CAPÍTULO II

VISUAL BASIC 6

Introducción.

Visual Basic es uno de tantos lenguajes de programación; orientado a objetos, que podemos encontrar hoy en día. Dicho lenguaje nace de BASIC (Beginner´s All-purpose Symbolic Instruction Code) el cual se desarrolló en el año de 1963 en Dartmouth College, Hanover, New Hampshire; con el propósito de ayudar a aquellas personas interesadas en iniciarse en algún lenguaje de programación; BASIC se caracterizaría por su sencillez, en comparación con otros lenguajes de la época, como COBOL, FORTRAN y ALGOR.

Luego de sufrir varias modificaciones, en el año 1968 se estableció el lenguaje BASIC estándar; debido a la sencillez del lenguaje, no tuvo aceptación entre los programadores avanzados por considerarlo "un lenguaje para principiantes".

En 1975 surge GW-BASIC, la primera versión de BASIC para computadoras personales; desarrollada por Microsoft y la cual era incluida en el sistema operativo MS-DOS; no fue hasta el año de 1983 cuando surge el primer compilador comercial, diseñado para desarrollar programas en lenguaje BASIC dentro de la plataforma DOS. Esta versión fue llamada QuickBASIC; con el paso del tiempo se desarrollo una nueva versión llamada QBASIC similar a QuickBASIC, las cuales servirían como bases junto con GUIs (Graphical User Interfaces), para crear en 1987 Visual Basic. La versión más reciente es la 6 que se incluye en el paquete Visual Studio 6 de Microsoft; esta versión

combina la sencillez de BASIC con un poderoso lenguaje de programación Visual, que juntos permiten desarrollar robustos programas de 32 bits para el ambiente de Windows.

Esta fusión de sencillez y estética permitió ampliar aún más el monopolio de Microsoft, ya que el lenguaje sólo es compatible con Windows o un sistema operativo de la misma empresa.

Visual Basic ya no es considerado "un lenguaje para principiantes" sino que es una perfecta alternativa para los programadores de cualquier nivel, que deseen desarrollar aplicaciones compatibles con Windows. Visual Basic permite escribir, editar y probar aplicaciones para Windows; además de incluir herramientas que: compilan archivos de ayuda, generan controles ActiveX e incluso aplicaciones para Internet.

Visual Basic cuenta con un verdadero compilador de código fuente que crea archivos ejecutables en menor tiempo; además incluye varios asistentes, los cuales presentan cuadros de dialogo basados en preguntas y respuestas que orientan al programador en la creación de aplicaciones. Visual Basic facilita el diseño de aplicaciones en un entorno gráfico (GUI-GRAPHICAL USER INTERFACE) como Windows 98, Windows NT o superior.

Existen tres ediciones de Visual Basic 6, que son: la de Aprendizaje (Learning Edition), la Profesional (Professional Edition), y la Empresarial (Enterprise Edition). La mayoría de los programadores sólo necesitan las ediciones de Aprendizaje o Profesional; mientras que la edición Empresarial está dirigida a aquellos individuos que

se especializan en la creación de aplicaciones cliente/servidor. Esta edición ha sido mejorada para auxiliar a los programadores que trabajan en el desarrollo de aplicaciones cliente/servidor muy específicas.

Actualmente se necesita mucho más que sólo un lenguaje; se necesita herramientas de desarrollo gráfico que pueda aprovechar todas las capacidades gráficas, multimedia, de conectividad y multitareas que ofrece Windows. Visual Basic es una herramienta; más que un lenguaje, Visual Basic permite generar aplicaciones que interactúan con todos los recursos de los sistemas operativos de Windows.

Una ventaja con la que cuenta Visual Basic es que es capaz crear aplicaciones gráficas para Windows insertando el código dentro de las mismas interfaces, evitando la escritura del código necesario para dibujar los cuadros de diálogo. Gracias a la interfaz Visual Studio, podemos saber el aspecto que va atener nuestra aplicación sin necesidad de compilarla ni ejecutarla.

Características de Visual Basic.

