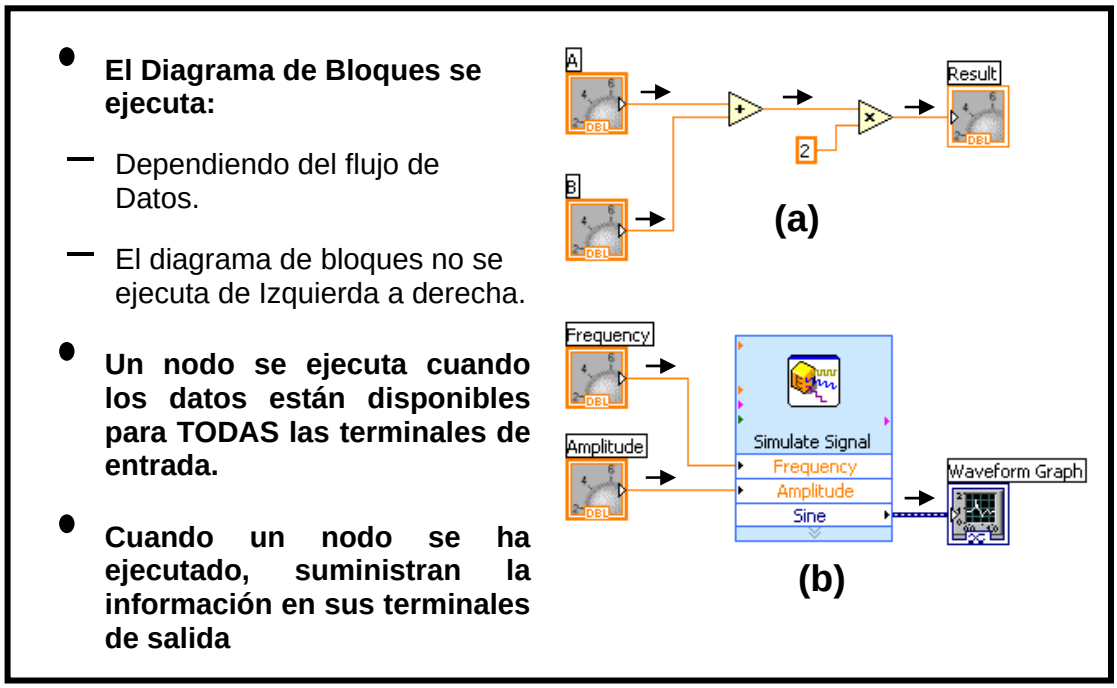


Figura 2.9 Programando el flujo de datos.

Se debe aclarar que la parte a y b de la **figura 2.9** forman parte del mismo VI, es decir, que ambos se están ejecutando de forma paralela, esto significa que ambos empiezan ejecutándose al mismo tiempo pero de forma independiente.

Esto es algo importante que incorpora LabVIEW, ya que, con las nuevas tecnologías de procesadores de doble núcleo ó en otros casos de multi-núcleo se podrían aprovechar para que se ejecute ciertos códigos de forma independiente en cada procesador, y algo importante: sin necesidad de implementar un código adicional para realizar dicha tarea.

Ahora, se debe aclarar que no siempre el código se ejecuta de izquierda a derecha, todo depende del flujo de datos de las funciones para saber cual función ó nodo se ejecuta primero, por ejemplo, en la **figura 2.9** el segmento **a** y el **b** se ejecutara primero la suma ó el VI *Simulate Signal Express*, ya que no se puede saber porque las entradas a las funciones de suma y al VI Express están disponibles al mismo tiempo y la constante numérica no tiene entradas. Es una situación en donde un segmento del código se debe ejecutar antes que otro, y no existe dependencia de datos entre las funciones. Claro que esto se puede solucionar, si es que se desea que un segmento se ejecute primero luego otro, etc., se puede utilizar una estructura de Secuencia para forzar el orden de la ejecución, (disponible en la subpaleta de funciones "*Structure*"), así se da un orden secuencial, en el caso de que se necesite.

2.1.7 Técnicas para Eliminar Errores (Debugging).

LabVIEW posee un compilador intérprete en línea, es decir, mientras se está realizando el programa, LabVIEW automáticamente va realizando un Debugging del segmento de código, lo que permite al diseñador ó programador saber a tiempo que es lo que está haciendo mal, cosa que no pasa con otros lenguajes de programación que se tenía que programar 50 ó más líneas de código, compilar y saber recién en donde está el error.

Cuando se produce un error de programación, LabVIEW no permite que el VI se ejecute, y aparece el botón de ejecutar ("*RUN*") con una flecha quebrada en la barra de Herramientas, **figura 2.10 (a)**.

a. Encontrando Errores.

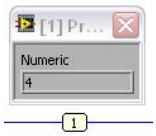
El botón "***Run***" aparece con la flecha rota, avisando que hay un error de programación e impidiendo que el VI se ejecute

b. Resaltando la Ejecución

Botón de ejecución resaltada; el flujo de datos es animado utilizando "burbujas". Los valores se despliegan en los cables.

c. Herramienta de Prueba

Probe



Al dar clic con el botón derecho sobre el cable se muestra la ventana de prueba y así se ve los datos mientras fluyen por el segmento de cable.

También puede seleccionar la herramienta de prueba desde la paleta de herramientas y hacer un clic en el cable.

Figura 2.10 Algunas técnicas para eliminar errores.

Encontrando errores: Para hacer una lista de los errores, se hace clic en la flecha quebrada y si se desea localizar el objeto malo se hace clic en el mensaje del error y va ha resaltar el error en el Diagrama de Bloques.

Resaltando la Ejecución: (**figura 2.10 (b)**) Anima el diagrama de bloques y traza el flujo de datos, permitiendo ver los valores intermedios, para realizar esto se debe hacer clic en el bombillo incandescente (Light bulb) en la barra de herramientas.

Probe: (**figura 2.10 (c)**) Utilizado para ver los valores en los arrays (arreglos) y clusters, o simplemente para ver que valor esta circulando por un cable, para utilizar esta herramienta se hace clic derecho sobre el cable u objeto que se desea ver el valor y se selecciona "*Probe*", ó simplemente cuando el VI está ejecutándose se acerca el puntero del ratón sobre el cable y se da un clic.

Herramienta para retener los valores en los cables: tiene un uso parecido al probe, muestra el valor de la última iteración del VI.

Break point ó Punto de Paro: Coloca pausas en diferentes lugares del diagrama, para esto se hace clic en los cables ó en los objetos con la herramienta de Punto de Paro para colocar dichos puntos.

2.1.8 Opciones de Ayuda

LabVIEW tiene una ventana adicional llamada "***Context Help***" ó Ayuda Contextual, en la que muestra la información básica de los objetos colocados ya sea en el Panel Frontal ó en el Diagrama de Bloques, cuando se mueve el cursor por encima del objeto que se desee ver. Esta ventana es de gran ayuda en el Diagrama de Bloques sobre todo al momento de hacer el cableado para saber a que terminales se están conectando. En la **figura 2.11** se muestra un ejemplo que hace alusión a ésta ventana.

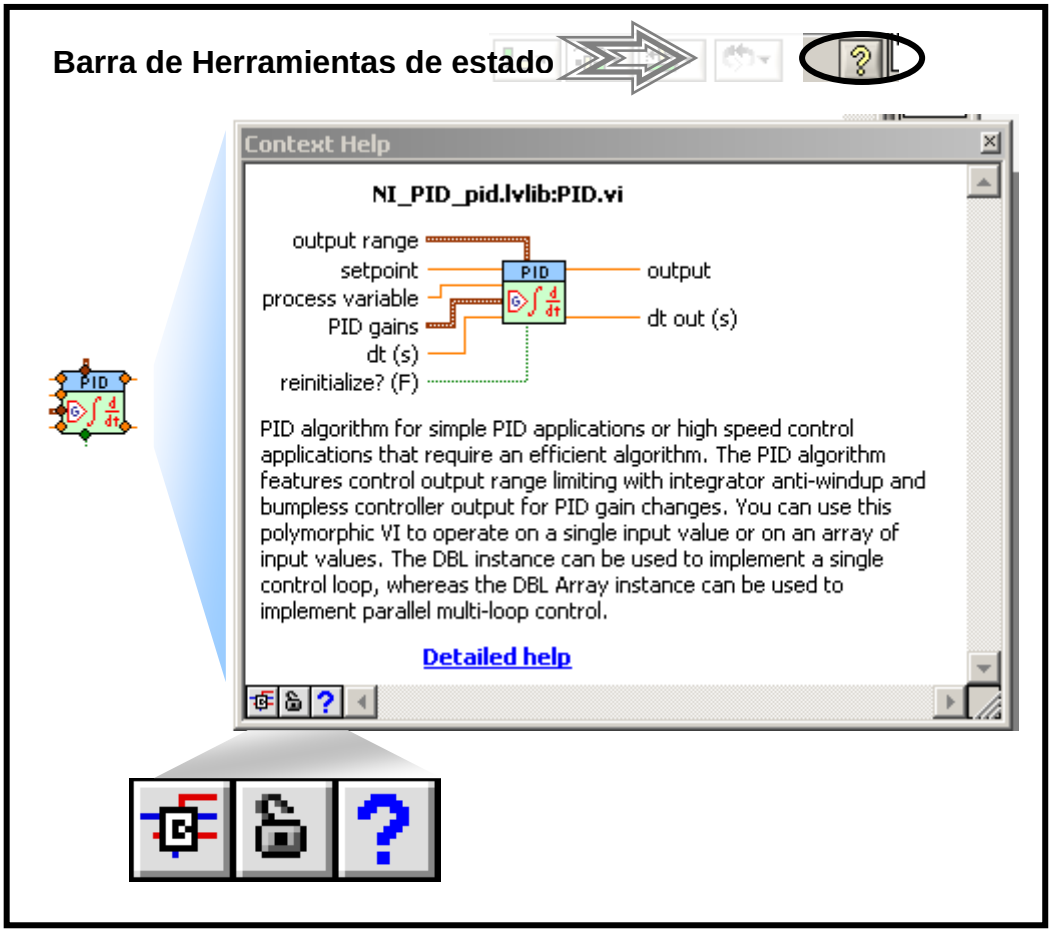


Figura 2.11 Ventana de Ayuda Contextual.

Para desplegar la ventana de Context Help, se puede seleccionar "*Help>>Show Context Help*", presionando las teclas <Ctrl+H>, ó como se observa en la **figura 2.11** al hacer un clic en el icono "*Show Context Help*" de la barra de herramientas.

En la ventana de Ayuda Contextual muestra la información del objeto de forma resumida haciendo énfasis en lo más importante, pero también se puede, si es que

se desea, mostrar una ayuda más detallada haciendo clic en “Detailed Help” ó en el icono ➔  .

El icono  sirve para mostrar las terminales opcionales, en el caso de haberlas, además muestra el path ó la dirección en donde se encuentra ubicado por ejemplo: C:\...ts\LabVIEW 8.6\vi.lib\addons\control\pid\pid.llb\PID.vi .

El icono  congela el contenido actual de la ventana de ayuda contextual, es decir, si es que se pasa el puntero del ratón sobre otro objeto para ver la ayuda la ventana va ha seguir mostrando la ayuda del objeto en el cual se bloqueo. Para desbloquear se vuelve hacer un clic sobre dicho icono.

2.1.9 Algunos “Tips” para trabajar en LabVIEW.

En LabVIEW se pueden realizar algunas combinaciones de teclas que hacen que sea más fácil realizar el trabajo de la programación. Las combinaciones más comunes se muestran en la **tabla 2.3**.

Se había comentado que una de las herramientas que permiten realizar un trabajo más fácil es la herramienta de selección automática, pero también cuando ésta se encuentra desactivada se pueden realizar algunos “trucos”, aplastando la tecla “tab” se pueden intercambiar entre las principales operaciones de la paleta de herramientas que se encontraban en la **tabla 2.1 (a)**. Si es que se desea volver a activar la herramienta de selección automática se puede hacer la combinación “shift + tab”.

Si es que se desea configurar algunas opciones del panel frontal y del diagrama de bloques a nuestro gusto se puede ir a **Tools >> options**, para configurar los colores, impresiones, y otras más opciones.

También hay como configurar las propiedades especificas del VI, para esto se puede ir a **File >> VI Properties**… En éste lugar se puede realizar operaciones como la documentación del VI, configuraciones de seguridad, cambiar la apariencia de la ventana y hacer cambios de la ventana a nuestro gusto.

Combinación	Función
Ctrl + H	Activa o desactiva la venta de Ayuda
Ctrl + B	Remueve todos los cables rotos del Diagrama de Bloques
Ctrl + E	Cambia entre el Panel Frontal y el Diagrama de Bloques.
Ctrl + Z	Deshace los cambios. (Disponible también en el menú edición)
Ctrl + R	Ejecuta el VI.
Ctrl + **.**	Aborta el VI.

Tabla 2.3 Combinación de teclas en LabVIEW.

2.1.10 Elementos de una programación típica.

A continuación se va ha describir algunos elementos típicos usados al momento de realizar un programa en LabVIEW.

2.1.10.1 Ciclos

En LabVIEW se utilizan principalmente los ciclos “While Loop” y el “*For Loop*” ó conocidos como el ciclo **Mientras**, y el ciclo **Para**. Ambos se encuentran localizados dentro de la paleta de Funciones en la subpaleta “*Structures*”. La diferencia principal entre estos dos ciclos es que el ciclo **Para** se ejecuta un número determinado de veces y el ciclo **Mientras** se ejecuta un número indeterminado de veces hasta que se cumpla cierta condición que le haga verdadera ó falsa, según se configure el while loop, generalmente se coloca un botón de parada booleano, para salir del ciclo.

2.1.10.1.1 Ciclo Mientras.

El ciclo while es usado en algunos lenguajes de programación basados en texto, conocido también como **Do While**, ó simplemente **Do**, **Repeat-Until loop**. El ciclo while mostrado en la **figura 2.12 (a)** ejecuta un sub-diagrama hasta que la condición de programa se haya cumplido. Se puede hacer que el ciclo “while” termine de ejecutarse ya sea cuando la condición sea verdadera o falsa, de la

siguiente forma, por defecto el ciclo while deja de ejecutarse cuando la terminal dependiente (), encontrada por defecto en la parte inferior derecha recibe un valor Verdadero ó "*True*", para que deje de ejecutarse cuando reciba un valor falso ó "*false*" se da un clic sobre dicha terminal y cambia la imagen del icono (). Esto se usa según la necesidad del programa. El icono ➔ indica el numero de iteraciones del ciclo while. El conteo de iteraciones siempre empieza en cero. Durante la primera iteración, la terminal de iteración regresa a cero.

2.1.10.1.2 Ciclo Para.

Como se había mencionado el ciclo **For** ó **Para**, mostrado en la **figura 2.12 (b)**, se ejecuta cierto numero de veces; el número de veces a ejecutarse esta determinado por la terminal **N**, por ejemplo si coloco ➔ ➔ entonces el numero de iteraciones () que se va ha realizar comienza desde 0, 1, 2, 3 y 4, es decir, 5 veces. Al igual que en el ciclo while, el conteo de iteraciones siempre empieza en cero. Durante la primera iteración, la terminal de iteración regresa a cero.

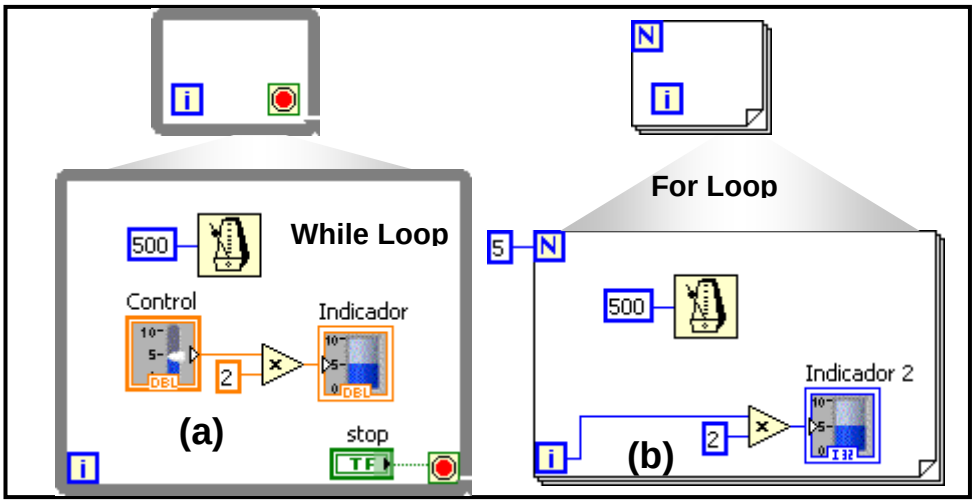


Figura 2.12. Ciclos. (a) Ciclo Mientras y (b) ciclo para.

2.1.10.2 Tipos de Funciones

2.1.10.2.1 VI Expreso.

A partir de la versión 7.0 de LabVIEW se introduce un nuevo tipo de sub-VIs llamados "*Express VIs*" ó VIs Expresos, **figura 2.13 (a)**, que se caracterizan por ser VIs interactivos, es decir, al ser colocados sobre el Diagrama de Bloques se

muestra un cuadro de Dialogo que permite al usuario personalizar la funcionalidad del mismo, luego de elegir las características del VI expreso LabVIEW genera un subVI con dichas características.

2.1.10.2.2 VI estándar.

Son VIs modulares y personalizadles mediante cableado. Son VIs que son usados dentro de un VI principal, este tipo de VIs también son conocidos como SubVIs, por lo tanto tienen un panel frontal y un diagrama de bloques.

