UNIVERSIDAD NACIONAL
AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES
ACATLÁN

Programa "CACHO" para Cuantificación de planos en Autocad, realizado en lenguaje Visual Basic 6.0

TESIS PROFESIONAL

PARA OBTENER EL TÍTULO DE
INGENIERO CIVIL

PRESENTA
Gabriel González López
No. De Cta. 4-0000226-4

ASESOR
Mtro. Pablo Miguel Pavía Ortíz

FECHA: FEBRERO 2013

OPCION DE TITULACIÓN
Tesis

TITULO
Programa "CACHO" para cuantificación de planos en Autocad, realizado en lenguaje Visual Basic 6.0

OBJETIVO GENERAL:
Desarrollar en lenguaje Visual Basic 6.0, un programa que apoye en la obtención de cantidades de obra, a partir de los planos en formato digital, para optimizar el proceso de generación.

INTRODUCCION

En cualquier campo de la ingeniería, algo que no se puede dejar de lado, es el trabajo del área de Costos, uno de los más demandantes de tiempo, rapidez y exactitud.

Una actividad de Costos, si no es la principal si es una de las más demandantes, es el presupuesto; donde interviene dos partes principales, la cantidad y el PU. Donde la cantidad es muy solicitante de tiempo, dedicación y esfuerzo.

El propósito del presente trabajo es desarrollar un programa que apoye en la realización del presupuesto, tomando la cantidad como parte fundamental para elaborar costos; el segmento de la obtención de las cantidades de obra durante mucho tiempo ha sido controlado por los generadores de obra, para registrar y mostrar de donde se ha obtenido dicho valor de lo cuantificado. Al presupuesto sólo le llega el total, pero detrás de este valor hay bastante tiempo invertido, por ello la necesidad de desarrollar una herramienta que ayude a optimizar el tiempo y la precisión con que se desarrollan estos trabajos.

Actualmente, la mayoría de los trabajos de cuantificación se realizan en planos en formato digital, "dwg"; que es la extensión de los archivos generados en "Autocad", y se toman medidas del programa por la gran precisión que tiene y por la facilidad para cambiar y obtener la información. Cualquiera que sea el campo de aplicación nos remitimos a que en Autocad todos los conceptos están representados mediante objetos de este programa como líneas, arcos, polilíneas, círculos, bloques, etc, de donde se puede obtener, longitudes, áreas y piezas, principalmente. Es aquí es donde se necesita una herramienta para agilizar el proceso de toma de medidas y de registro y control de la información.

INDICE

CAPITULO 1. ANTECEDENTES

Objetivo

Citar los programas existentes, AutoCAD y Excel, que son utilizados como herramienta de trabajo en las diferentes áreas de aplicación de la ingeniería civil.

1.1. Generalidades

Con el paso del tiempo nos hemos apoyado en la tecnología disponible para desarrollar el trabajo de ingeniería, actualmente destacan dos programas como herramientas del ingeniero, Excel y AutoCAD, para cálculos y dibujo de planos, respectivamente.

En diferentes áreas de la ingeniería civil la administración de información es fundamental; cuando es bastante la información por administrar,la herramienta ideal es una base de datos, la cual es considerada como infinita, el límite es la capacidad del disco duro de la PC en uso. Y la combinación de estas tres herramientas le da al ingeniero de gabinete un gran respaldo y soporte a los trabajos que realiza.

1.2. AutoCAD. Origen, antecedentes y aplicación en la ingeniería civil.

AutoCAD proviene; Auto de Autodesk[1]empresa creadora del software y CAD de sus siglas en inglés "Computer Aided Design", diseño asistido por computadora. Este programa tiene sus orígenes a mediados de la década delos 80; el primer producto notable de la empresa Autodesk fue **AutoCAD** un derivado del CAD diseñado para funcionar en las plataformas de microcomputadoras de la época incluyendo computadoras de 8 bits que ejecutaban el sistema operativo CP/M [2]y DOS[3] de los entonces nuevos sistemas operativos de 16 bits como Victor 9000 y la IBM PC.

Esta herramienta de **CAD** permitía crear dibujos técnicos detallados, y era económicamente accesible para pequeñas empresas de diseño, ingeniería y arquitectura. En la versión 2.1 se presentó un nuevo concepto en industria del CAD y del software: el software plataforma abierta, por medio de la introducción de un intérprete de lenguaje de programación **lisp**y**AutoLisp**, modificado para las soluciones particulares incorporadas en AutoCAD. Además, también implementaron un subconjunto de la bibliotecas de lenguaje de programación C y fue puesto a disposición de programadores.

Desde el lanzamiento de la versión 12, la compañía deja de soportar el sistema operativo Unix y Apple Macintosh, y tras la versión 14 descontinuó MS-DOS como plataforma, trabajando en conjunto con Microsoft para compartir sus tecnologías y obtener un mayor desempeño en el sistema operativo de Windows.

---

[1]**Autodesk**, Inc., es una compañía dedicada al software de diseño en 2D y 3D para las industrias de manufactura, infraestructura, construcción, medios y entretenimiento y datos transmitidos vía inalámbrica. Autodesk fue fundada en 1982 por John Walker y otros doce cofundadores. A lo largo de su historia, ha tenido varias sedes, se encuentra actualmente en San RafaelCalifornia.

[2]**CP/M** (Control Program for Microcomputers) es un sistema operativo desarrollado por Gary Kildall para el microprocesadorIntel 8080, Se trata del sistema operativo más popular entre las PC en los años 1970.