Diseñador de entorno de datos: Es posible generar; de manera automática, conectividad entre controles y código mediante la acción de arrastrar y colocar sobre formularios o informes.

Los Objetos ActiveX: Son una nueva tecnología de acceso múltiple; que puede ser manipulada como cualquier objeto (arrastrar y colocar) y permite el intercambio de datos entre programas.

Asistente para formularios: Genera de manera automática formularios que administran tablas de registro o consultas pertenecientes a una base de datos, hoja de cálculo u objeto (ADO-ACTIVE DATA OBJECT).

Asistente para barras de herramientas: Es factible incluir barras de herramientas personalizadas, donde el usuario selecciona los botones que desea visualizar durante la ejecución.

Interfaz gráfica de Visual Basic 6: La cual podemos apreciar en la **Figura 2.1**, la cual muestra el área de trabajo, la barra de herramientas que permite acceder fácilmente a los elementos insertables, necesarios para el desarrollo de nuestras aplicaciones; por solo citar algunos podemos encontrar: botones, recuadros de texto, objetos ole, reproductor de video, barras de desplazamiento, entre otros. En la parte derecha, se encuentra una ventana en donde se expone la información referente a las propiedades del elemento o herramienta, en ella se editan dichas propiedades de acuerdo a nuestros requerimientos.

Objetos de datos ActiveX.

La última y más grande estrategia de acceso a datos de Microsoft son los Objetos de Acceso a Datos ActiveX, a los que se hace referencia como ADO. Se define como ADO a una interfaz de programación de alto nivel con proveedores de menor nivel para el acceso a datos en diferentes formatos y localizaciones. Estos objetos se caracterizan por las extensiones .DLL y la gran mayoría de ellos se encuentran en la carpeta System32 de Windows.