2.1.10.2.3 Funciones.

Constituyen los elementos fundamentales de operación de LabVIEW, estos no tienen ni panel frontal ni diagrama de bloque. Son exclusivamente bloques de construcción.

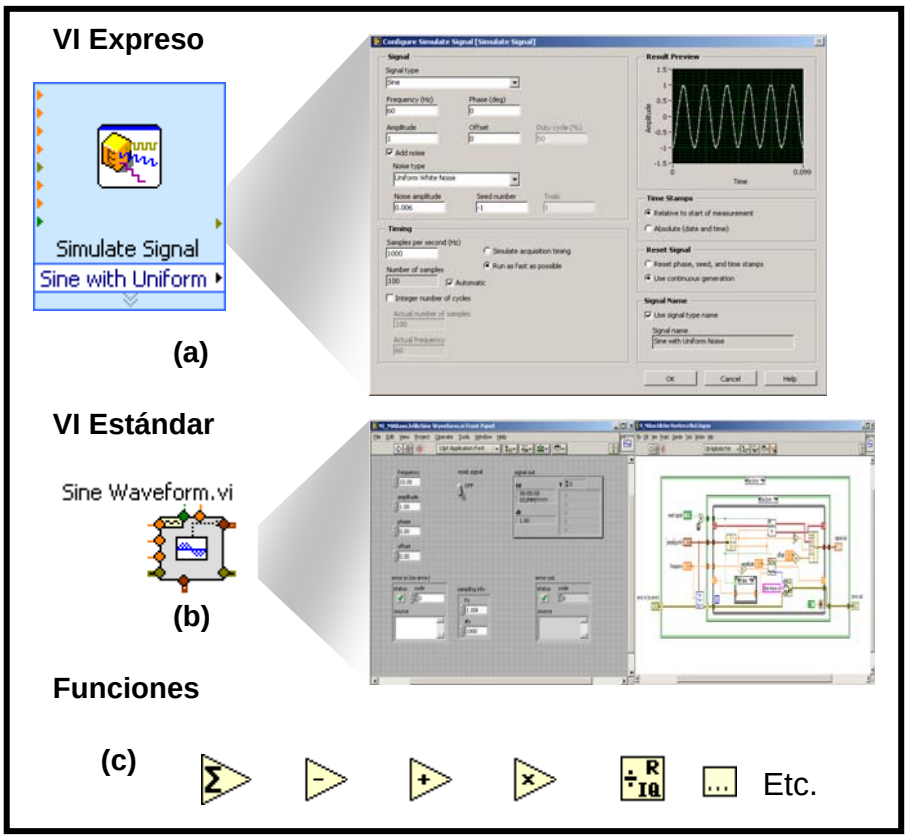


Figura 2.13 Tipos de funciones en LabVIEW.

2.1.10.2.4 Funciones que están disponibles.

LabVIEW incluye, dentro de sus librerías, varios cientos de funciones pre-construidas y que ayudan a la adquisición, análisis, y la presentación de los datos.

La mayoría de estas funciones están resumidas dentro de la paleta de funciones en la subpaleta "Express", **figura 2.14**. De una manera resumida, se tiene las siguientes funciones:

Entrada y Salida.

- Simulación de Datos y señales.
- Adquisición y generación de señales reales mediante módulos de adquisición y generación (DAQ).
- Asistente para instrumentos de entrada y salida (Serial & GPIB)
- ActiveX para comunicarse con otros programas.

Análisis:

- Procesamiento de señales.
- Estadísticas.
- Matemática y creación de formulas avanzadas.
- Soluciones en tiempo continuo.

Almacenamiento:

- De Archivos de entrada y salida.

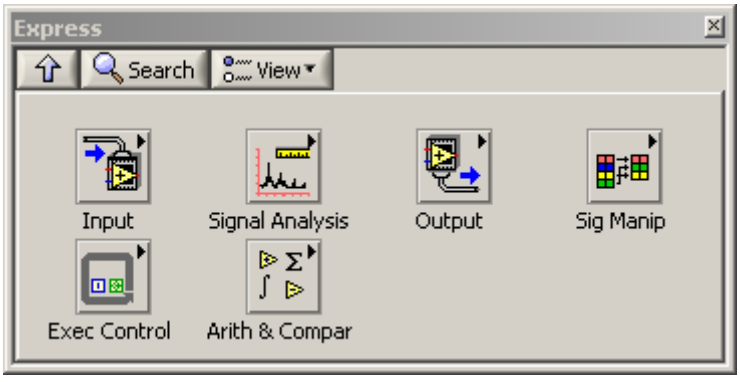


Figura 2.14 Paleta de Funciones "*Express*".

Adicionalmente, si es que se quiere construir funciones relacionadas a otras tareas, ya sea de Visión Artificial, programación de Módulos FPGAs, tareas de Supervisión Control y Registro de Datos, programación en Sistemas Embebidos, etc.… existen los llamados "*toolkits*" que son elementos adicionales al paquete de LabVIEW, y que dependiendo de la aplicación del diseñador puede adquirir uno u otro módulo ó un "*toolkit*" adicional según las necesidades del proyecto que se desee construir. Esto se debe más que nada para no tener elementos innecesarios para realizar una aplicación, y así economizar en elementos no útiles. Si se desea más información se puede consultar el la pagina Web http://www.ni.com/toolkits/.

2.1.11 Buscando VIs, Controles y Funciones.

Si es que se desea buscar funciones, controles, indicadores ó VIs, ya sea en la paleta de funciones ó en la paleta de control, según sea el caso, se hace un clic en la pestaña "*search*", a lo cual se muestra un cuadro en el que se pone el nombre de lo que se quiere buscar, en la **figura 2.15** se muestra un ejemplo en el que se ha buscado el VI PID. Las paletas están llenas de cientos de VIs, cuando se encuentra la función, VI, etc. que se buscaba se puede hacer doble clic para que nos muestre en que subpaleta se encuentra, ó si es que se desea colocar directamente se arrastra al diagrama de bloques y listo. Es de gran utilidad sobre todo en los casos en los que no se acuerda en que paleta se encuentra.

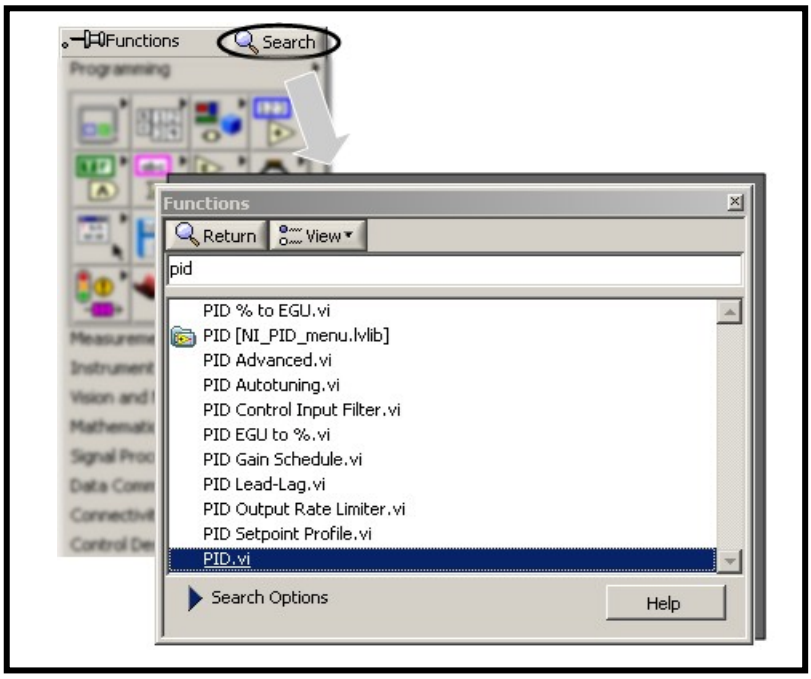


Figura 2.15 Buscando VIs, Controles, Funciones.

2.1.12 Como tomar decisiones en LabVIEW.

En LabVIEW existen dos posibilidades, la más conocida en la mayoría de lenguajes de programación basados en texto el llamado "***Case***" ó "**Caso**", llamado en LabVIEW **estructura de casos** ("*case structure*"), y la función **selección** ("***select***").

2.1.12.1 La estructura de casos.

Disponible dentro de la paleta de funciones, en la subpaleta "***structures***".
En la mayoría de veces el programador tiene que tomar decisiones ya sea para ejecutar uno u otro subdiagrama, o el que desee pero no de forma iterativa, esto se logra con éste tipo de estructura, en la que se pueden tener dos opciones: la **booleana**, que en este caso van a existir dos opciones Verdadero y Falso, **figura 2.16 (a)**, y la otra posibilidad es la **enumerativa**, que ofrece tomar tantas alternativas posibles según el valor que tome el selector, **figura 2.16 (b)**, máximo 214 casos. Aunque también puede tener la opción, además del booleano y enumerativo, de seleccionar casos de "*string*", ó cadena de caracteres, **figura 2.16 (c)**, también el valor que puede tener la terminal () de la estructura de casos puede ser un valor numérico tipo entero, **figura 2.16 (d)**.

2.1.12.2 Selección

Esta es una herramienta muy útil, parecida al "*case*" booleano solo que un poco más sencilla, consta de tres terminales de entrada y una salida, **figura 2.16 (e)**; las terminales de entrada son la terminal verdadera (**t**), la terminal falsa (**f**), y el seleccionador del caso (**s**) sea verdadero ó falso y que según esto, a la terminal de salida va ha salir el valor que se encuentre en la terminal **s**, es decir, si ese es verdadero a la terminal de salida va a salir el valor que esté en **t**, si **s** es falso, la terminal de salida tendrá el valor que tenga la terminal **f**.

Esta función está disponible en la subpaleta de funciones "**programming >> *comparison***".

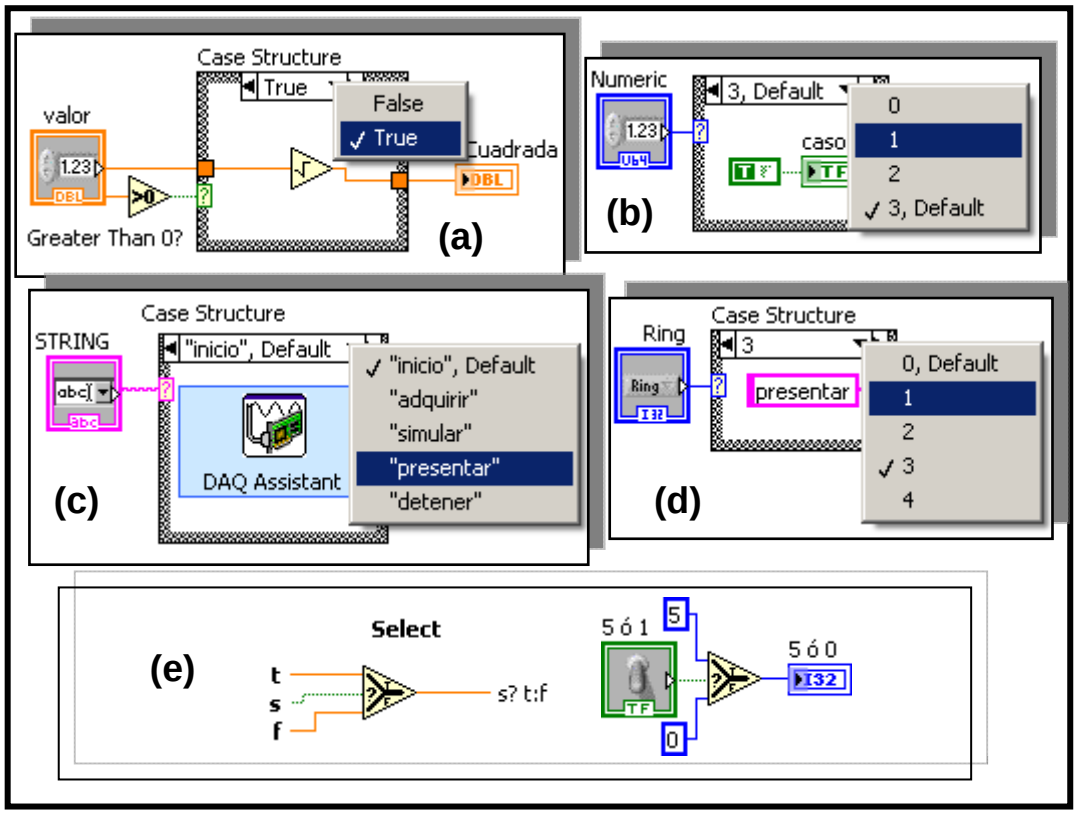


Figura 2.16 Como tomar decisiones en LabVIEW.

2.1.13 Manejo de Archivos.

Hoy, en la era de la información, es necesario respaldar ó grabar la información de un proceso dado para que en algún momento poder procesarla y así sacarla provecho. Para esto LabVIEW permite la grabación y lectura a y desde un archivo. LabVIEW crea ó usa los siguientes formatos de archivos: binarios, textos (ASCII), LVM que es una extensión que usa LabVIEW para leer y crear datos, y TDM que es un producto creado por la National Instruments.

Para manejar archivos de Entrada y Salida existe la subpaleta de funciones "***FILE I/O***", **figura 2.17 (a)**, que se encuentra dentro de la paleta de funciones "***programming***". La función principal de "FILE I/O" esta el de pasar datos desde y hacia archivos. Por ejemplo se puede:

- Abrir y cerrar archivos de datos.
- Lectura de datos desde y escribir datos a archivos.
- Lectura y escritura de datos a documentos en formato de hoja de calculo (spreadsheet), por ejemplo Excel.

- Mover y renombrar los archivos y directorios.
- Cambiar las características del archivo.
- Crear, modificar y leer archivos de configuración, etc.

También es posible en LabVIEW crear reportes de un VI, es decir, se puede crear un archivo en formato de Microsoft Word que contenga los gráficos del Panel Frontal, o de una señal que se desee, etc., o crear un reporte en Microsoft Excel, para esto se puede utilizar el toolkit "*NI LabVIEW Report Generation*" para Microsoft office. En la **figura 2.17 (b)**, se muestra la paleta de funciones de este toolkit.

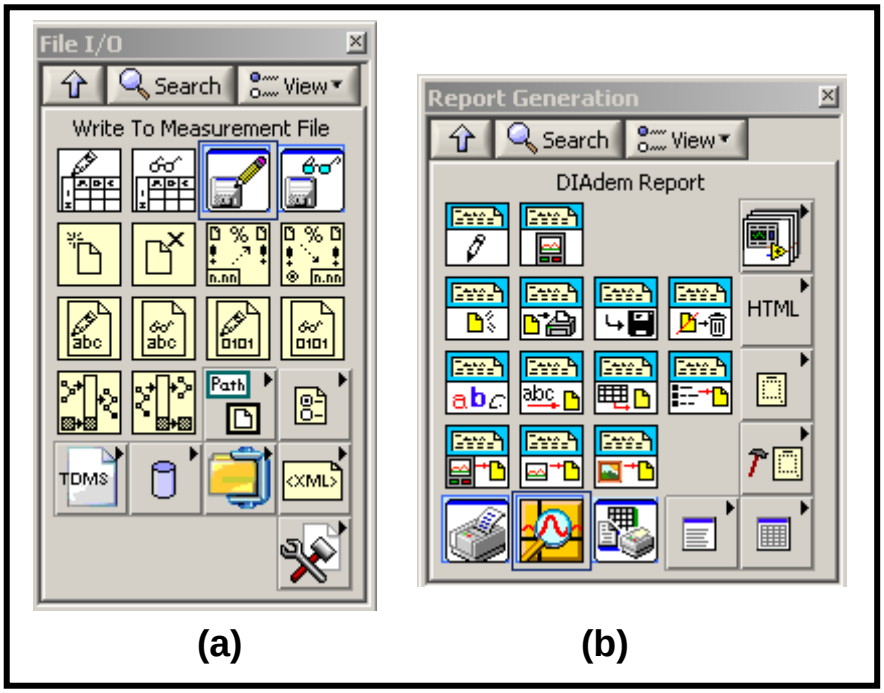


Figura 2.17 Manejo de Archivos. (a) Paleta de Funciones "FILE I/O", (b) Paleta de Funciones "Report Generation".

A continuación se va ha dar un ejemplo del uso del VI Express "*Write To Measurement File*", en el cual se va ha crear un archivo con una señal simulada, y luego se va ha leer esa misma señal, también se expondrá como se ven los datos en Microsoft Excel, en el cual se graficara usando las herramientas propias del Excel. En la **figura 2.18 (a)** y **(b)** se expone lo anteriormente dicho.