[3]**DOS** es una familia de sistemas operativos para PC. El nombre son las siglas de *disk operating system* ("sistema operativo de disco"). Fue creado originalmente para la familia IBM PC, que utilizaban los procesadores Intel8086 y 8088, de 16 bits.

En 2002, Autodesk compró un software de modelado paramétrico relacionado, llamado Revit, que pertenecía a la empresa ubicada en Massachusetts llamada **Revit Technologies** por un importe $133 millones de dólares. Revit, está hecho para soluciones del edificio y el grupo de la infraestructurae Inventorpara el grupo de fabricación, son ahora el cimiento para los futuros productos de Autodesk separándose de su base de código durante 20 años que fue AutoCAD.

El 4 de octubre de 2005, Autodesk anunció su intención de adquirir **Alias** que concretó el 10 de enero de 2006, por la suma de 197 millones de dólares.

Parte del programa **AutoCAD** está orientado a la producción de planos, empleando para ello los recursos tradicionales de grafismo en el dibujo, como color, grosor de líneas y texturas. AutoCAD, a partir de la versión 11, utiliza el concepto de espacio modeloy espacio papel para separar las fases de diseño y dibujo en 2D y 3D, de las específicas para obtener planos trazados en papel a su correspondiente escala. Es en la versión 11, donde aparece el concepto de modelado sólido a partir de operaciones de extrusión, revolución y las booleanas de unión, intersección y sustracción. Este módulo de sólidos se comercializó como un módulo anexo que debía de adquirirse aparte. Este módulo sólido se mantuvo hasta la versión 12, luego de la cual, AutoDesk, adquirió una licencia a la empresa Spatial, para su sistema de sólidos ACIS.

AutoCAD gestiona una base de datos de entidades geométricas (puntos, líneas, arcos, etc) con la que se puede operar a través de una pantalla gráfica en la que se muestran éstas, el llamado editor de dibujo. Las versiones modernas del programa permiten la introducción de éstas mediante una interfaz gráfica de usuario o en inglés **GUI**, que automatiza el proceso.

Como todos los programas de CAD, procesa imágenes de tipo vectorial, aunque admite incorporar archivos de tipo fotográfico o mapa de bits, además permite organizar los objetos por medio de capas, ordenando el dibujo en partes independientes con diferente color. El dibujo de objetos seriados se gestiona mediante el uso de bloques, facilitando la definición y modificación única de múltiples objetos repetidos.

La extensión del archivo de AutoCAD es dwg, aunque permite exportar en otros formatos como es el dxf y formatos IGES y STEPpara manejar compatibilidad con otros programas de dibujo.Los formatos de archivo DXF y DWGson los más comunes para el intercambio del CAD.

El formatodxf permite compartir dibujos con otras plataformas de dibujo CAD, reservándose AutoCAD el formatodwg para sí mismo. El formatodxf puede editarse con un procesador de texto básico, por lo que se puede decir que es abierto. En cambio, eldwg sólo podía ser editado con AutoCAD, si bien desde hace poco tiempo se ha liberado este formato (DWG), con lo que muchos programas CAD distintos del AutoCAD lo incorporan, y permiten abrir y guardar en esta extensión, con lo cual lo del DXF ha quedado relegado a necesidades específicas.

Actualmente todos los profesionistas vinculados con el diseño de proyectos Arquitectónicos y de Ingeniería Civil necesitan herramientas de dibujo para el desarrollo de los mismos. La posibilidad de administrarlos, compartirlos y editarlos con gran facilidad. En México en la ingeniería civil, AutoCAD es utilizado en todas las áreas, ya sea consulta de información y/o generando esta; por mencionar algunas, al realizar una carretera, se necesita plasmar en planos el trazo de esta, perfiles del terreno y del proyecto para conocer la cantidad de material a utilizar, curvas de nivel, puentes, entronques con otras vías y toda la infraestructura inducida por la construcción de dicha carretera; en estructuras, para planos de taller de estructura metálica, planos de secciones estructurales ya sea de concreto y/o acero, detalles de armado de varillas en los elementos estructurales como cimentación, columnas, vigas, losas, etc.; en el área ambiental, desde la información topográfica y de vialidades para ubicar sitios de disposición final de residuos sólidos para determinar el terreno, el proceso de clausura o apertura de un sitio; diseño de elementos como celdas, drenes, formación de basura, etc; aún en el área de administración de obra, donde no se genera información pero si se necesita revisar planos y especificaciones para poder suministrar materiales en las cantidades correctas es utilizado esta herramienta de manera cotidiana. Enfocándonos en el área de costos, donde después de generar la información en formato digital, es primordial obtener, de cada uno de los planos, las cantidades de obra para que el presupuesto se lleve a cabo; hace unos años las cuantificaciones se realizaban en un plano físico, ya sea realizado a mano u obtenido de manera impresa; actualmente se puede cuantificar desde el archivo digital; con cualquier proceso es necesario invertir suficiente tiempo para tener información confiable, precisa y calidad aceptable; además por supuesto de conocimientos técnicos para la adecuada interpretación de la información.