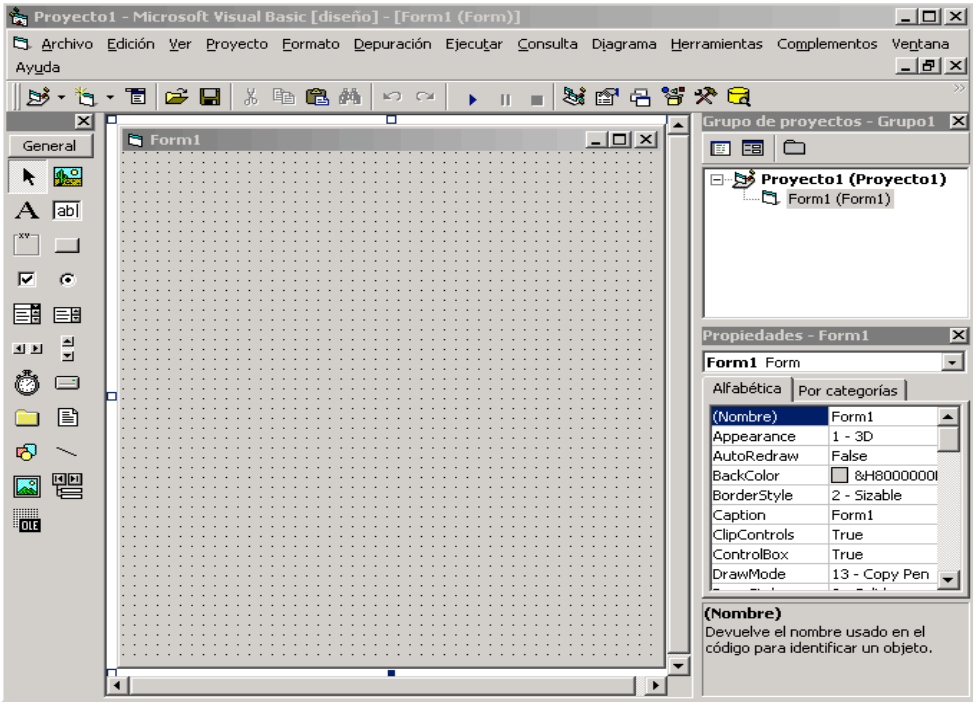

Figura 2.1 Interfaz gráfica de Visual Basic

Los antecesores de los contres ActiveX fueron los objetos OLE (Vinculación e incrustación de objetos); desde 1992 este tipo de objetos aparecen en el ambiente de Windows y fue en Windows 3.1 donde hicieron su primera aparición. Los objetos OLE utilizaban DDE como el sistema subyacente para comunicar aplicaciones entre sí y permitió a los usuarios y programadores crear documentos compuestos (es decir, documentos que contenían datos provenientes de diferentes aplicaciones, por ejemplo, una hoja de cálculo de Excel dentro de un documento de Word). Dependiendo las necesidades del usuario, los documentos compuestos podían encerrar completamente a otros documentos más simples (incrustación) o contener una referencia a otros documentos existentes (vinculación); es decir, cuando el usuario pulsaba sobre un documento incrustado o vinculado dentro de un documento compuesto, Windows ejecutaba la aplicación que se había utilizado para crear ese tipo particular de documentos, desplegando la información en el documento compuesto.

Utilización de OLE

Compartir información sobre distintas aplicaciones es una de las características más poderosas que posee Windows. Al intercambiar información, lo que en realidad ocurre es compartir objetos que son creados por dos o más aplicaciones Windows. Aquí es donde entra en juego OLE, sentando las bases para crear y compartir objetos entre distintas aplicaciones.

Visual Basic actúa como nexo entre las aplicaciones Windows al permitir el uso de objetos OLE en sus proyectos; lo cual da la capacidad de tener en un mismo formulario una hoja de cálculo de Excel y un documento de Word. Visual Basic puede utilizar los objetos de otras aplicaciones de la misma forma que cualquier otro objeto dentro de Visual Basic; estableciendo o leyendo sus propiedades, utilizando sus métodos, etc. A esto se le denomina Automatización OLE.

El contenedor OLE.

Mediante el uso del control contenedor OLE, se puede crear formularios de Visual Basic en los que se incluya información proveniente de distintas aplicaciones Windows **(Figura 2.2).**

Figura 2.2 Control contenedor OLE

Estos formularios son conocidos como documentos compuestos, en los que la funcionalidad que se necesita no la proporciona la aplicación creada por Visual Basic, sino que se vale de las aplicaciones que crean los objetos insertados.

Así es que al momento de programar, si se desea conseguir que nuestra aplicación presente la funcionalidad de una hoja de cálculo o un procesador de texto; no será necesario programarlo, se valdrá de la misma aplicación para mostrar lo requerido.

Para generar un objeto OLE en nuestra aplicaciones se presiona el control contenedor OLE **(Figura 2.2)** el cual mostrara el cuadro de dialogo que se muestra en la **Figura 2.3.** En el cuadro de dialogo se mostraran los distintos tipos de objetos que pueden ser insertados en el formulario; estos objetos provienen de las aplicaciones compatibles con OLE que estén instaladas en la computadora.