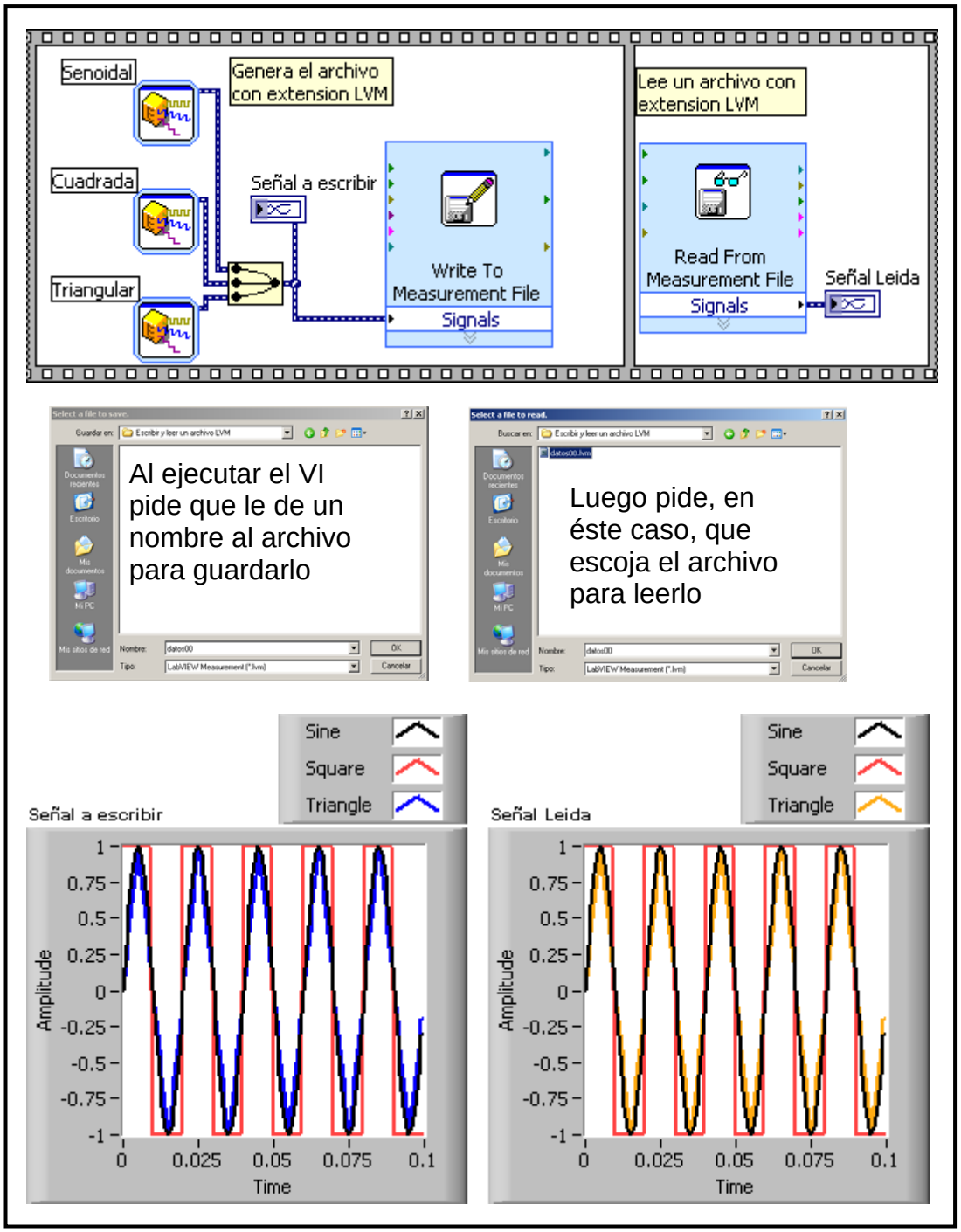


Figura 2.18 (a) Generación y lectura de un Archivo con extensión LVM.

Como se puede observar el diagrama de bloques en la **figura 2.18 (a)**, el VI Express "*Write To Measurement File*" es fácil de usar y tiene un alto nivel de abstracción.

En la **figura 2.18 (b)** se puede observar el archivo en Excel en el cual se ha graficado las señales generadas en LabVIEW, es así entonces que se tiene varias alternativas para analizar los datos almacenados.

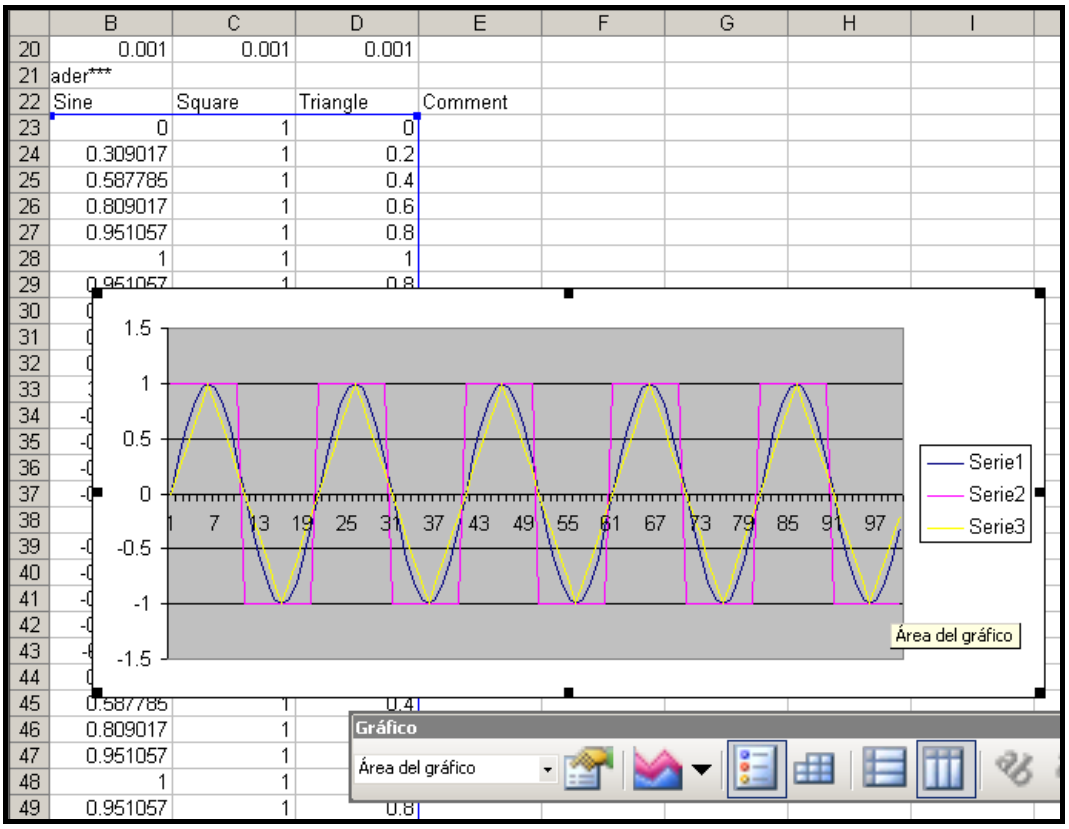


Figura 2.18 (b) Visualización en Excel del archivo generado por el VI Express "*Write To Measurement File*".

Claro que no es la única forma de guardar o leer un archivo, la forma que se vio anteriormente es la más rápida y fácil de usar, pero también se puede utilizar otras las otras herramientas de la paleta de Funciones "*FILE I/O*" (**figura 2.17(a)**). En la **figura 2.19** se muestra un modelo típico de programación que se aplica tanto para adquisición de datos, instrumentos de control, Archivos de Entrada y Salida y muchos otros esquemas de comunicación.

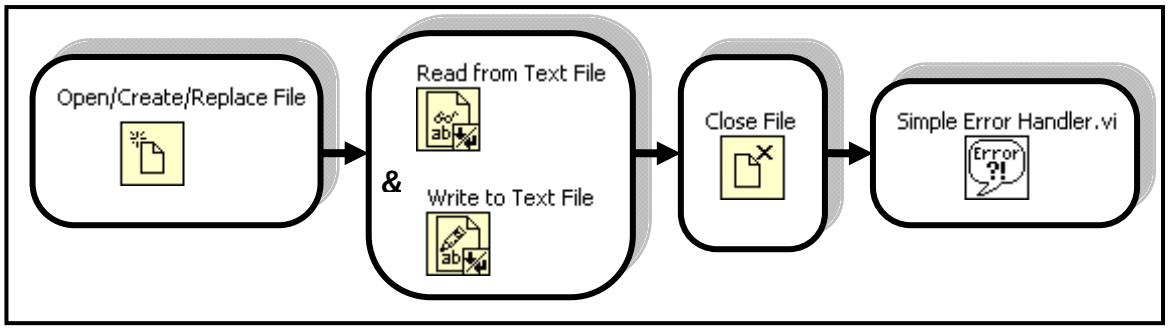


Figura 2.19 Modelo de programación para archivos de Entrada y Salida.

En este caso el modelo de programación lo construye el propio diseñador y/o programador de acuerdo a las necesidades del programa, que podría ser en este caso leer ó escribir un archivo cualesquiera.

También existe otra posibilidad de escribir un archivo, por ejemplo en formato de hoja de cálculo como en Excel, así mismo, utilizando las herramientas de la subpaleta de funciones "*File I/O*", "String", entre otras. Entonces las posibilidades se incrementan al poder realizar escrituras de datos en columnas y poderlas abrir en una hoja de cálculo de Excel para poder revisar los datos y tener un historial de un proceso dado, o de algún experimento que se esté realizando, etc. Entre las funciones principales más usadas para crear este tipo de archivos están las que se muestran en la **figura 2.20**.

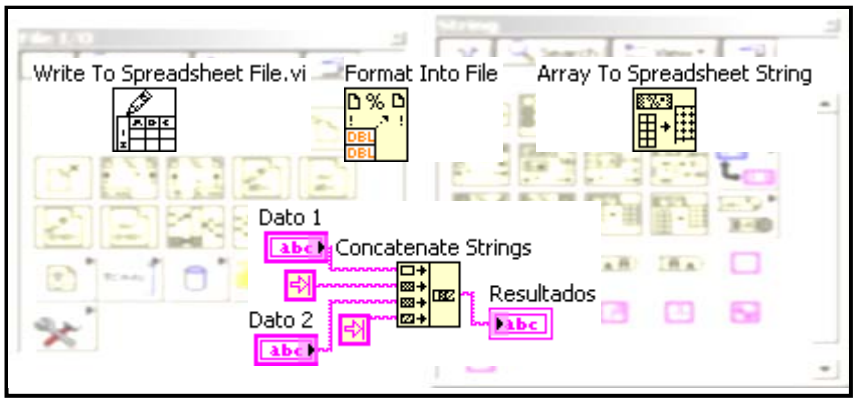


Figura 2.20 Algunos elementos para grabar archivos en formato de hoja de cálculo.

2.1.14 Presentación de los resultados.

La parte más importante para el usuario: la presentación de de los datos, análisis, resultados, etc. En LabVIEW existen diversas formas de presentar los resultados, obviamente dependiendo de que tipo de datos se trate y de cómo le convenga al usuario final observar su proceso ó práctica, etc. Como se ha venido mencionando, todo lo que es la interfaz entre el usuario y la máquina se va ha observar en la ventana del panel frontal.

Para visualizar los resultados en LabVIEW existen diversos tipos de controles e indicadores, ya sean estos, numéricos, booleanas, gráficos, etc. A continuación se van ha mencionar algunos de estos controles e indicadores.

2.1.14.1 Tipos de Controles e Indicadores disponibles en LabVIEW.

Dentro de la paleta de control, disponible en el panel frontal, existen varios tipos de controles e indicadores que le permiten al usuario interactuar con el programa; los controles son los encargados de ingresar los datos al programa y los indicadores por su parte son los que van ha visualizar los resultados de dicho programa. Para mostrar la paleta de control se hace clic derecho sobre una parte libre del panel frontal. En la **figura 2.21** se puede observar la paleta de control.

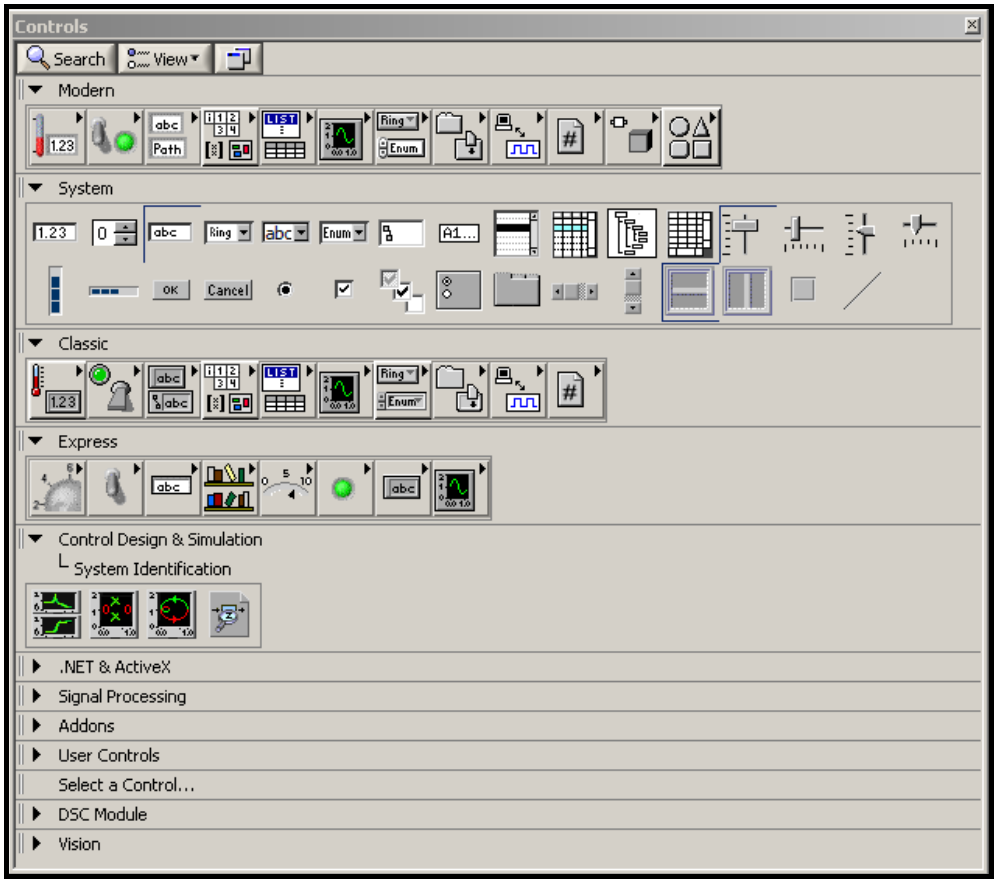


Figura 2.21 Paleta de Control de LabVIEW.

Además de los controles e indicadores disponibles por defecto al instalar LabVIEW, al momento de instalar módulos y/o toolkits adicionales se instalan también los controles e indicadores relacionados con dichos módulos y/o toolkits, por ejemplo el módulo para Visión Artificial, el Módulo DSC, el toolkit para identificación de sistemas, etc. …

Entre los controles e indicadores más usados, **figura 2.22**, se tiene:

- Datos numéricos: sirve para ingresar ó mostrar valores numéricos.

- Datos Booleanos: se pueden encontrar Leds, botones, switch, etc.

- Arreglos de datos y matrices: dentro de ésta categoría se tiene indicadores y controles tipo “array”, es decir arreglos de datos, también indicadores gráficos como el “chart” y el “graph”, y otros más avanzados como: “XY Graph”, “Intensity Graph”, “3D Graph”, etc.

- También se tiene una paleta que sirve para decorar el instrumento virtual, esta paleta no contiene ningún tipo de control y/o indicador, por lo tanto no va ha tener una terminal en el diagrama de bloques.

- Otros tipos más de controles e indicadores, se tiene por ejemplo los tipo “string” o cadenas de caracteres, cuadros de textos, indicadores y controles para colocar imágenes, etc.…

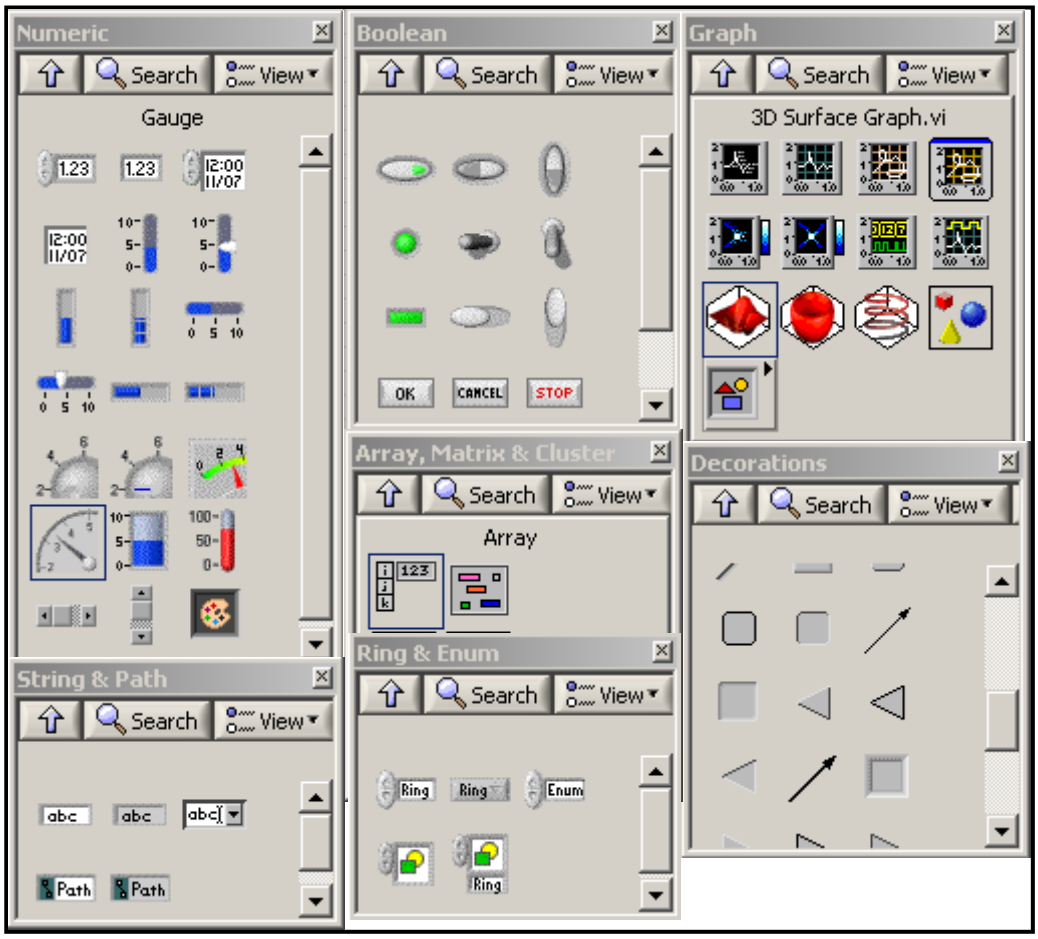


Figura 2.22 Diferentes tipos de controles e indicadores.