Para propósitos de este trabajo, que pertenece al área de costos; Autocad nos permite obtener la información de manera precisa y rápida, directamente de la computadora, sin necesidad de imprimir y lidiar con las escalas, sólo con dar lectura e interpretación de los planos para poder obtener lo requerido; cabe mencionar que el error más frecuente para poder consultar la información, es que en el dibujo se duplican los elementos representativos, por ejemplo un muro o alguna trayectoria que visualmente es una línea, está representada con más elementos y eso llega a confundir las mediciones si no se tiene la precaución de revisar el cómo está elaborado; una vez que se conoce el contenido y la información a obtener podemos aplicas diversos métodos para conseguirla, pero Autocad no está diseñado para brindar toda la información, así que es necesario una herramienta externa.

1.3. Excel. Origen, antecedentes y aplicación en la ingeniería civil.

Microsoft comercializó originalmente un programa de Hoja de cálculo llamado Multiplan en 1982, que fue muy popular en los sistemas CP/M, pero en los sistemas MS-DOS perdió popularidad frente al Lotus 1-2-3. Microsoft publicó la primera versión de Excel para Mac en 1985, y la primera versión de Windows (numeradas 2-05 en línea con el Mac y con un

paquete de tiempo de ejecución de entorno de Windows) en noviembre de 1987. Lotus fue lenta al llevar 1-2-3 para Windows y esto ayudó a Microsoft a alcanzar la posición de los principales desarrolladores de software para hoja de cálculo de PC. Microsoft empujó su ventaja competitiva lanzando al mercado nuevas versiones de Excel, por lo general cada dos años. La versión actual para la plataforma Windows es Excel 14.0, también denominada Microsoft Excel 2010.

Excel ofrece una interfaz de usuario ajustada a las principales características de las hojas de cálculo, el programa muestra las celdas organizadas en filas y columnas, y cada celda contiene datos o una fórmula, con referencias relativas, absolutas o mixtas a otras celdas.

Excel fue la primera hoja de cálculo que permite al usuario definir la apariencia de las fuentes, atributos de carácter y celdas. También introdujo cálculo inteligente de celdas, donde celdas dependientes de otra celda que han sido modificadas, se actualizan al instante (programas de hoja de cálculo anterior recalculaban la totalidad de los datos con un comando específico del usuario).

Desde 1993, Excel ha incluido Visual Basic para Aplicaciones (VBA), un lenguaje de programación basado en Visual Basic[4], que añade la capacidad para automatizar tareas en Excel y para proporcionar funciones definidas por el usuario para su uso en las hojas de trabajo. VBA es una poderosa anexión a la aplicación que, en versiones posteriores, incluye un completo entorno de desarrollo integrado. La grabación de macros puede producir código para repetir las acciones del usuario, lo que permite la automatización de simples tareas. VBA permite la creación de formularios y controles en la hoja de trabajo para comunicarse con el usuario. Admite el uso de DLL[5] de ActiveX (COM); versiones posteriores añadieron soporte para los módulos de clase permitiendo el uso de técnicas de programación básicas orientadas a objetos.

En la ingeniería Excel es una herramienta muy útil, donde podemos realizar diversas hojas de cálculo para diferentes áreas. Por mencionar algunas; en resistencia de materiales hasta estructuras, cálculos en la parte de capacidad de carga para apoyo de diseño de las secciones requeridas, desde la cimentación, columnas, trabes y losas; en hidráulica, los diámetros de tuberías y redes de agua potable, alcantarillado y otro tipo de obras hidráulicas; en ingeniería ambiental como generación de documentos para costeo, modelos financieros para optimización de recursos, horas máquina para rellenos sanitarios,

[4] **Visual Basic** es un lenguaje de programacióndirigido por eventos, desarrollado por el alemán Alan Cooper para Microsoft. Este lenguaje de programación es un dialecto de BASIC, con importantes mejoras y herramientas agregadas. Su primera versión fue presentada en 1991, con la intención de simplificar la programación utilizando un ambiente de desarrollo completamente gráfico que facilitara la creación de interfaces gráficas y, en cierta medida, también la programación misma.

[5] **DLL**es una biblioteca de enlace dinámico(sigla en inglés de *dynamic-link library*) es el término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda de un programa por parte del sistema operativo. Esta denominación es exclusiva a los sistemas operativosWindows siendo ".dll" la extensión con la que se identifican estos ficheros, aunque el concepto existe en prácticamente todos los sistemas operativos modernos.

estaciones de transferencia (ETRS), ruteo de camiones; en el área de compras se utiliza para gestionar toda la información generada ya que se obtienen los resultados de manera automática, administra los ingresos y egresos en cuestión de materiales como los anticipos, pagos, saldos, retenciones, financiamientos, fianzas, y listado de precios históricos o como el directorio digital;en el área de costos de la construcción, aunque ya existen programas muy completos para administrar la información, en muchas empresas y siendo válido, Excel es la herramienta de preferencia para desarrollar el presupuesto por la facilidad de cálculo y de ajuste de celdas parapresentarlo en un formato de calidad y tener una presentación aceptable, gestionar información para cálculo de precios unitarios, generación de obra, administración presentación de estimaciones y balances, cierres administrativos, etc.En fin puedes hacer innumerable cantidad de cálculos, con una hoja de Excel, sobre todo porque una hoja de cálculo la diseñas una vez y te sirve para resolver considerables problemassimilares, con cambiar algunos datos en las variables.

Para propósitos de este trabajo, que pertenece al área de costos; Excel es la herramienta que recibe la información de las cuantificaciones del plano y donde se gestiona para calcular diferentes cantidades de obra y presentarlos como generadores. Y facilita la utilización de formatos para aumentar la facilidad y velocidad del trabajo.