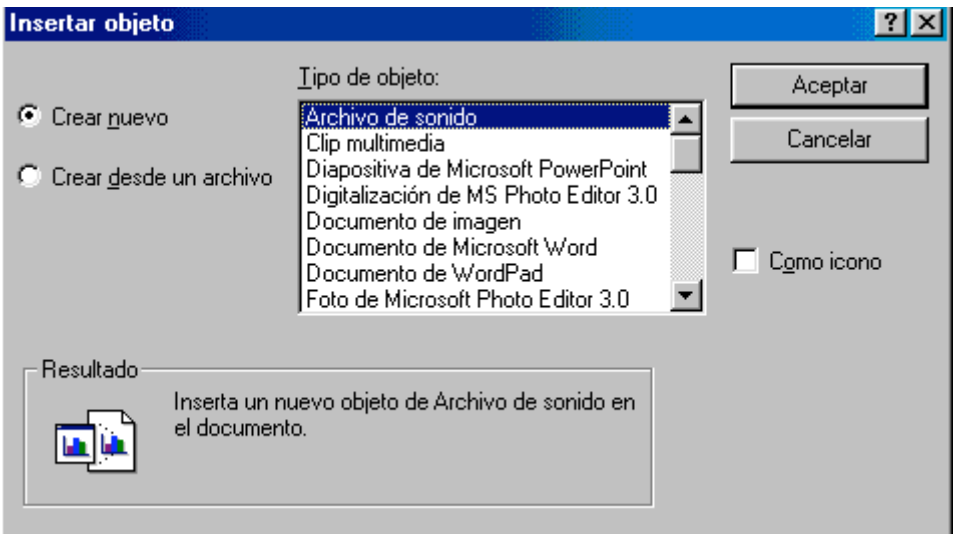

Figura 2.3 Cuadro de dialogo para seleccionar el tipo de objeto OLE

Visual Basic reconoce los objetos OLE disponibles a través del Registro de Windows; el registro es una base de datos donde Windows almacena información

referente a las aplicaciones que se encuentran instaladas. Entre otras cosas, el registro de Windows almacena los objetos insertables que ofrecen dichas aplicaciones, que posteriormente serán desplegadas en el cuadro de diálogo Insertar Objeto **(Figura 2.3).**

Se puede insertar un objeto en el formulario utilizando la opción Crear Nuevo o utilizar uno ya existente mediante la opción crear desde archivo; en este último caso se tiene que especificar la ruta del archivo que se desea insertar. Además se tiene la opción de mostrar el objeto insertado como un icono; para ello se tiene que activar la casilla mostrar como icono, del cuadro de diálogo insertar objeto.

Cuando se incrusta un objeto, se crea una copia y es almacenada en la aplicación Visual Basic; por lo que sólo podrá ser utilizada en esta, en cambio al vincular un objeto, el programa almacena el tipo de archivo que se utilizó para crear el documento, haciendo referencia al mismo y no el objeto en sí.

Edición del objeto.

Cuando se inserta un objeto a través del control contenedor OLE; estará en disposición de editar el objeto, para esto se hace doble click sobre el objeto. Enseguida la aplicación servidora se inicializará y estará lista para interactuar; el usuario podrá utilizar toda la funcionalidad que se presente. Es importante mencionar que algunas aplicaciones permiten la edición en el propio control contenedor, mientras que otras se editan en una ventana independiente.

Para que la edición del objeto sea en el propio contenedor OLE, es necesario incrustar el objeto, de esta manera la vinculación mostrará una ventana dentro de nuestro programa, que desplegará la aplicación servidora. Por otra parte, si el objeto se representa como un icono, la aplicación servidora se exhibirá en una ventana independiente; en este caso los menús y barras de herramientas se verán de manera normal; en cambio, al ser insertado el documento, se tendrá que negociar dichos elementos; es decir, se tiene que indicar que menús y barras de herramientas sean visibles. Esto se logra agregando un elemento de menú dentro de nuestro formulario; utilizando el editor de menús del menú herramientas, en el se especifica los elementos a desplegar.

La propiedad Negociate Position que aparece en el editor de menús, establece la posición de los elementos del menú del formulario. Puede tener los siguientes valores:

- 0: Indica que dicho elemento del menú se verá cuando se edite el objeto, mostrándose únicamente el menú de la aplicación servidora.

-1, 2 y 3: Establece la posición del menú (izquierda, centro o derecha).

Independientemente de la propiedad Negociate Position, es preciso establecer la propiedad Negociate Menú en True; de lo contrario, la negociación de menús no se llevará a cabo.

Es importante recordar que toda vinculación o uso de la opción Mostrar como Icono implicarán que el objeto se edite en una ventana independiente y por consiguiente resulta innecesario vincular los menús a menos que el usuario lo requiera.

Objetos insertables.