A continuación se van ha mencionar algunos de los indicadores gráficos más comunes y usados.

2.1.14.1.1 Indicadores gráficos.

En muchas ocasiones es necesario, para una mayor comprensión de los resultados obtenidos, representarlos de forma gráfica. Es por eso que LabVIEW incorpora diferentes tipos de indicadores gráficos dentro de la subpaleta de control “Graph”, divididos, por lo general, en dos categorías los indicadores gráficos tipo “*Chart*” y los indicadores tipo “Graph”.

2.1.14.1.1.1 Indicador Gráfico “Waveform Chart”.

Es un tipo especial de indicador numérico que puede mostrar una o más graficas, con la característica principal que retiene en su pantalla un cierto número de datos que es definido previamente por el usuario. Los datos en un “Waveform Chart” se grafican punto por punto, o si es un array, array por array. En la **figura 2.23** se muestra un ejemplo de dicho indicador. En éste tipo de indicador los datos nuevos se añaden a lado de los ya existentes, de forma que se puedan comparar entre ellos.

En los indicadores gráficos se pueden cambiar los valores de valor máximo y mínimo de las ordenadas y de las abscisas, así como también, se puede dejar que LabVIEW automáticamente detecte la escala y la ajuste, también, como todo elemento que se coloque sobre el panel frontal, tiene sus diferentes propiedades, tanto dentro del panel frontal como en el diagrama de bloques.

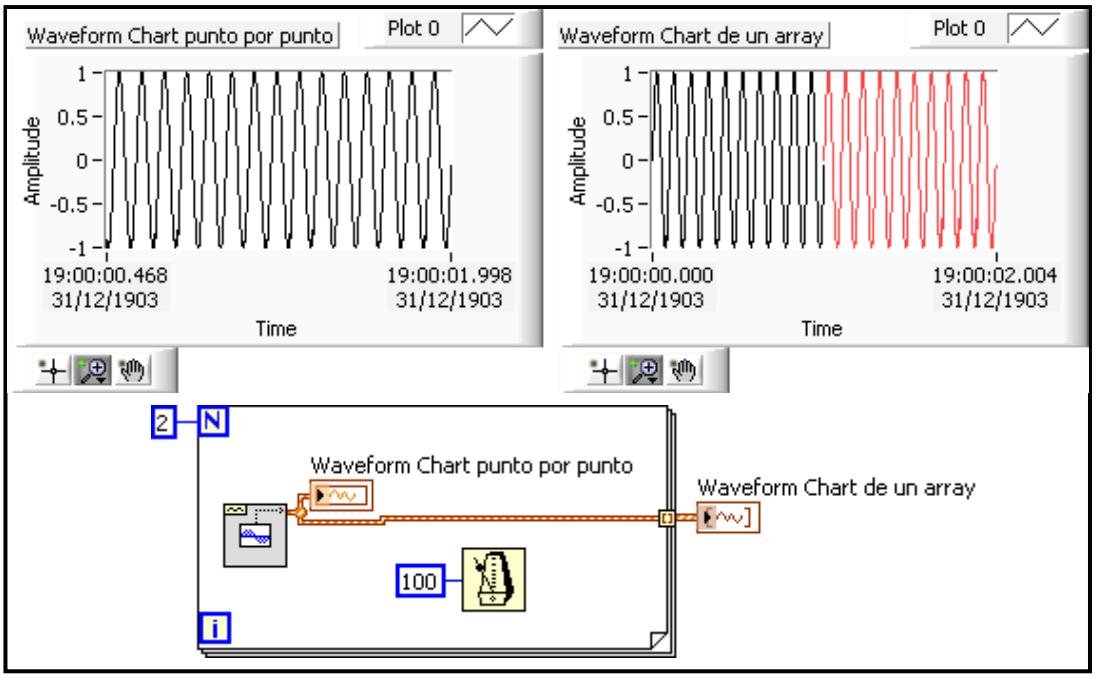


Figura 2.23 Indicador gráfico tipo "*Waveform Chart*"

2.1.14.1.1.2 Indicador Gráfico "Waveform Graph"

Este tipo de indicador gráfico a diferencia de un "*chart*" que graficaba punto por punto los datos, un "*waveform graph*" grafica todos los puntos de los datos a la vez, es decir, grafica los datos como un arreglo ó "*array*". Otra diferencia entre el "*chart*" y el "*graph*" también es que el "*waveform graph*" los datos antiguos se pierden y se colocan los nuevos datos, al contrario de el "*chart*" que se añadían a continuación del anterior.

Todos los controles e indicadores permiten personalizarlos a nuestro gusto y conveniencia. Generalmente los indicadores gráficos son usados para mostrar una gran cantidad de información. También se pueden mostrar múltiples graficas dentro de un mismo indicador gráfico, para eso se puede usar la función "*Merge Signals*", para unir todas las señales, algo así como un multiplexor.

También como se había mencionado en párrafos anteriores, cada control e indicador tiene sus propiedades, para encontrarlas se hace clic derecho sobre el elemento que se desea ver, y se elige "*properties*" ó propiedades. En éste caso para el "*waveform graph*" se puede elegir por ejemplo diferentes propiedades para las diferentes gráficas, como puede ser color, ancho de la línea, tipo de línea, tipo de gráfica, la escala, opciones de cursor, etc.…

Los indicadores gráficos también permiten exportar una imagen del indicador, por ejemplo para colocar en algún informe técnico, etc. En la **figura 2.24** se ilustra un ejemplo de este indicador.

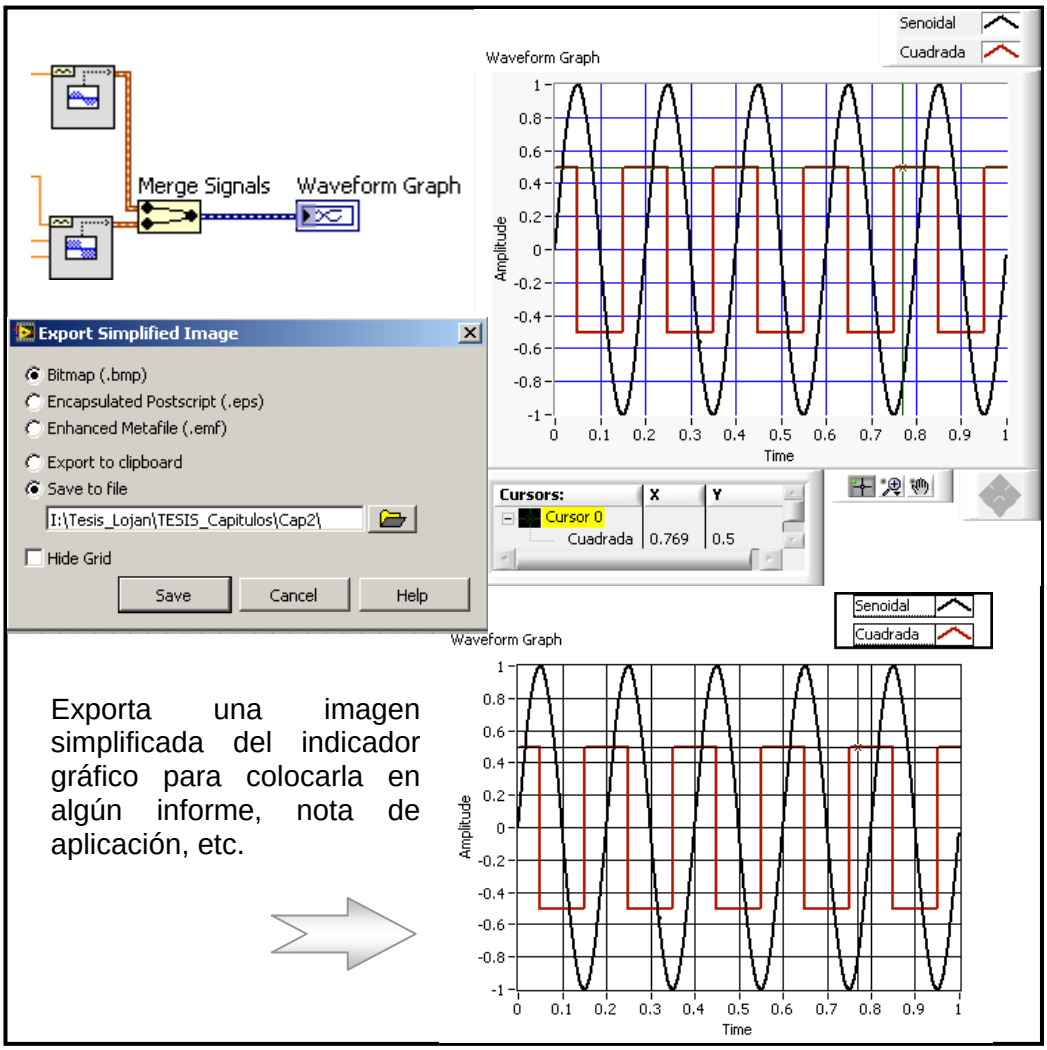


Figura 2.24 Indicador gráfico tipo "*Waveform Graph*"

2.1.15 Matemática textual en LabVIEW.

Una de las herramientas importantes y fundamentales dentro de la educación, la ingeniería y otras afines es la matemática y todo lo que ella conlleva, es así que LabVIEW incorpora poderosas herramientas de desarrollo para realizar aplicaciones, simulaciones, etc. en un solo entorno pudiéndose está combinar con otras aplicaciones para desarrollar un instrumento virtual muy bueno y de alto desempeño. LabVIEW soporta algunos softwares importantes de matemática como son: ***Mathscript script node***, ***Mathematica***, ***Maple***, ***MathSoft***, ***Xmath*** y el software más conocido y usado en nuestro medio ***MATLAB***®. Entonces se puede desarrollar

algoritmos, explorar conceptos relacionados con la matemática, analizar resultados, etc. y todo en un solo ambiente.

2.1.15.1 Desarrollo de algoritmos matemáticos con "MathScript Node"

Como es conocido LabVIEW es un ambiente de programación gráfico, pero incorpora también herramientas para desarrollar algoritmos de programación en el lenguaje tradicional, que es el basado en la programación textual, una de éstas herramientas es el llamado "*MathScript Node*", **figura 2.25 (a)**, que permite realizar algoritmos y ecuaciones matemáticas tal y como si se estuviera programando en el software MATLAB, y además es compatible, generalmente, con los archivos de extensión "**m**" que se generan en MATLAB. "***MathScript Node***" se encuentra dentro de la subpaleta de funciones "***programming >> structures***" ó también dentro de la subpaleta de funciones "***Mathematics >> Scripts & Formulas***".

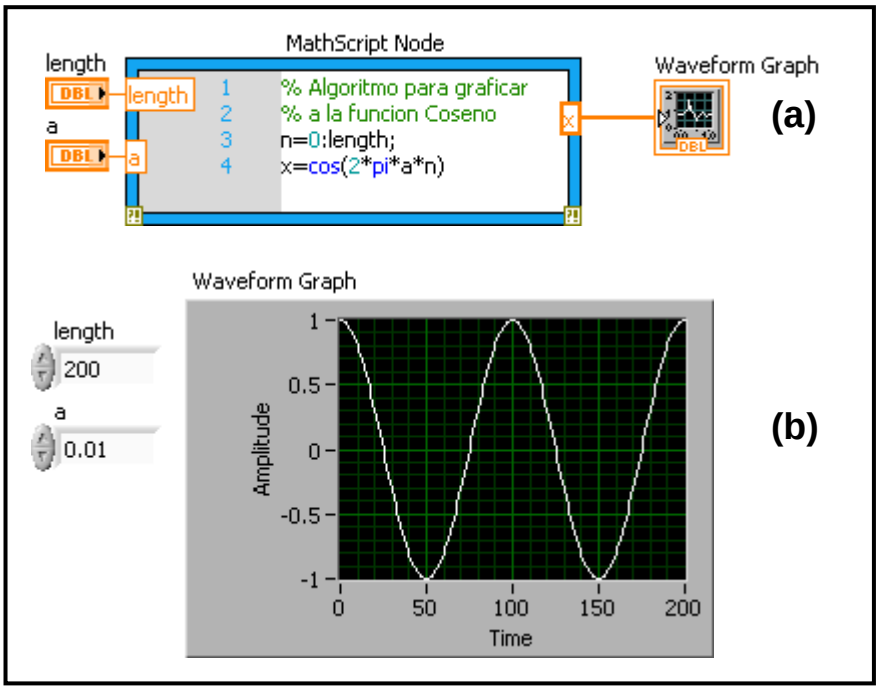


Figura 2.25 (a) algoritmo para graficar la función Coseno usando la herramienta "*MathScript Node*" (b) Panel frontal, muestra la grafica de la función coseno.

También LabVIEW ha incorporado una ventana interactiva denominada "**Math Script**", la cual es una interfaz muy parecida a la que se tiene comúnmente en el software MATLAB, para mostrar esta ventana se va al menú "***Tools >> MathScript***

Windows…", en la **figura 2.25 (c)**, se muestra dicha ventana y un ejemplo de la misma.

Si se desea obtener más ayuda de esta ventana se puede escribir en la ventana de comandos la palabra "help".

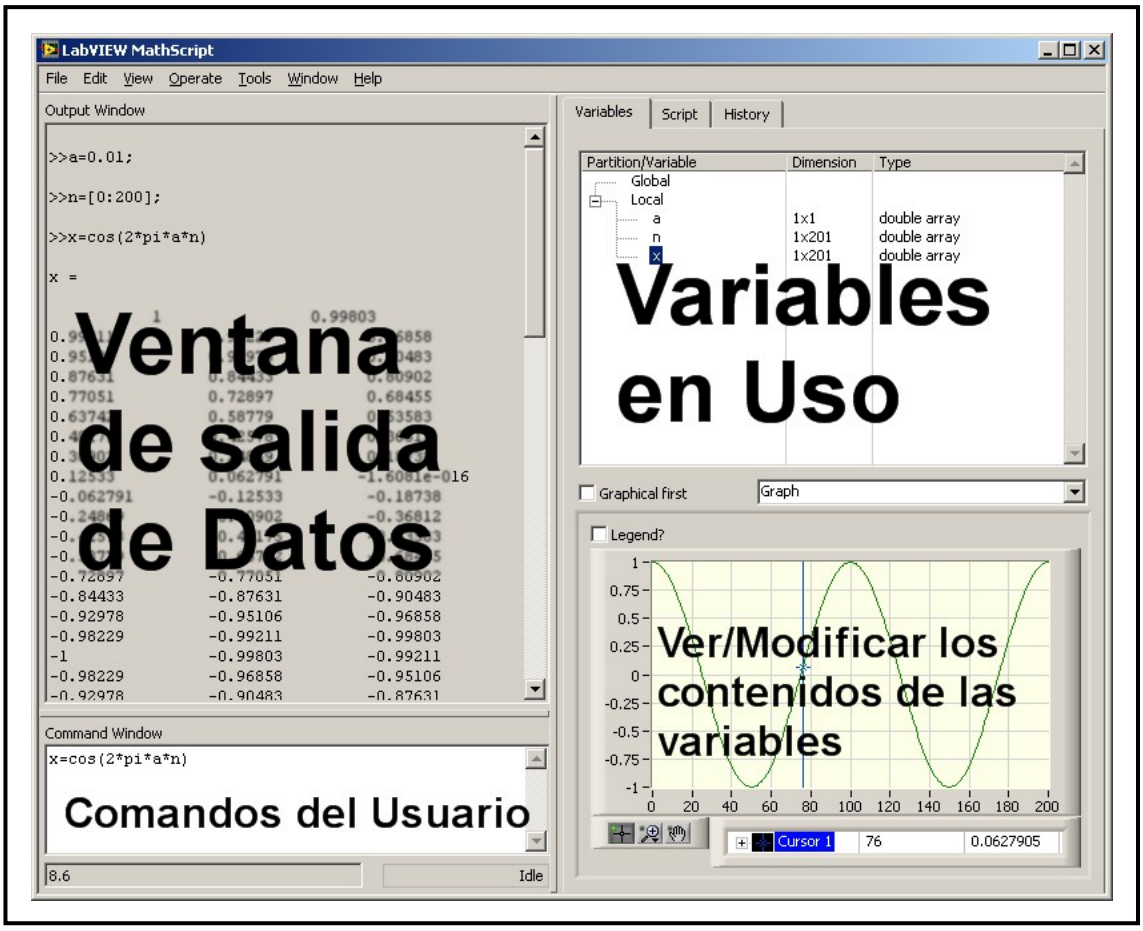


Figura 2.25 (c) Ventana de LabVIEW MathScript.

Cabe destacar que se pueden hacer aplicaciones importantes tal y como lo permite el software MATLAB, y adicionándole otras funciones de LabVIEW permitiendo crear un VI muy interesante y rápido de desarrollar.