1.4. Base de datos. Origen, antecedentes y aplicación en la ingeniería civil.

Visual Basic 6.0es una excelente herramienta de programación que permite crear aplicaciones(programas) para Windows[6] y muchos otros programas que contienen plataforma en este lenguaje. Con este se puede crear desde una calculadora simple hasta una hoja de cálculo, pasando porun procesador de textos o cualquier otra aplicación que se le ocurra al programador. Ya que todo el entorno de Windows está basado en objetos y elaborado en este lenguaje.Sus aplicaciones en Ingeniería son casi ilimitadas, representación de movimientos mecánicos o de funciones matemáticas, gráficas termodinámicas, simulación de circuitos, etc. Este programa permite crear ventanas, botones, menús y cualquier otro elemento gráfico de una forma fácil e intuitiva.

Visual Basic nos permite trabajar directamente con distintas bases de datos como ACCESS, dBaseIII, dBaseIV, dBase 5, Excel3, Excel4, Excel5, Excel7, FoxPro2.x, Foxpro3.0, LotusWK1, LotusWK3, LotusWk4, Paradox3.x, Paradox4.x y Paradox5.x;esto lo logra mediante el **Motor de Bases de Datos Jet;** DAO que en una interfaz orientada a objetos para Microsoft Jet, Visual Basic utiliza esta herramienta para abrir, agregar y recuperar registros y gestionar transacciones, es la forma más sencilla y rápida de acceder a una base de datos Access instalada en el propio disco duro o en red.

[6]**Windows**es una familia de sistemas operativos desarrollados por Microsoft desde 1981; comenzó por ser una extensión gráfica de MS DOS; al paso del tiempo ha ido agregando componentes que han evolucionado la aplicación, interfaz del usuario mediante ventanas, kernel que es la función que permite la multitarea, Plug and Play que detecta componentes físicos de la PC, seguridad y características de red profesional, funcionalidad para TV, CD y DVD, y a la fecha el multi-touch para el acceso desde la pantalla y mejoras en el rendimiento sobre todo en velocidad y con menores recursos.

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

En la ingeniería, en general, las bases de datos son para administrar la información almacenar, consultar y modificar datos, de gran tamaño con numerosas registros; propiamente en una base de datos la información puede ser consultada con una mayor velocidad que en un archivo de Excel, y puede ser de forma invisible sin tener que abrir alguna aplicación, sólo de donde se reportan los datos; además que los campos de información no son limitados.

Existen programas denominados sistemas gestores de bases de datos, que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Para los propósitos de este trabajo, se ha elegidopara su utilización y administración la base de datos de Microsoft Access.

CAPITULO 2. PRECIOS UNITARIOS

Objetivo

Explicar los conceptos de un presupuesto, el análisis del precio unitario (desarrollo dela matriz), y procedimientos para la obtención los volúmenes de obra.

2.1. Presupuesto. Partes y descripción.

Un presupuesto es un plan expresado en términos numéricos; también se conceptúa como un sistema de información, pues procesa datos cuantitativos y cualitativos que se resumen en informes administrativos y estados financieros. El presupuesto es la suma del producto precio unitario y cantidad. Contiene uno o varias partidas, la cual se puede definir como actividad a realizar y en casos.

Desde hace mucho tiempo, el hombre ha tenido la necesidad de planear su quehacer futuro, en términos de construcción el objetivo es que en la incursión de la obra esta cumpla con lo primordial, el generar utilidades; el presupuesto refleja las cifras que se espera obtener, puede ser elaborado para cualquier tipo de obra; la importancia del presupuesto radica en la posibilidad de presentar con anticipación los principales indicadores administrativos y financieros, como por ejemplo la productividad, la liquidez, la rentabilidad, los niveles de demanda; las cifras que sustenta el presupuesto deben ser calculados considerando un cierto grado de riesgo, ya que existen crear supuestos de su información. Las principales ventajas del presupuesto son:

a) La obtención de estados financieros para la toma de decisiones preventivas
b) El establecimiento de objetivos más claros y específicos por parte de la dirección de la empresa
c) Contar con una organización bien definida en sus niveles y áreas de responsabilidad, autoridad y comunicación.
d) Facilita la concertación de compromisos en corto plazo.
e) Motivar al personal de la empresa a involucrarse con las cifras, pues es más fácil a manera los objetivos a lograr.
f) Presenta indicadores financieros y administrativos con anticipación.
g) Vincula la organización con los escenarios económicos del futuro.
h) Está relacionado con eficiencia operacional.
i) Es una herramienta para el empleo óptimo de los recursos

Un presupuesto está compuesto de los costos directos y los costos indirectos. El costo directo es la suma de los costos de materiales, mano de obra (incluyendo prestaciones),equipos, herramientas, y todos los elementos requeridos para un proceso productivo.Estos costos directos que se analizan de cada una de las partidas conformantes de una obra puedentener diversos grados de aproximación de acuerdo al interés propuesto. Sin embargo, el efectuar unmayor refinamiento de los mismos no siempre conduce a una mayor exactitud porque siempre existirándiferencias entre los estimados de costos de la misma partida. Ello debido a los diferentescriterios que se pueden asumir, así como a la experiencia del Ingeniero que elabore los mismos.

El costo indirecto es la suma de los gastos técnicos y administrativos necesarios para la correcta realización del proceso productivo. Se dividen en costos de administración de oficina central y en costos de oficina de campo.