Algunas aplicaciones proporcionan objetos que pueden utilizarse como controles personalizados en Visual Basic; de esta forma se pueden agregar a la barra de herramientas de Visual Basic sin tener que utilizar el control contenedor OLE. Para agregar los controles se utiliza el menú contextual de la barra de herramientas o se selecciona componentes del menú de proyecto **(Figura 2.4).**

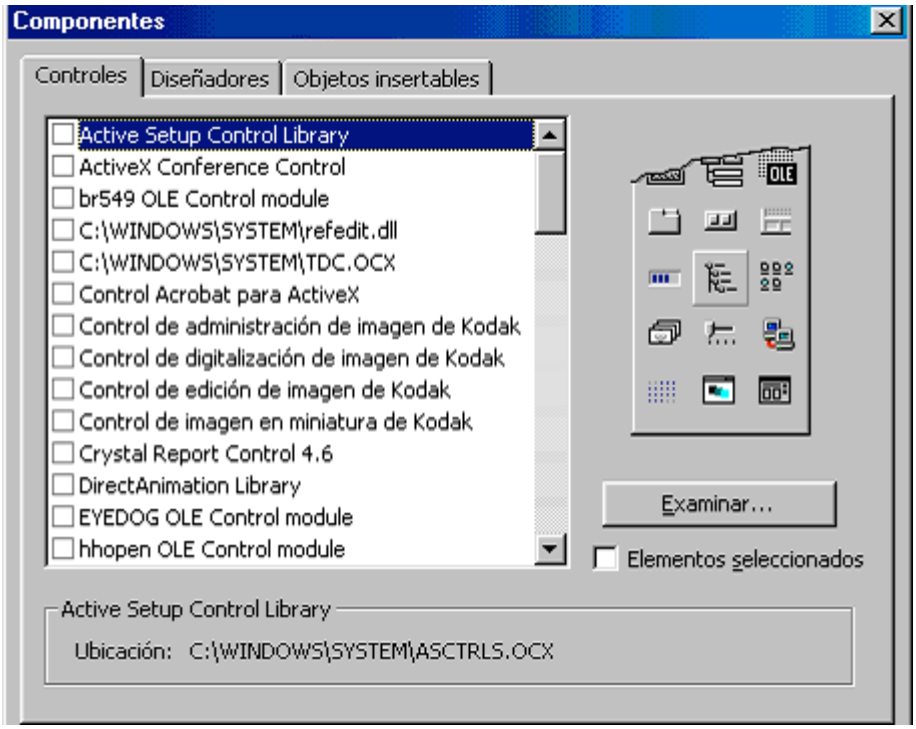

Figura 2.4 Selección de controles personalizados.

Al seleccionar cada casilla se desplegará el control deseado; por ejemplo si se seleccionara la casilla Documento de Microsoft Word del menú objetos insertables, se exhibirá el control en la barra de herramientas, de modo que se accesará directamente a la función y no será necesario utilizar el control contenedor OLE. Además es posible que una aplicación aporte más de un tipo de objeto insertable; como es el caso de Excel, que permite la inserción de objetos hoja de cálculo y gráficos.

Es importante recalcar que al utilizar un objeto insertable; sólo se podrá incrustar un objeto nuevo, debido que no cuenta con la opción de utilizar archivos existentes, pero si podrá editar el objeto en tiempo de ejecución con las mismas reglas que con el control contenedor OLE.

Conclusiones.

Como se mencionó al inicio de este capítulo; Visual Basic es un lenguaje con instrucciones muy simples y nemónicas; por eso el nombre de Basic "básico", además de ser una herramienta que se caracteriza por el manejo de programas externos, en especial los programas desarrollados por Windows. Debido a que era primordial utilizar el reproductor de video **Windows Media Player**; Visual Basic se adapto perfectamente a nuestras necesidades, de lo contrario la implementación del tutorial se habría complicado considerablemente.

Aunque Visual Basic es capaz de crear controles ActiveX, durante el desarrollo del Tutorial no fue necesario implementarlos; ya que vasto con los objetos insertables. Los objetos insertables que se utilizaron fueron: Windows Media Player (reproductor de

video), Word y Excel; estos dos últimos se utilizaron para vincular los archivos de texto y las tablas.