2.1.16 Resumen de los tipos de Datos que se encuentran en LabVIEW.

Los tipos de datos disponibles en LabVIEW se diferencian básicamente por los colores que tienen los cables, por ejemplo los datos tipo booleanos tienen un color verde olivo, los números de doble precisión (double), es decir, los que manejan punto flotante tienen un color anaranjado, y así sucesivamente. En la **figura 2.26** se muestra un resumen de los tipos de datos en LabVIEW.

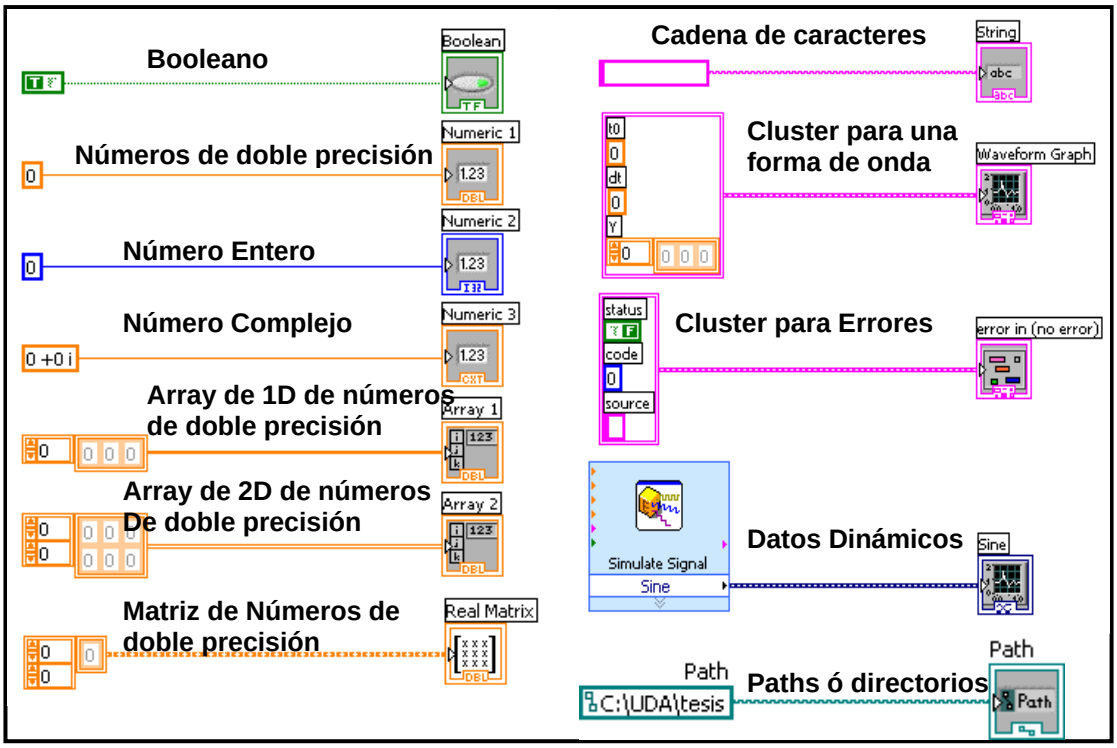


Figura 2.26 Resumen de los tipos de Datos en LabVIEW.

Algunas definiciones:

- **Array**: un array o arreglo, es un conjunto de datos, siempre del mismo tipo, ya sean booleanos, numéricos, Springs, etc. y estos pueden ser controles, indicadores ó constantes. Un array está formado por elementos y dimensiones. Los elementos son los datos que forman el array; mientras las dimensiones, especifican el tamaño en largo, ancho y profundidad del array. Un array puede tener una ó más dimensiones y $2^{(31)} - 1$ elementos posibles por cada dimensión y de memoria permitida.

- **Cluster**: un cluster a diferencia de un array permite asociar a elementos de diferentes tipos, ya sean booleanos, numéricos y/o Springs, etc. es decir, se podría comparar a esta función como si fuese un conjunto de cables como los que hay en la red de telefonía, ó como un bus de datos, donde cada alambre dentro del conjunto de cables representa un elemento diferente del cluster.

2.1.17 Modularidad en LabVIEW.

En muchas ocasiones en el desarrollo de un programa es necesario utilizar cierta parte del programa más de una vez y para eso en los lenguajes tradicionales basados en texto se crean las llamadas subrutinas, que lo que hacen principalmente es realizar cierto subprograma cuando es llamado, en LabVIEW también existe la posibilidad de realizar esta tarea, en este caso se crean los llamados SubVIs, que se podría decir que son VIs utilizados dentro de otro VI.

Para crear los SubVIs por lo general existen dos posibilidades, la primera es a partir del mismo VI en el cual deseamos cierta parte del programa para usarla en otra parte, para eso seleccionamos la sección del programa que deseamos reutilizarla y luego en el menú "***Edit***" seleccionamos "***Create SubVI***" y se crea un icono que contiene esa parte del programa tal y como las funciones estándares que vienen hechas en LabVIEW, es decir tendrá sus entradas y salidas de datos, en este caso siguiendo este procedimiento LabVIEW automáticamente configura el numero de controles e indicadores para el SubVI. En la **figura 2.27 (a)** muestra un ejemplo de lo anteriormente mencionado; la línea entrecortada es la parte que se desea crear el SubVI. Si se doble clic sobre el SubVI creado se abre el panel frontal y el diagrama de bloques de dicho SubVI en el cual se puede observar el segmento del programa que se había seleccionado, luego se guarda en un lugar que sea de fácil localización, ya que cada vez que se abra el VI principal va ha ser como si abriera dicho VI más el SubVI.

Si es que se desea utilizar este mismo SubVI pero en otro VI solamente se abre el SubVI y se arrastra el icono del SubVI al Diagrama de Bloques del nuevo VI a trabajar.

La otra forma de crear un SubVI, es programarlo de acuerdo a nuestras necesidades y luego utilizando algunas herramientas propias de LabVIEW como el Icono y el Panel Conector, en inglés "***Icon and Connector Panel***", que básicamente sirven para asociar las entradas y salidas con los correspondientes controles e indicadores respectivamente.

Entonces una vez que se haya terminado de construir el VI se puede utilizar dichas herramientas anteriormente mencionadas y asociar las entradas y salidas para que luego éste VI pueda ser usado como SubVI.

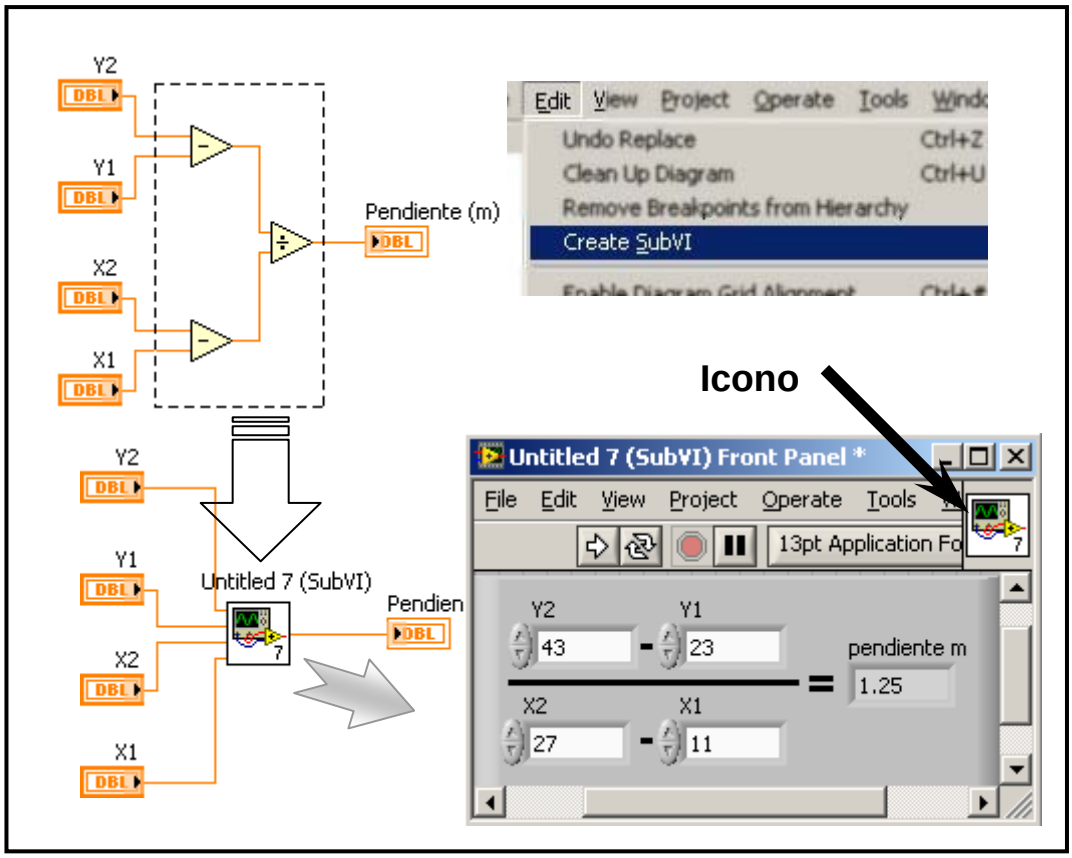


Figura 2.27 (a) Creación de un SubVI usando "Create SubVI"

En la **figura 2.27 (b)** se muestra un ejemplo de la forma de construir un SubVI, para mostrar el panel conector se hace clic derecho sobre el icono del VI, y se selecciona "*Show Connector*", a lo que aparecerá el panel conector estándar, generalmente para mantener ordenado las entradas y las salidas se asocia del centro a la izquierda como entradas y del centro hacia la derecha como salidas. Si es que se desea agregar más entradas y/o salidas se hace clic derecho sobre el panel conector, y se selecciona "Patterns" en el cual se tienen algunos paneles conectores, y simplemente se hace clic derecho y se selecciona "Add Terminal", para agregar una terminal más.

Se recomienda el uso de los SubVIs para hacer los programas más cortos.

Una vez que se muestra el panel conector se acerca el puntero del ratón a dicho panel y se hace clic donde se desea que vaya el control o indicador y luego se hace clic dentro del panel frontal en el control y/o indicador al que se desea asociar.

También hay la posibilidad de personalizar el icono, es decir, colocar o crear una imagen a nuestro gusto, principalmente para diferenciar de los iconos de los SubVIs

propios de LabVIEW. Para esto se hace doble clic sobre el icono y aparecerá la ventana del "***Icon Editor***" en donde se puede realizar lo anteriormente dicho.

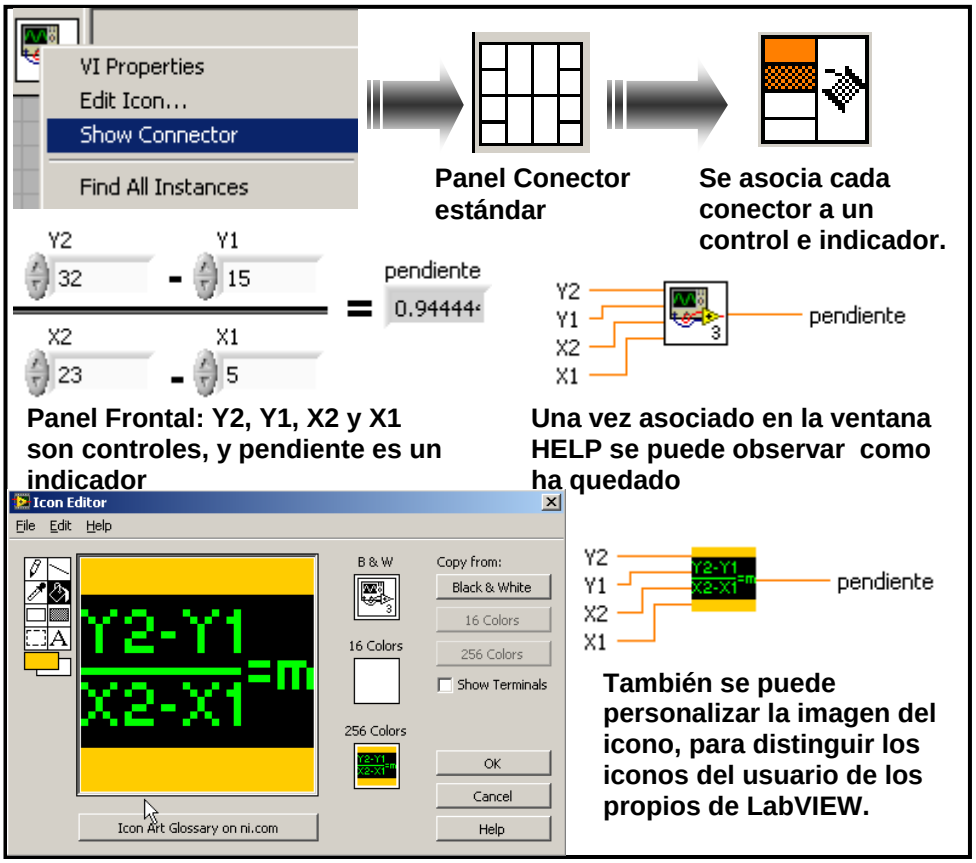


Figura 2.27 (b) Creación de SubVIs y edición del icono.

2.1.18 Variables Locales.

Uno de los elementos más usados dentro de la programación son las variables, las cuales ayudan a colocar ó generar datos para ser usados dentro de uno u otro subprograma, sin la necesidad de estar escribiendo el valor del dato cada vez sino que solamente se coloca el nombre de la variable y se le asigna el valor una sola vez produciéndose el cambio del valor donde dicha variable se encuentre.

Las variables locales en éste caso sirven para trabajar solo dentro el mismo programa, no siendo así con las variables globales, y las variables compartidas ó "Shared Variables".

Las variables locales son utilizadas para pasar datos entre lazos ó también para evitar cableados largos, ya que dicha variable se puede usar como indicador ó control desde una o más localizaciones dentro el programa.
Las variables locales ayudan de alguna manera a ahorrar y rompen con el paradigma del flujo de datos.

En la **figura 2.28** se muestra un ejemplo del uso de ésta variable.

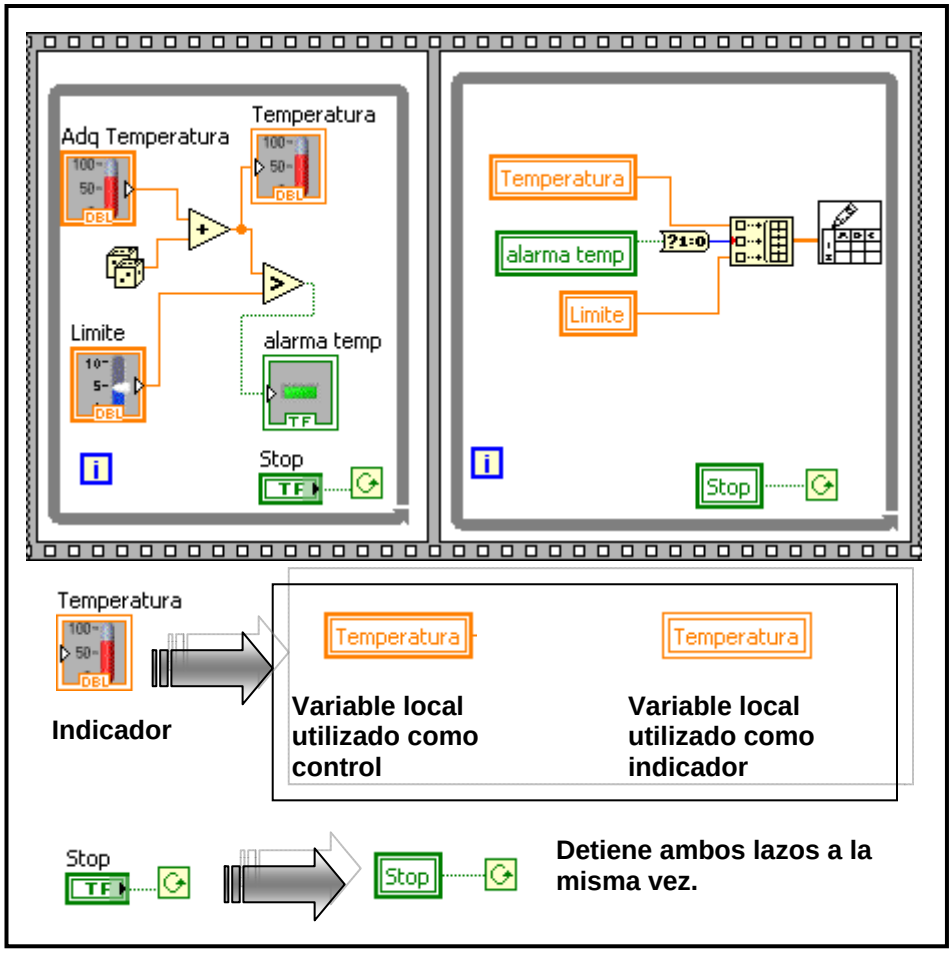


Figura 2.28 Ejemplo del uso de las variables locales.

Para crear una variable local primero se debe tener indicadores o controles, luego se puede ir a la paleta de funciones "***Programming >> Structures***" y se selecciona "local" (LOCAL); luego se da un clic sobre el icono que aparece en el diagrama de bloques ([?]) y se selecciona a que control ó indicador se desea asociar. Otra forma también de crear una variable local se hace clic derecho sobre el control o indicador, y se selecciona "***create >> Local Variable***".

2.1.19 Desarrollando programas grandes con LabVIEW.

Hay ocasiones en las que la mayoría de programas realizados en LabVIEW son grandes y existe la necesidad de herramientas para hacer más fácil el desarrollo del programa ó VI. Para eso a continuación se mencionan tres herramientas importantes para realizar dicha tarea. Cabe recalcar que estas herramientas están disponibles a partir de la versión 8.0X de LabVIEW.