Los costos de oficina central son los cargos técnicos y/o administrativos que representan la estructura ejecutiva, técnica, administrativa de una empresa como los honorarios o sueldos; rentas y/o depreciaciones de inmuebles y servicios necesarios para el buen desempeño de las funciones de la empresa, rentas de oficinas y almacenes, servicios de teléfono, energía eléctrica, internet, gastos de mantenimiento tanto de oficina, almacén y vehículos así como también las depreciaciones y gastos de la organización y de la instalación; obligaciones y seguros son los gastos necesarios y convenientes para la dilución de riesgos a través de seguros que impidan la súbita descapitalización por siniestros; materiales de consumo necesarios para la empresa como combustibles y lubricantes para automóviles al servicio de la oficina central , papelería, impresiones, copias, artículos de limpieza, pasajes, alimentos en la oficina; capacitación y promoción como clases de inglés, curso de algún programa de computación, congresos, exposiciones, actividades deportivas, celebraciones de oficina, regalos anuales a clientes y empleados, atenciones a clientes, concursos no obtenidos y proyectos no realizados; presentación de concursos, financiamiento, utilidad, fianzas.

Los costos de oficina de campo son los cargos de técnicos y administrativos que representan la estructura ejecutiva, técnica, administrativa de la obra como los sueldos, honorarios, viáticos de cada persona involucrada en el proyecto; transporte que es aplicable en el traslado a la obra foránea ya sea trasporte terrestre, aéreo o marítimo, mudanzas, peajes, combustibles, servicios, etc; comunicaciones y fletes son los gastos que tienen por objeto mantener comunicación entre la oficina central y la de obra, así como suministro de equipo idóneo a la bodega central, como gastos de teléfono, radio, internet, giros, transporte de equipo mayor y menor, mantenimiento, combustibles depreciaciones de vehículos; los gastos por construcciones provisionales son para proteger los intereses del cliente y de la empresa, así como para mejorar la productividad, como puertas, bardas, casetas, oficinas, bodegas y cubiertas, sanitarios, comedores, cocinas, instalaciones diversas, caminos de acceso, etc; los consumos energéticos en la etapa constructiva se requiere en mayor o menos escala, equipos especiales y requerimientos locales, también el consumo eléctrico provisional para el funcionamiento de la oficina, fotografías, papelería, rentas, servicios, cuotas sindicales, señalamientos y letreros.

2.2. Matriz. Partes y descripción.

El análisis de precios unitarios es un modelo matemático que adelanta el resultado, expresado en moneda, de unescenario relacionado con una actividad sometida a estudio. Cada partida del presupuesto tiene asociado uno o varios precios unitarios. Para conocer el PU se debe aplicar la metodologíaconocida como análisis de precio unitario. Este análisis está sometido al tiempo ya que se debe indicar la fecha del análisis, debido a que la inflación puede variar los precios de los insumos de una fecha a otra, se debe indicar el lugar geográfico donde se realiza la actividad a analizar porque también los precios de los insumos pueden variar de un lugar a otro; y conceptualizar las dimensiones

de lo que se va a construir porque la logística a aplicar es distinta al fabricar 1 casa que al fabricar 200 y a las condiciones del entorno como proveedores y características, usuario y características, normativa vigente, donde se realiza la obra. La mano de obra suele estar anclada a un tabulador de salarios, elemento que se deriva de una convención colectiva. De esta también se desprende el factor de costos asociados al salario, concepto que se explorará un poco más adelante. El analista también influye sobre algunos detalles que pueden ser de mucha importancia en los resultados finales, ya que su criterio al analizar la actividad estará presente constantemente.Este modelo matemático se basa en la agrupación de los componentes divididos en 3 apartados,materiales, mano de obra, herramienta y equipo. A pesar de ser un modelo matemático, que sugiere ser objetivo, desligado de sentimientos y otras influencias, incluye conceptos como el de rendimiento que se entiende como: la cantidad de obra realizada en un día, con el personaladecuado, utilizando las herramientas y equipos, en algunos casos son totalmente discrecionales y sometidos a cualquier clase de influencia, sobretodo en actividades no documentadas o no estudiadas. Análogamente, se incluyen el factor de rendimiento que pondera los renglones de equipos y mano de obra para racionalizarlos, porcentajes de costo indirecto e impuestos.

Materiales; cantidad correcta incluyendo, anclajes, traslapes y desperdicios, por ejemplo, la cantidad de azulejo para un único baño que tiene una área de 2.50 m2, y las cajas son de 1 m2 cada una, el desperdicios es del 20%. Los materiales adicionales, es lo que se requiere para la colocación del producto principal como pueden serlimpiadores, adhesivos, selladores, clavos, taquetes, tornillos, aditivos, bastidores, reducciones, adaptadores, etc. El rendimientos, para el caso de cimbras, tapiales, etc. en necesario determinar cuántos usos se le pueden dar, en el caso de la cimbra de contacto el promedio es de 4, las empresas de vivienda logran 8 usos, para el caso de los tapiales los usos van de 1 a 4, aunque el promedio es de 2. En cuanto a la ficha técnica, es importante localizarla sobre todo si no se está familiarizado con el producto, en ella se encontrara la manera correcta de emplearlo y su rendimiento, aunque este no siempre sea exacto como el caso de la pintura cuyo rendimiento cambia dependiendo de la superficie en la que se aplique, el tono de la misma con respecto al existente, etc.