2.1.19.1 Ventana de navegación.

La ventana de navegación ó "***Navigation Window***", **figura 2.29**, permite navegar por el panel frontal ó por el diagrama de bloques. Para mostrar esta ventana se va al menú "***View >> Navigation Window***" ó mediante las teclas de acceso rápido "Ctrl + Shift + N". Se puede usar ésta ventana para navegar especialmente por el diagrama de bloques en las ocasiones en las que el programa sea demasiado grande. Para mostrar una región determinada se da un clic sobre dicha ventana o si es que se desea navegar sobre ella se mantiene presionado el puntero del ratón y se navega por ella.

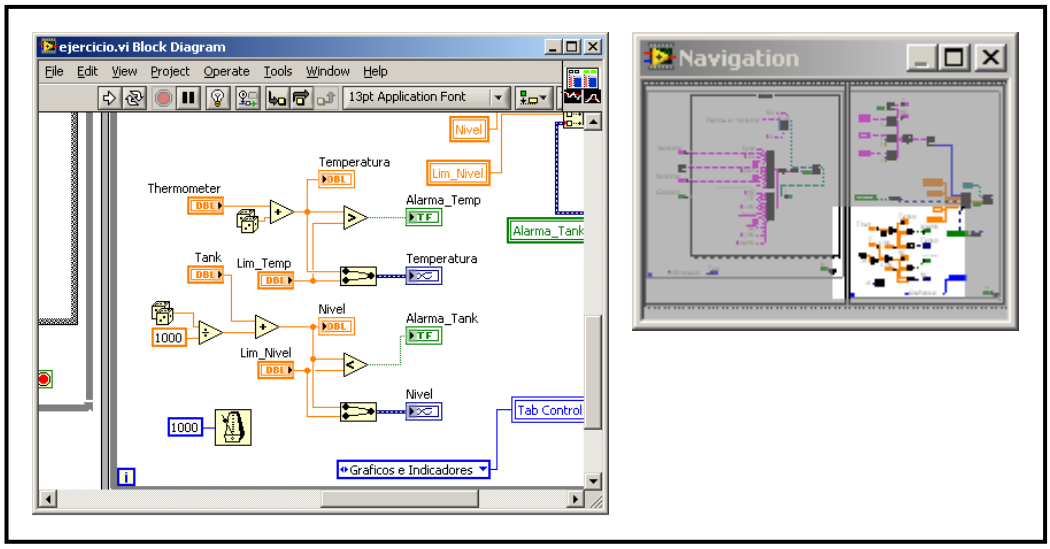


Figura 2.29 Ventana de navegación de LabVIEW.

2.1.19.2 Creación de proyectos en LabVIEW.

Esta es quizás una de las herramientas más importantes de LabVIEW en el momento de crear programas grandes, ya que permite ordenar y agrupar los VIs, administrar Hardware y sus entradas y salidas, administrar los VIs para diferentes tareas, construir librerías y ejecutables, administrar grandes aplicaciones de LabVIEW, etc.

Además, LabVIEW Project permite administrar archivos en otras extensiones como la de Excel, Word, etc.

Para sacar la ventana de LabVIEW Project, existen por lo menos tres posibilidades, la primera: cuando se inicia LabVIEW seleccionar "***Empty Project***", la segunda podría ser también desde la misma ventana anterior seleccionar "***File >> New Project***", tercera opción: puede ser desde el panel frontal o desde el diagrama de bloques seleccionamos el menú "***File >> New…***" a lo que aparecerá la ventana "***New***" en la que se pueden seleccionar varias alternativas entre ellas un proyecto en blanco "***Empty Project***". En la **figura 2.30** se muestra lo anteriormente mencionado.

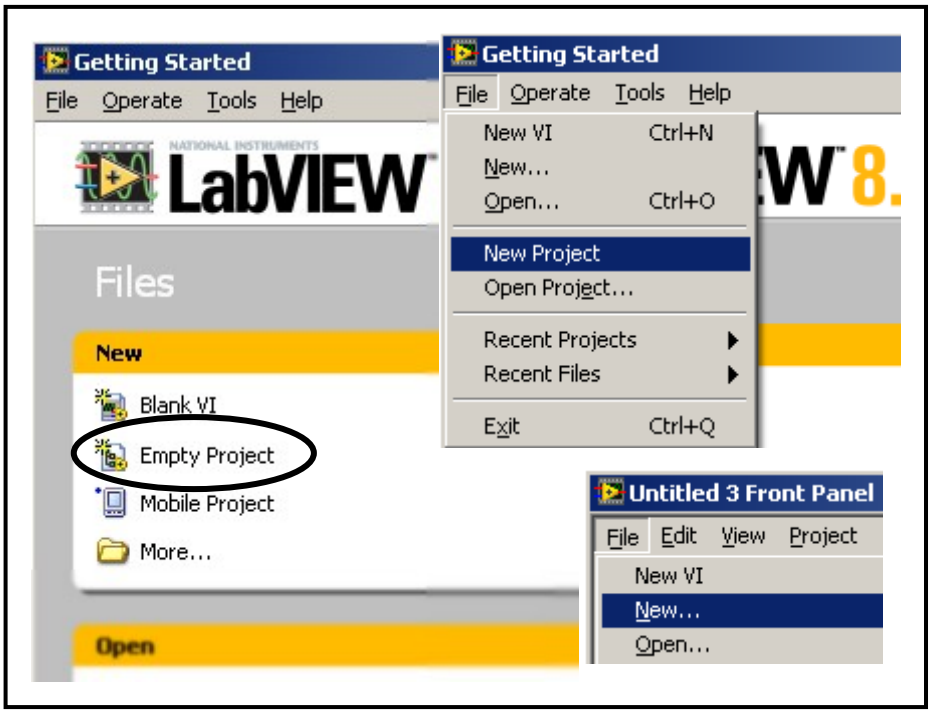


Figura 2.30 Formas de selección para crear un nuevo proyecto.

La ventana de LabVIEW Project se muestra en la **figura 2.31 (a)**, en la **figura 2.31 (b)** se puede observar un proyecto creado, en la que se encuentran archivos en otras extensiones como la de Word, Excel, Power Point, RTF, permitiendo de esta manera realizar una mejor administración de la información y ser más ordenados en nuestros proyectos ya que se les puede documentar.

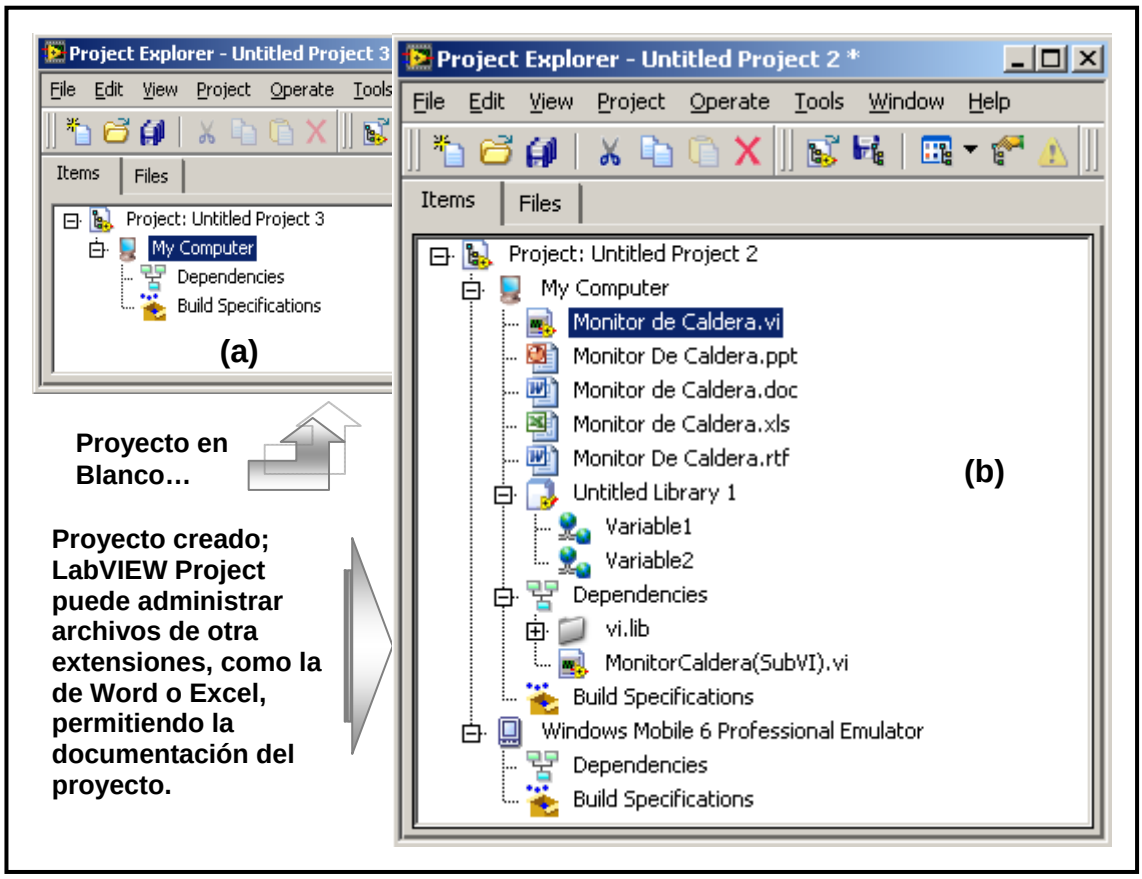


Figura 2.31 Ventana LabVIEW Project; (a) Proyecto en Blanco, (b) Proyecto creado.

Con LabVIEW Project también se pueden crear y/o descargar archivos en otros dispositivos en los cuales pueda ejecutarse el VI, estos dispositivos ó máquinas son conocidas mas comúnmente como "*targets*". National Instruments ofrece algunos dispositivos en los cuales se puede descargar el VI y funcionar de forma independiente de la PC y en tiempo real, en tal caso la PC quedaría como un sistema únicamente de supervisión y en algunos casos hasta de control. Entre los dispositivos más comunes que tiene National Instruments están: FPGA, RT (Real Time), etc. También hay como descargar el VI en una PDA o en un dispositivo móvil que sea compatible con Windows y con LabVIEW. Para todos estos targets y

aplicando la filosofía modular de LabVIEW se tienen diferentes softwares que contienen los diferentes módulos para cada aplicación en específico.

Cuando se guarda un proyecto LabVIEW crea un archivo con la extensión "**.lvproj**", la cual incluye configuraciones de la información, crear información del proyecto, indicar referencias a archivos dentro del proyecto, entre otras.

2.1.19.3 Variables compartidas (Shared Variables)

Existen aplicaciones en las cuales es necesario comunicar datos entre computadoras u otros dispositivos para lo cual LabVIEW ofrece una elegante forma de hacerlo, con las llamadas variables compartidas ó "***shared variables***", soportado por lo general con máquinas que soporten Windows.

Aunque también las shared variables pueden usarse dentro de una misma máquina o PC como se hacia con las variables locales para pasar datos entre VIs, lazos, etc.

Con las shared variables, los VIs en diferentes máquinas o redes pueden leerse desde ó escribirse hacia las variables sin la necesidad de estar programando una red ó algo más complejo para tener información de una máquina a otra.

Para crear un variable compartida ó shared variable se puede hacer, por lo común, la forma más correcta es utilizando el LabVIEW Project, para esto se hace un clic derecho sobre "***My Computer >> New >> Variable***", **figura 2.32 (a)**, a lo que se presentará el cuadro de dialogo "***Shared Variable Properties***", **figura 2.32 (b)** ó propiedades de la variable compartida, en el cual se podrá elegir algunas opciones como: el **Tipo de dato**, es decir si es booleano, entero, de doble precisión, un array, etc., también se podrá elegir el **Tipo de variable**, que son dos, "***Network-Published***" y "***Single Process***" por ejemplo si se desea que sea una variable para compartirla en la red ("*Network-Published*"), para que pueda se accesible desde una computadora o un target remoto, o también puede ser "*single prowess*" en éste caso solo podrá ser accesible dentro de la misma PC o máquina en la cual se esté programando, entre otras opciones más que se podrán encontrar para ésta variable.

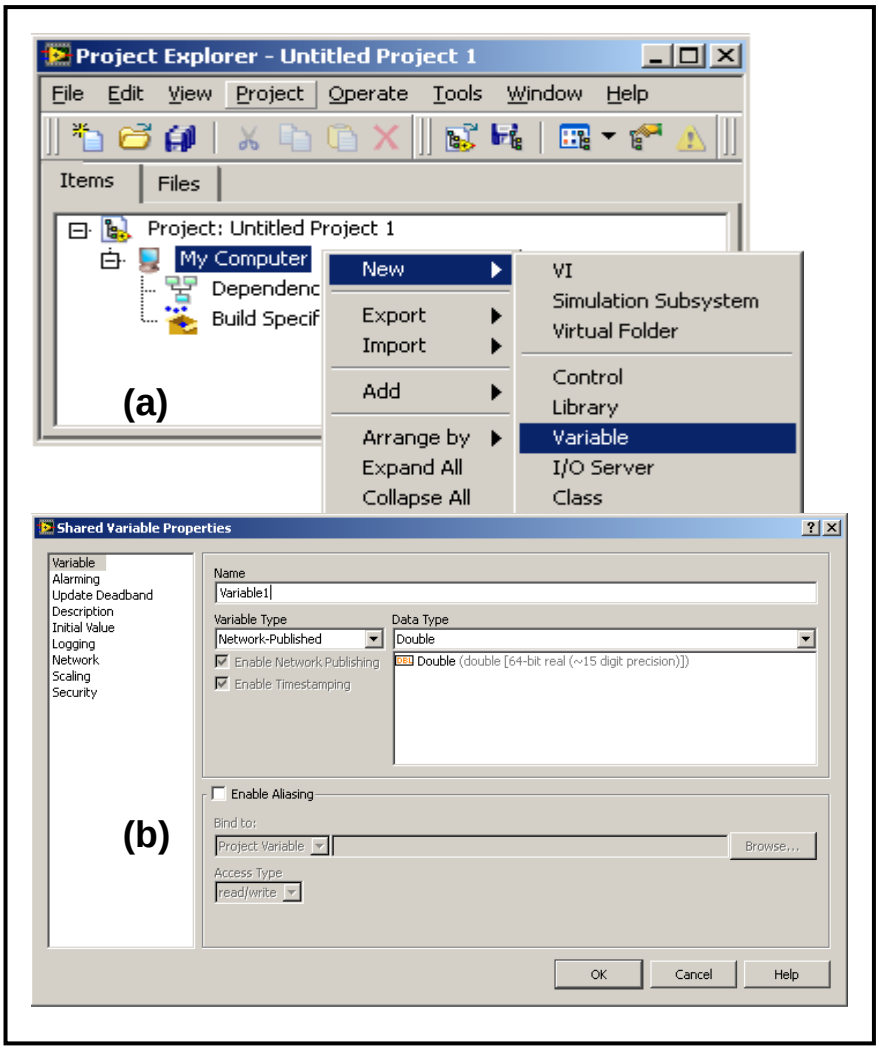


Figura 2.32 (a) Crear una variable compartida, (b) Cuadro de dialogo de las propiedades de la variable compartida a crear.

En la **figura 2.33** se ilustra un ejemplo simple de el uso de esta variable, simulando la conexión entre dos VIs de dos computadoras, en la cual la computadora 2 va ha encender un Led de la computadora 1.

Para colocar la variable compartida solamente se arrastra de la ventana de LabVIEW Project al diagrama de bloques.

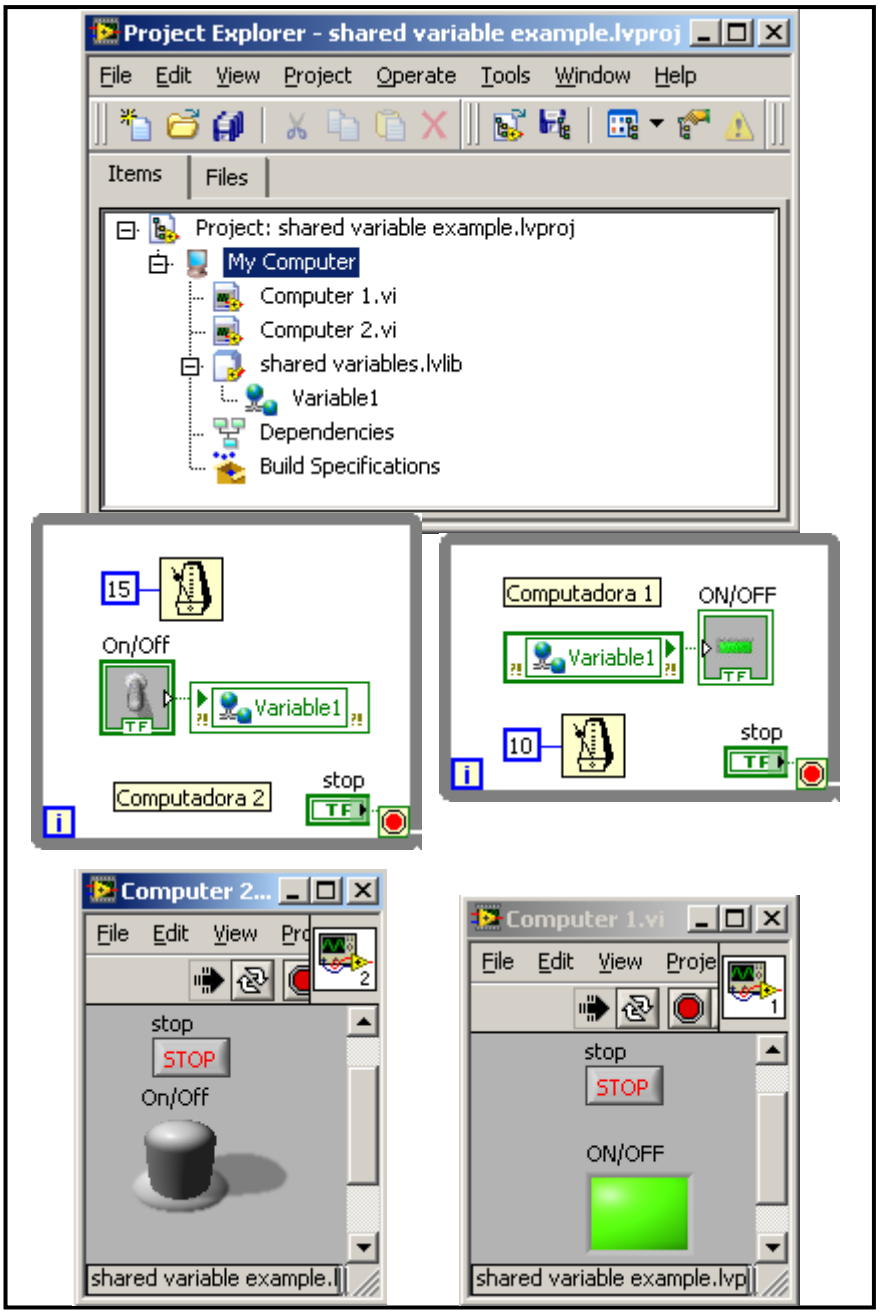


Figura 2.33 Ejemplo de uso de la variable compartida.

Hasta aquí se ha visto y mencionado las partes más importantes de LabVIEW, su entorno de programación, algunos tips y herramientas indispensables para la construcción de una aplicación y creación de un VI.

A continuación se dará paso a la segunda parte de éste capitulo que trata un tema importante dentro de la automatización industrial, la ingeniería y en algunos campos científicos: La Visión Artificial basada en PC.

2.2 VISIÓN ARTIFICIAL BASADA EN PC CON LabVIEW.

2.2.1 Introducción a la Visión Artificial Basada en PC

La Visión Artificial por Computador es un subcampo de la inteligencia artificial. La Visión Artificial tiene muchos usos dentro de algunos campos como en: la automatización Industrial, la Robótica y en el área de Medicina sobre todo en equipamientos, y en implantes a humanos.

En este apartado se va ha estudiar algunos conceptos básicos relacionados a la Visión Artificial en LabVIEW de una manera general. Sobre todo se enfocará al uso del paquete "***NI VISIÓN 8.6 DEVELOPMENT MODULE***" de LabVIEW, que es el software que permite la adquisición y análisis de imágenes, como se verá más adelante, además, debido a que en éste presente tesina se usará una cámara Web ó "WebCam" se mencionaran algunos aspectos que se tendrán en cuenta para la adquisición y análisis de las imágenes, así como las pocas ventajas y desventajas que se tiene al usar este tipo de cámaras.

Previamente se van ha realizar algunas definiciones y elementos que son base para el diseño y programación para crear una aplicación de Visión Artificial.

2.2.2 Definiciones en Visión Artificial.

2.2.2.1 Definición de Imagen:

Una imagen puede ser definida matemáticamente como una función bidimensional:

$$f(x, y)$$

Donde ***x*** & ***y*** son coordenadas espaciales (en un plano), y ***f*** en cualquier par de coordenadas es la intensidad o nivel de gris de la imagen en esa coordenada.

Se podría decir que una imagen también es una distribución espacial de intensidad lumínica en una escena, ya que la imagen depende mucho de la luminosidad que haya en el lugar donde se desea adquirir la imagen.

2.2.2.2 Imagen Digital:

Cuando ***x*** & ***y*** los valores de ***f*** son todas cantidades finitas y discretas, decimos que la imagen es digital.

Una imagen digital se compone de un número finito de elementos, cada uno con un lugar y valor específico. Estos elementos son llamados pixeles.

Se podría decir entonces que una imagen digital es la distribución espacial discreta de energía lumínica en una escena formada de $n \times m$ píxeles.

2.2.2.3 Píxel

Viene del acrónimo inglés "***Picture Element***", en español: elemento de imagen, y se define como la menor unidad homogénea en color que forma parte de una imagen digital, ya sea ésta una fotografía, video (fotograma) ó un gráfico. En la **figura 2.34** se puede observar una imagen digital en la que se ha realizado un zoom sobre una parte de la imagen y se pueden observar los pixeles de dicha imagen.

2.2.2.4 Parámetros de una Imagen Digital.

La vista es nuestro sentido más avanzado, y no es sorprendente que las imágenes jueguen un papel importante en la percepción humana.

Los parámetros principales de una imagen digital son:

Tamaño: el tamaño de una imagen digital se mide en pixeles, tanto el ancho como el alto de la imagen. Por ejemplo: 240x322 pixeles.

Resolución: la resolución tiene que ver con el tamaño de cada píxel, por ejemplo una imagen de 28 pixeles/centímetro, estaría diciendo que en un centímetro hay 28 pixeles. Mientras más alta la resolución de mejor calidad es la imagen.

Profundidad: la profundidad por su parte dice la cantidad de escalones de color que tiene la imagen digital. Por ejemplo si se dice que una imagen tiene 8 bits mono, se está hablando que está en escala de grises, donde $2^8 = 256$, (0 a 255).

Donde el 0= negro y el 255= blanco, de esa manera se definen los pixeles de una imagen.

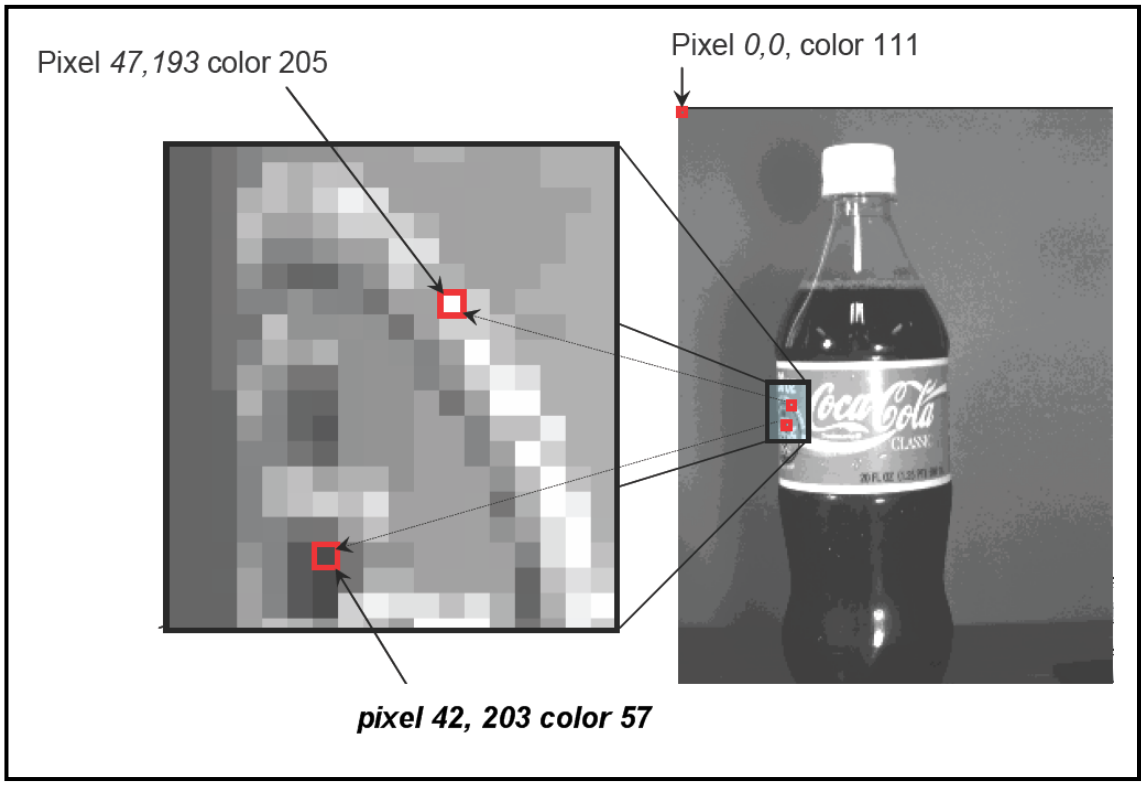


Figura 2.34 Ejemplo de una imagen digital y del píxel.

(ORELLANA, Visión Artificial basada en PC, 2009)

2.2.2.5 Elementos en la Visión Artificial.

En un sistema de visión por computadora, siempre están presentes los siguientes elementos:

- **Objeto**: es lo que se quiere procesar, es sobre lo que se tiene que trabajar.
- **Iluminación**: se había dicho que una imagen es la distribución espacial de luz, por lo tanto si es que no se tiene una buena iluminación no va ha ser posible procesar bien la imagen.
- **Cámara**: Es el elemento que se va ha utilizar para capturar la imagen y pasarla al computador.
- **Computador**: va ha ser el encargado de tener los puertos y/o tarjetas de adquisición necesarios para obtener la imagen capturada por la cámara.

- **Procesamiento y análisis**: se refiere a que software se va ha utilizar, en este caso LabVIEW, para el análisis, filtrado, procesado, etc. de la imagen digital.

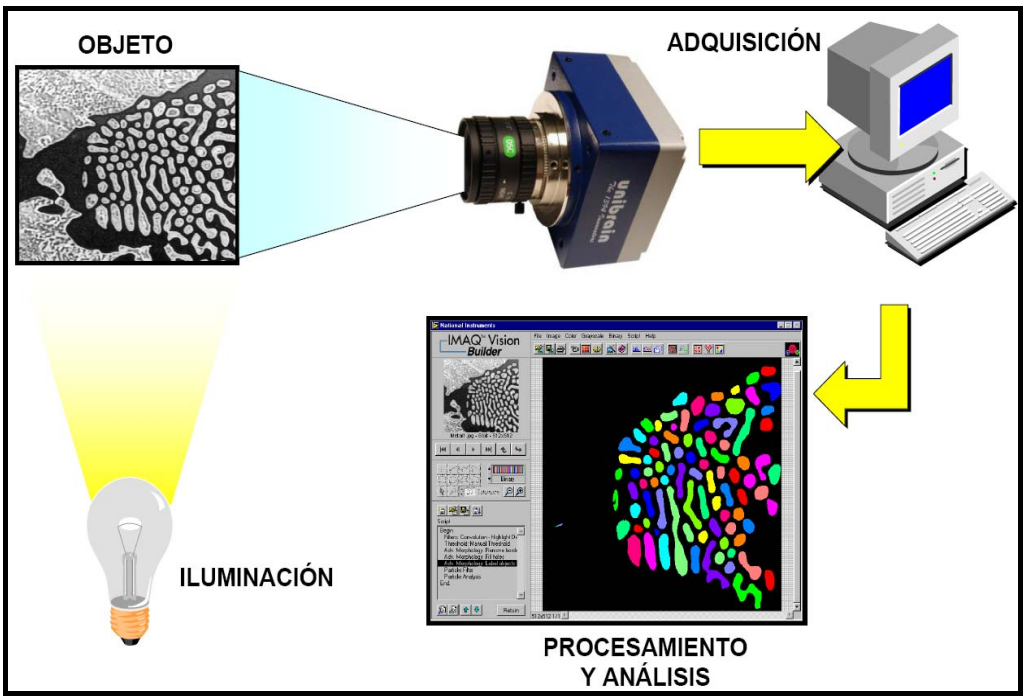


Figura 2.35 Elementos presentes en la adquisición de imágenes en la Visión Artificial.

(ORELLANA, Visión Artificial basada en PC, 2009)

2.2.2.5.1 Consideraciones del Objeto.

El objeto es el elemento sobre el cual se va ha trabajar, medir y/o analizar, por lo tanto no está sujeto a cambios ni modificaciones, refiriéndose a que no se puede escoger ó cambiar a la conveniencia del programador ó diseñador.

Lo que sí se puede cambiar y/o modificar son en el resto de componentes como la cámara, el lente, iluminación, etc. de acuerdo al objeto y al ambiente en el que se va ha trabajar.

2.2.2.5.2 Consideraciones sobre la iluminación.

La iluminación es un factor **extremadamente** importante, ya que, la luz transfiere información de un objeto sobre un detector (cámara). La luz tomada en el detector debe proveer la suficiente información para distinguir las características de la Región de Interés (**ROI**= ***Region Of Interest***).

Es importante seleccionar una correcta iluminación para resaltar sobre las otras características, es decir, realzar el contraste para separar lo que se quiere de lo que no y también para suprimir las señales no deseables como la iluminación del ambiente, polvo, vibración, orientación del objeto, etc. También se usa para homogenizar la luz sobre el campo de Visión (**FOV**= ***Field Of View***). Es importante también para eliminar reflejos en el objeto. Con todas estas características de la iluminación y así tener una imagen que sea claramente analizada y procesada.

2.2.2.5.2.1 Características de la superficie según la iluminación.

- **Difusa**: éste tipo de superficie refleja la luz en todos los ángulos.

Figura 2.36 (a) Superficie difusa.

- **Especular**: refleja la luz según al ángulo de incidencia.

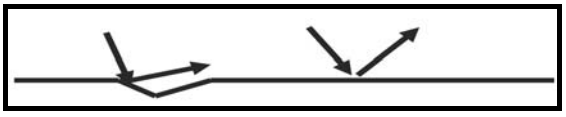


Figura 2.36 (b) Superficie especular.

- **Absorsiva**: éste tipo de superficie no refleja luz.

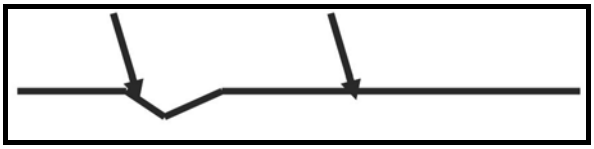


Figura 2.36 (c) superficie absorsiva.

2.2.2.5.2.2 Técnicas de iluminación

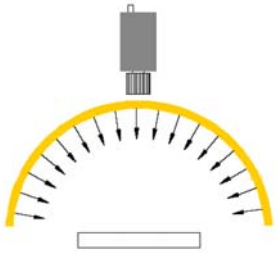
Existen algunas técnicas que permiten, según sea el caso, realizar de una mejor manera la toma de la imagen, a continuación se detallan algunas técnicas más utilizadas:

- **Puntual o Directa:**

- Iluminación de una sola fuente.
- Produce sombra.
- Tiene buena eficiencia.
- Es de fácil montaje.
- Mejor definición de bordes y texturas.
- No aplicable en superficies muy reflectivas.

Figura 2.37 (a)
Iluminación Puntual[1]

- **Difusa:**



- Iluminación desde múltiples direcciones.
- No produce sombras y limita reflejos especulares.
- Disimula texturas y borronea bordes.
- Difícil montaje.

Figura 2.37 (b)
Iluminación difusa[1]

- **Front Light: (Figura 2.37 (c))**

- Luz desde la semiesfera de la cámara, se clasifican en dos tipos:
  - **Bright Field**: El FOV aparece claro y con pocas sombras.
  - **Dark Field:** Solo se ven las irregularidades de la superficie.

- **Back Light: (Figura 2.37 (d))**

- Luz desde la semiesfera opuesta a la cámara.
- Siluetas.
- Objetos traslúcidos.
- **Contraluz** facilita análisis de contornos.

- **Luz Estructurada:**

Éste tipo de iluminación es usada para obtener información espacial del objeto, por ejemplo. Para medir distancia se utiliza una luz estructurada tipo punto, para medir altura se utiliza una línea, para medir superficies se utiliza una grilla. En la **figura 2.38** se muestra un ejemplo de este tipo de iluminación.

[1] Fuente: GONZÁLES. Conferencia de Visión Artificial. 2008.

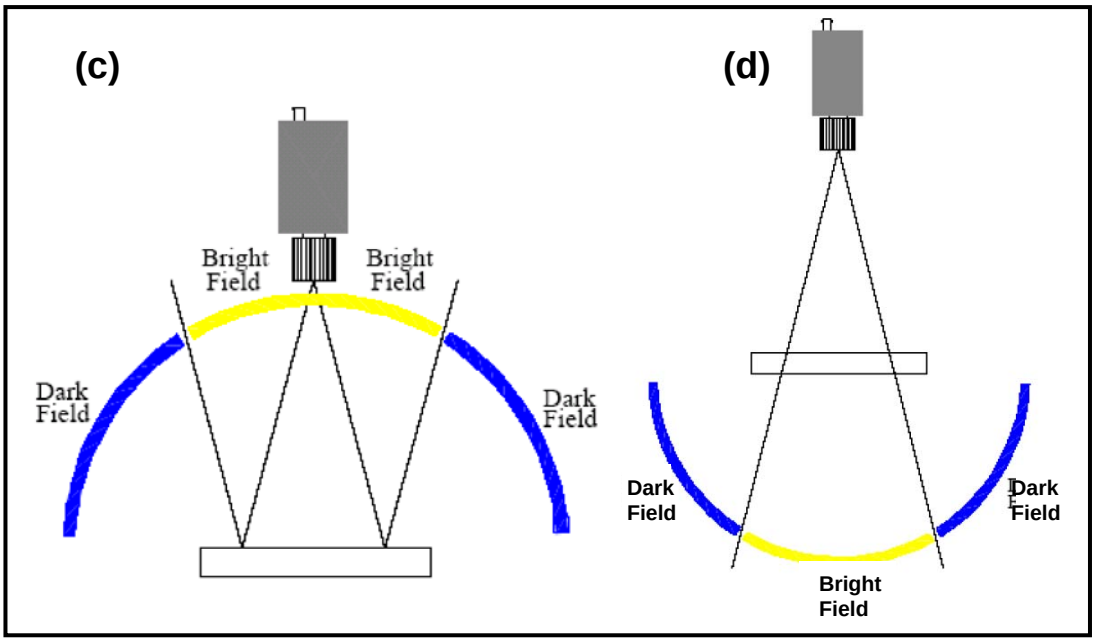


Figura 2.37 (c) Iluminación Front Light (d) Iluminación Back Light.