Mano de Obra; se determina directamente por el rendimiento, este es uno de los datos más complicados a determinar, ya que depende deltrabajo y de las condiciones del mismo, además de que cada trabajador es diferente y por consiguiente el resultado también, los rendimiento se observan en obra y se determina un promedio; El promedio del rendimiento no en válido para todos los casos, por ejemplo; para un m2 de pintura el rendimiento es diferente si una casa es nueva o si está habitada, si la superficie es lisa o rugosa, si la altura promedio es de 2.40m o 6.00m. El factor de salario real (FSR) deberá estar integrado al salario de cada trabajador, este integra la relación de días pagados contra días laborables, días no laborables como son; festivos, sindicatos, por costumbre, mal tiempo y las prestaciones: aguinaldo, prima vacacional e IMSS. Este factor oscila entre un 64 y 73% dependiendo de diversos factores, a groso modo para un trabajador cuyo

sueldo es de 1200 pesos a la semana, el salario diario es $171.43, si su facto de salario es 72%, el salario real es de $171.43 x 1.72 = $294.86, con este se elaboran los PU.

Herramienta menor, la participación en las tarjetas de PU, se asigna como un factor que oscila entre un 2% y 3% de la mano de obra, en este porcentaje se contemplan; martillos, cinceles, seguetas, palas, picos, carretillas, brochas, talados de 1/2", remachadoras, cortadores de tubo, sopletes, etc.

Maquinaria, rentada o propia, la determinación del costo y rendimiento de cada máquina es primordial para lograr un costo competitivo, para ello es importante considerar el clima, las condiciones de trabajo, la dureza del terreno, el fletes de la maquinaria, los horarios de trabajo.

Ejemplo

Castillo de 15x15cm, de concreto hecho en obra de F'c=200 kg/cm2, acabado aparente,armado con 4 varillas de 3/8" y estribos del No.2 a cada 15 cm., incluye: materiales,acarreos, cortes, desperdicios, traslapes, amarres, cimbrado, coldado, descimbrado, manode obra, equipo y herramienta.	M	$243.01

ANALISIS DE PRECIO UNITARIO				
Concepto	**Unidad**	**Costo**	**cantidad**	**Importe**
MATERIALES				
VARILLA DE 3/8" 9.5 MM	KG	$10.40	2.6000	$27.04
ALAMBRON	KG	$11.90	1.0800	$12.85
ALAMBRE RECOCIDO	KG	$13.10	0.1200	$1.57
CLAVOS DE 2 A 4 "	KG	$15.00	0.1000	$1.50
DUELA DE PINO DE 3a DE 3/4"x3.5"x8.25"	PZA	$22.66	0.5000	$11.33
CHAFLAN DE PINO DE 1a DE 3/4"x3/4"x8.25"	PZA	$7.21	0.5000	$3.61
TRIPLAY DE PINO DE 16 MM	PZA	$298.70	0.0350	$10.45
Total Materiales				**$68.35**
MANO DE OBRA				
CUADRILLA No 5 (1 ALBAÑIL+1 PEON)	JOR	$837.32	0.1111	$93.04
Total Mano de obra				**$93.04**
EQUIPO Y HERRAMIENTA				
HERRAMIENTA MENOR	%	$93.04	0.0300	$2.79
Total Equipo y herramienta				**$2.79**
BASICOS				
CONCRETO DE F'c=200 KG/CM2, HECHO EN OBRA, T.M.A.= 19 MM, RESISTENCIA NORMAL	M3	$1,113.07	0.0240	$26.71
MATERIALES				
ARENA	M3	$176.57	0.5300	$93.58
GRAVA	M3	$176.57	0.6300	$111.24
AGUA (MANEJO)	M3	$18.54	0.2300	$4.26
CEMENTO GRIS	TON	$1,776.00	0.3900	$692.64
MANO DE OBRA				
CUADRILLA No 22 (1 ALBAÑIL + 5 PEONES)	JOR	$2,126.55	0.0830	$176.50
EQUIPO Y HERRAMIENTA				
HERRAMIENTA MENOR	%	$176.50	0.0300	$5.30
REVOLVEDORA P/CONCRETO DE 1 SACO 8 DE HP	HOR	$59.09	0.5000	$29.55
Costo directo				**$1,113.07**
Total Basicos				**$26.71**
COSTO DIRECTO				**$190.89**
INDIRECTOS	15%			$28.63
SUBTOTAL				$219.52
FINANCIAMIENTO	2.5%			$5.49
SUBTOTAL				$225.01
UTILIDAD	8%			$18.00
PRECIO UNITARIO				**$243.01**
DOSCIENTOS CUARENTA Y TRES PESOS 01/100 M.N.				

Tabla 2.1. Matriz de ejemplo, concepto castillo15x15

2.3. Cuantificación de obra.

A la fecha no existe un solo procedimiento para cuantificar un proyecto debido a las diferentes especialidades que intervienen, y de los requerimientos que nos pide el cliente ya sea persona física o institución, porque esto depende de la complejidad en el formato de presentación de la información.

Arquitectónicos; estos son los más sencillos ya que solo se cuantifican piezas, longitudes, áreas y volúmenes

Estructurales; son complicados debido a que no siempre están a escala y de ellos hay que obtener diferentes conceptos como lo son cimbra, acero y concreto, elementos que muchas veces no están dibujados y sólo en base a la experiencia y pericia del analista se obtiene la información.

Instalaciones; son diferentes ya que se requieren el conocimiento de cada especialidad y en el proyecto no se dibujan todos sus elementos como pueden serconexiones, reducciones, soportes, trayectorpias verticales, en el caso de la instalación eléctrica las líneas que representan los cableados no tienen la longitud real y se requiere adicionar cocas de conexiones.

La cuantificación de las cantidades involucradas con el fin de un análisis de precios unitarios de los conceptos, se hará necesario cuantificar en forma exacta las cantidades de obra involucradas.

Para asignar a un concepto la unidad correspondiente de peso, volumen, área o longitud, tomaremos en cuenta la unidad del integrante dominante, así como también la forma más fácil de llevar a cabo dicha medición. La unidad para dimensionar el concreto hidráulico debería ser la tonelada, ya que, al principal integrante es el cemento y este se estima en toneladas, sin embargo la dificultad de controlar en obra, esa medidanos conduce a la conveniencia de usar el metro cúbico.Cuando un elemento medido por volumen presenta condiciones de semiconstante una de sus medidas, es muy conveniente por facilidad de cálculo, dimensionarlo en metros cuadrados, por ejemplo las losas de concreto, o dimensionarlas en metro lineal, como el mismo concreto ahora utilizado en trabes o castillos. Las condiciones del presupuesto pueden variar en el transcurso de la obra, por lo cual en conveniente realizar las cubicaciones de tal manera sistematizadas, para que permitan revisarlas y entenderlas. Se sugiere resumir por partidas congruentes la cuantificación obtenida y concentrarla en un instrumento donde se documentan las cuantificaciones, a esta herramienta se le conoce como generador. Finalmente a la revisión parcial tanto numérica como de concepto, es recomendable una revisión global con base en los parámetros lógicos tales como la cantidad, espesor promedio, áreas semejantes de la cantidad en acabado, suma de recubrimientos semejantes, recomendando también en forma selectiva, cuantificación de elementos representativos o promedio para asignar limites mas precisos a nuestra revisión paramétrica.

2.4. Planos y Generadores.

Los planos son la representación gráfica y exhaustiva de todos los elementos que plantea un proyecto. Constituyen, los planos, la geometría plana de las obras proyectadas de forma que las defina completamente en sus tres dimensiones.

Los planos nos muestran cotas, dimensiones lineales superficiales y volumétricas de todas construcciones y acciones que comportan los trabajos los desarrollados por el proyectista y componen el documento del proyecto más utilizado a pie de obra.

Los planos son los documentos más utilizados de los que constituyen el proyecto y por ello deben incluir toda la información necesaria para poder ejecutar la obra en la forma más concreta posible y sin dar información inútil o innecesaria.

Los planos han de contener todos los detalles necesarios para la completa y eficaz representación de los trabajos. Las dimensiones en todos los planos, generalmente, se acotarán en metros y con dos cifras decimales. Como excepción, los diámetros de armaduras, tuberías, planos de taller, mobiliario, maquinaria, etc. las dimensiones se suelen acotar en mm.Deberá poder efectuarse, salvo en casos especiales, las mediciones de todos los elementos sin utilizar más dimensiones que las acotadas.

En particular, de no incluirse despiece detallado, deberá poderse deducir directamente de los planos, todas las dimensiones geométricas de los mismos, mediante las oportunas notas o especificaciones complementarias que las definan inequívocamente.

Los planos pueden ser generales y de detallé tanto para la ejecución de obra en campo como de los equipos en taller.Su número no debe prefijarse y habrá que realizar tantos planos como sean necesarios, teniendo en cuenta su uso casi exclusivo en la obra y a todos los niveles.

Normalmente los planos originales se depositan en el archivo de la Oficina Técnica, empleándose copia de los mismos, tanto para la tramitación legal del proyecto como para su ejecución.

En el plano de planta general se indican a escala[7] reducida todos los elementos del proyecto que nos permiten situar sus partes dentro de un todo. La planta general viene a ser una vista aérea del conjunto.Las escalas a utilizar para la planta general varían en función de las magnitudes de la obra proyectada.La planta, como proyección vertical, es indispensable para la definición geométrica de las obras proyectadas. El número de planos de planta de un proyecto puede ser numeroso y será tal que permita conocer con precisión y exactitud todo aquello que pretendemos construir.

[7] **Escala** es la relación entre la longitud del segmento dibujado y la longitud por él representada. Las escalas más utilizadas con 1:200, 1:125, 1:100, 1:75, 1:50, 1:25, sin embargo se puede emplear cualquier otra, tomando en cuenta que sea medible con algún instrumento disponible.

En los planos de planta deben situarse los servicios agua, electricidad, gas, teléfono, drenaje, etc; no obstante cuando la inclusión de estos servicios pueda confundir o complicar un plano de planta se repetirá su dibujo solo para aquellos cometidos, apareciendo de esta forma los planos que denominamos, planos de instalaciones.

Los alzados de una figura geométrica representan la proyección o vista horizontal de esa figura en sentido normal a sus distintos ejes.

El número de planos de alzado será función de las caras de la figura y de sus ejes de simetría. En una edificación, por ejemplo, habrá que dibujar tantos alzados como fachadas disponga. La escala a utilizar para los alzados debe ser semejante a las utilizadas para las plantas.