(GONZÁLES, Conferencia de Visión Artificial, 2008)

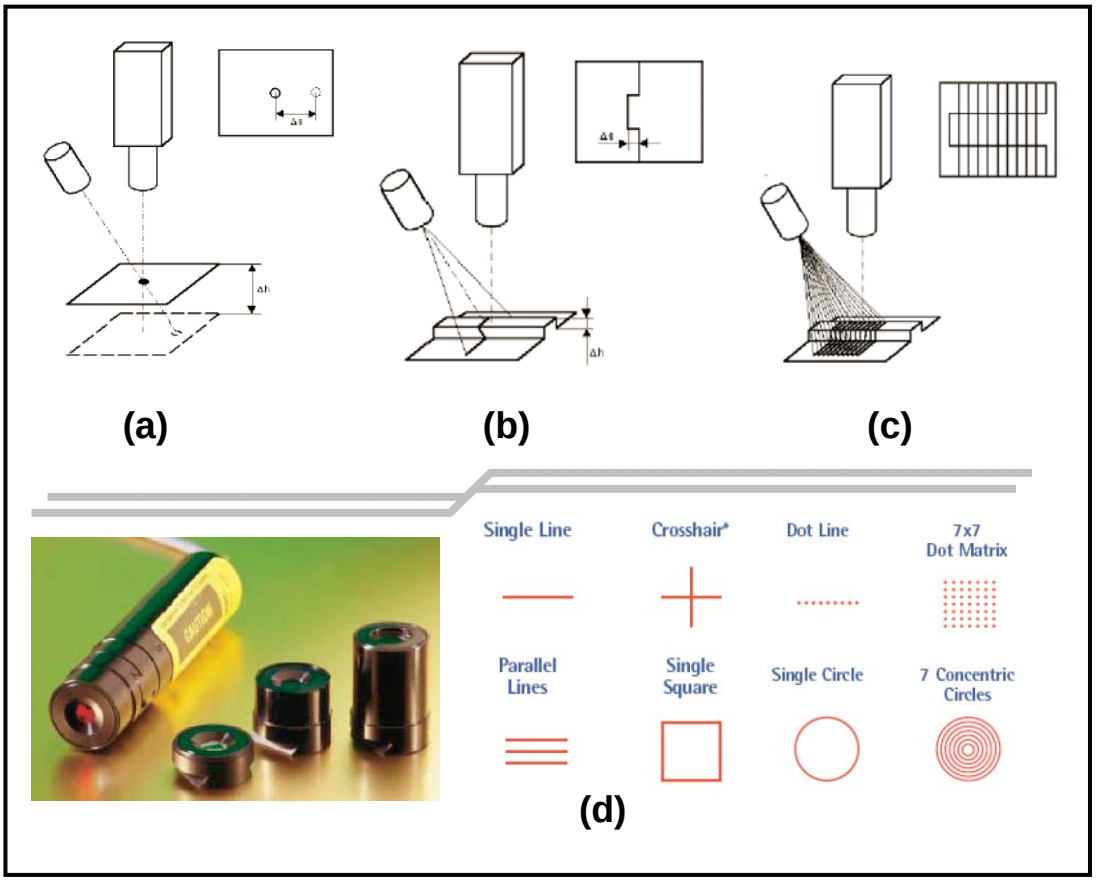


Figura 2.38 (a) medir distancia (b) medir altura (c) medir superficie (d) tipos de luz estructurada.

(GONZÁLES, Conferencia de Visión Artificial, 2008)

- También se tienen estos tipos de iluminaciones adicionales:

  - Luz “strobe”: minimiza efectos de movimiento.
  - Iluminación de “campo oscuro”: para detección de rupturas.
  - Uso de filtro y luz polarizada.
  - Uso de filtro de colores.
  - Tipo halógena, láser, fluorescente, etc.
  - Técnicas adicionales....

2.2.2.5.2.3 Equipos Iluminadores

La necesidad de obtener una imagen digital que sea lo suficientemente clara y que se ajuste a las necesidades del usuario final para cumplir ciertos requisitos para la inspección, procesamiento, etc. se hace necesario el uso de ciertos equipos iluminadores que permitan realizar una buena tarea de adquisición de imágenes, de acuerdo para cada necesidad. A continuación se van ha mencionar algunos de los equipos más utilizados.

2.2.2.5.2.3.1 Iluminación direccional frontal:

Este equipo permite realizar una iluminación puntual, ya sea iluminado frontalmente ó permitiendo un campo brilloso; entre las ventajas que se tiene al usar este tipo de equipo es que son sencillos de montar, y marca sombras y realza bordes. Las desventajas es que produce una iluminación despareja, y produce sombras no deseadas. En la **figura 2.39** se puede observar un ejemplo con un equipo de iluminación frontal direccional, en el que se analiza el borde de un chip para buscar posibles daños, como pines rotos o doblados, etc. se puede ver como se pierde el otro borde.

2.2.2.5.2.3.2 Iluminación de Campo Obscuro.

Éste iluminador realza superficies con cavidades en las cuales se trata de observar las características de dichas cavidades, por ejemplo en piezas en las cuales se tiene grabado el número de serie, etc.; la ventaja entonces sería que acentúa variaciones superficiales. Pero las desventajas que se tiene al usar este iluminador son: producen “Hot Spots” (manchas), debe estar muy cercano al objeto, produce sombras sobre la característica a resaltar.

En la **figura 2.40** se ilustra un ejemplo de este iluminador, en la que se desea tener información por ejemplo del número de la pieza.

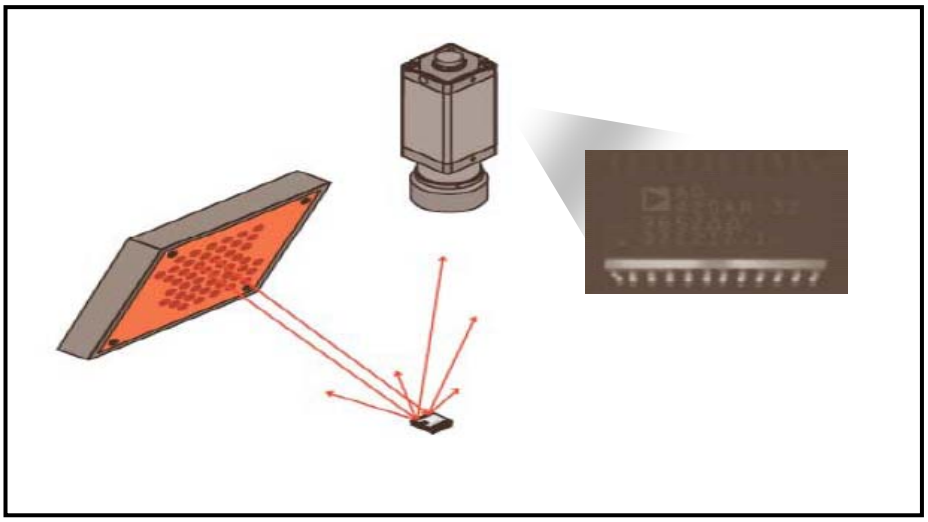


Figura 2.39 Ejemplo de un equipo iluminador Frontal Direccional.

(GONZÁLES, Conferencia de Visión Artificial, 2008)

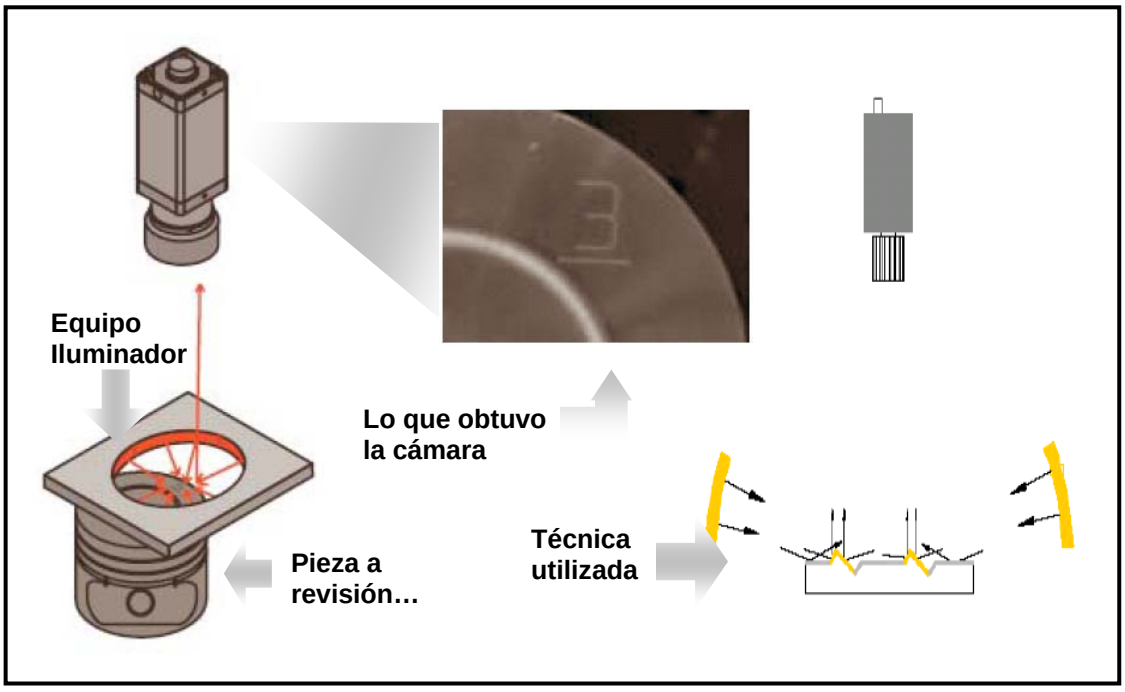


Figura 2.40 Ejemplo del equipo iluminador de Campo Obscuro.

(GONZÁLES, Conferencia de Visión Artificial, 2008)

2.2.2.5.2.3.3 Iluminador tipo Back Light.

Éste iluminador es quizás el más utilizado, posee características difusas ó puntuales colimadas. Las ventajas es que se pueden observar de mejor manera las siluetas, además se obtienen mediciones de precisión (Colimado). Las desventajas puede ser el montaje, y se obtienen bordes borrosos (no colimado), además también se pierde información de la superficie, como puede ser algo que este escrito, grabado, etc.

En la **figura 2.41** se puede observar un ejemplo de este iluminador, en la que se desea observar si algún pin de un chip está dañado, roto, doblado, etc.

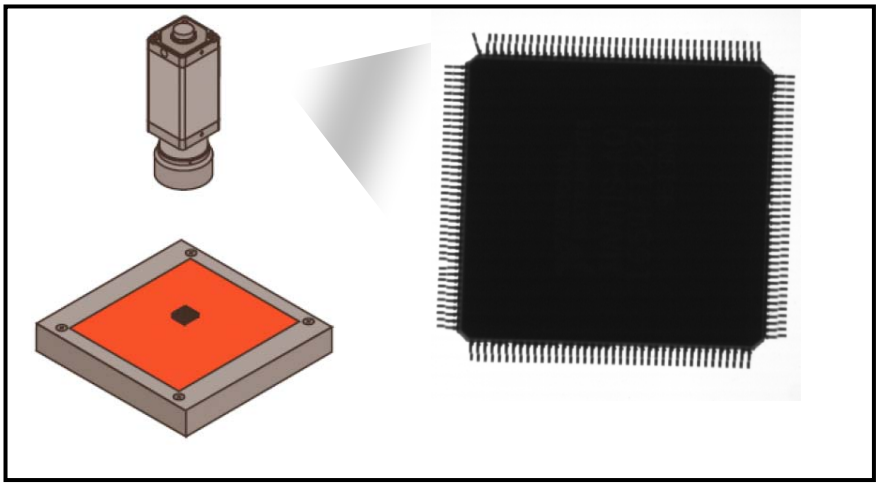


Figura 2.41 Ejemplo del iluminador Back Light.

(GONZÁLES, Conferencia de Visión Artificial, 2008)

2.2.2.5.3 La Cámara

Se había dicho que una imagen era la distribución espacial de intensidad lumínica en una escena, ahora la cámara lo que va ha realizar es la transducción de esa distribución espacial de luz a señales eléctricas.

El principio de funcionamiento de una cámara fotográfica se basa en el principio de la cámara obscura. ***Aristóteles*** la describió de la siguiente manera: "*Se hace pasar la luz a través de un pequeño agujero hecho en un cuarto cerrado por todos sus lados. En la pared opuesta al agujero, se formará la imagen de lo que se encuentre enfrente de forma invertida*". En la **figura 2.42** se ilustra un ejemplo de una cámara obscura.

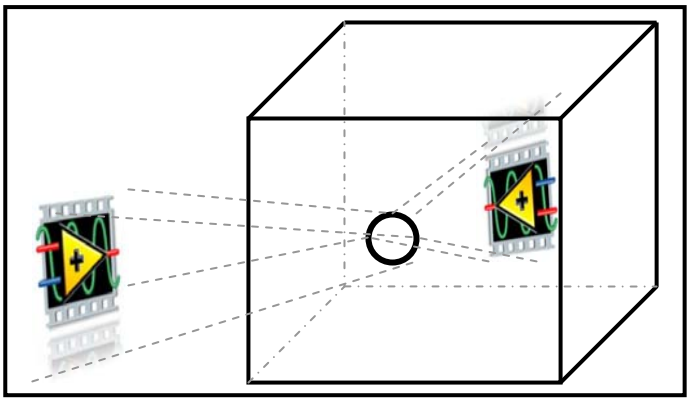


Figura 2.42 Ejemplo de una cámara Obscura.

Entonces se puede decir que una cámara entrega una representación de una imagen proyectada por un sistema óptico.

2.2.2.5.3.1 Clasificación:

Las cámaras se clasifican:

- Según su tipo de señal, es decir si es analógico ó digital.
- Según el tipo de adquisición: entrelazado, line-scan ó progresivo. De acuerdo al tipo de imagen: monocroma, a colores, o Infrarroja (IR), y finalmente…
- Según el tipo de sensor que utilice: CCD (Dispositivo Acoplado por Carga) ó CMOS.

En la **figura 2.43** se muestran algunos tipos de cámaras, para trabajar en visión artificial, para aplicaciones industriales.

Figura 2.43 Algunos tipos de cámaras, para trabajar en visión artificial.

(ORELLANA, Visión Artificial basada en PC, 2009)

2.2.2.6 Adquisición de las imágenes. (Hardware)

Se había mencionado que la cámara realizaba la transducción de la distribución espacial de luz a señales eléctricas, para realizar esta tarea se necesita de los sensores, que obviamente van ha estar dentro de la cámara. Los sensores en una cámara básicamente son un arreglo de fotodiodos que convierten luz en electrones, conocido como efecto fotoeléctrico. La colección de fotoelectrones va ha depender básicamente de la intensidad luminosa y del tiempo de exposición.

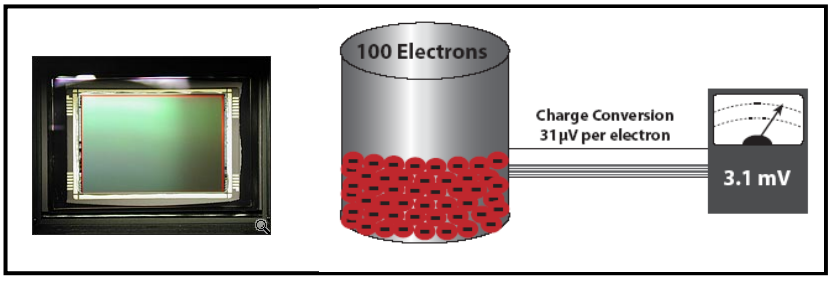


Figura 2.44 A la izquierda un ejemplo de un Sensor CCD, a la derecha se observa la conversión de los fotoelectrones a su equivalente en voltaje.

(GONZÁLES, Conferencia de Visión Artificial, 2008)

Existen dos tipos ó tecnologías de sensores de imagen los CCD y los CMOS, ambos son de silicio y son similares en cuanto a sensibilidad al espectro visible. Ambas tecnologías convierten la luz incidente (fotones) en carga electrónica (electrones) por el miso proceso de conversión.

2.2.2.6.1 Sensores CCD

Los Sensores CCD por sus siglas en inglés "***Charge-Coupled Devices***", su traducción al español sería "Dispositivo Acoplado por Carga", es un dispositivo compuesto por sensores de imagen que utilizan elementos semiconductores fotosensibles en forma de arreglos matriciales.

La carga eléctrica almacenada en la celda es posteriormente transportada utilizando un registro de desplazamiento (*Shift Register*) para conformar una señal de video como puede verse en la **figura 2.45**. Cabe señalar que en las cámaras CCD se discretiza la imagen en pixeles, sin embargo el valor de la carga eléctrica almacenado en cada celda no se digitaliza en el arreglo CCD sino en una conversión posterior realizada por un conversor analógico-digital.