Las secciones tanto longitudinales como transversales son indispensables para conocer el interior de las piezas diseñadas y por tanto poder ejecutarlas. Las plantas y alzados por si solas no pueden definir un volumen irregular, para la dimensión tridimensional de una figura geométrica es preciso recurrir a las secciones.

Las escalas a utilizar en las secciones serán análogas a las manejadas en las plantas y en éstas además se debe indicar el lugar por donde se secciona.

En la mayoría de los proyectos es necesario desarrollar esquemas en perspectiva de isométricos, de las diferentes redes de distribución interior electricidad, agua, gas, aire comprimido, etc; para el dibujo de estos esquemas no se utiliza escala alguna.

En un proyecto no debe quedar ningún elemento por definir. Los detalles los podemos dibujar en el propio plano donde aparece el elemento a detallar o en un conjunto de planos que llamaremos planos de detalles.Todos estos detalles pueden ir incluidos en los planos de planta, sección o alzado. No obstante es preciso en ocasiones realizar planos concretos de detalle, tales como: detalles de carpintería: puertas y ventanas, las escalas utilizadas en los detalles varían entre 1:50 y 1:20.

El generador, es el documento con un formato diseñado para contener la información necesaria donde se detallan las operaciones aritmeticas con las cuales se obtienen cantidades de obra, es decir largo x ancho x alto, esto es dependiendo cual es tu unidad de medida del concepto que estas generando. Actualmente las dependencias y la mayoría de las empresas ya cuentan con un formato de generador de obra el cual tienelos siguientes datos; nombre, logotipo, dirección, teléfonos de la empresa y/o dependencia; los datos de la obra, dirección, título, alcance, fecha de concurso, fecha de netrega, número de licitación, fecha de presentación del generador; los datos del concepto, partida, código, descripción del concepto, unidad de medición, localización, referencia al plano, columnas de dimensiones, largo, ancho, alto, cantidad, área y volumen, y observaciones; además una referencia gráfica (croquis) para ubicación dentro del plano. Debe de estar conforme al catálogo de conceptos, ya que el generador respalda las cantidades que presentan y se toman para posteriormente

establecer el presupuesto; de la misma forma toda la información numérica contenida en el generador, proviene de los planos y por ello la importancia de tener un proyecto completo porque de esto depende que se obtengan las cantidades reales y fiables; de lo contrario nos representaría perdidas y complicaciones en la ejecución de la obra. En la generalidad de las empresas y en la actualidad es muy demandante la rapidez con que se debe obtener dicha información y a su vez reflejarla en el presupuesto, por ello la importancia de tener una herramienta de apoyo para la cuantificación y generación de cantidades de obra, con la disposición de ser administrada de rápidamente entre las diferentes partes involucradas en el proceso de costos de la construcción.

2.5. Métodos de Cuantificación.

Se define así al conjunto ordenado de datos obtenidoso logrados mediante la interpretación y medición de los planos, los cuales como característica principal deben estar acotados preferentemente,y con pocas excepcionesapoyarse de una herramienta para deducir dimensiones como utilizando el escalímetro.

Las cuantificaciones se realizancon el objeto de calcular la cantidad de obra a realizar y queal ser multiplicado por los respectivos costos unitarios y sumados,obtendremos el costo directo.Se han establecido criterios y procedimientosuniformes respecto a la cuantificación de partidas para obras deedificación.

Se debe efectuar un estudio integral de los planos yespecificaciones técnicas del proyecto, relacionandoentre sí los planos de Arquitectura, Estructuras, Instalaciones sanitarias, hidráulicas y Eléctricas para el caso de edificación. Precisar la zona de estudio o de las cuantificaciones y trabajosque se van a ejecutar.El orden para elaborar generadores es primordial porquenos dará la secuencia en que se toman las medidas olecturas de los planos, enumerándose las páginas enlas cuales se escriben las cantidades incluyéndose lasobservaciones pertinentes. Todo esto nos dará lapauta para realizar un chequeo más rápido y poderencontrar los errores de ser el caso.

Es recomendable pintar con diferentes colores loselementos o áreas que se están midiendo para que deesta manera se pueda simplificar la revisión respectiva; así por ejemplo, en el caso de muros divisorios se puede marcar de color rojo, muros de carga decolor verde, los castillos de color azul ylas columnas de color amarillo; esto permite visualizar los elementos que se están cuantificando de manera directa.

CAPITULO 3. DESCRIPCION DE LOS COMANDOS DE VISUAL BASIC 6.0

Objetivo

Describir los comandos de Visual Basic 6.0 utilizados para la realización del programa de cuantificación; objetos, estructura, métodos, comandos de texto y herramientas visuales.

3.0. Introducción a Visual Basic 6.0

Visual Basic 6.0es uno de los lenguajes de programación que más entusiasmo despiertan entre los programadores de PCs, tanto expertos como novatos. En el caso de los programadores expertos por la facilidad con la que desarrollan aplicaciones complejas en poquísimo tiempo (comparado con lo que cuesta programar en Visual C++, por ejemplo). En el caso de los programadores novatos por el hecho de ver de lo que son capaces a los pocos minutos de empezar su aprendizaje. El precio que hay que pagar por utilizar Visual Basic 6.0 es una menor velocidad o eficiencia en las aplicaciones.

Visual Basic 6.0 es un lenguaje de programación visual, también llamado lenguaje de 4ª generación. Esto quiere decir que un gran número de tareas se realizan sin escribir código, simplemente con operaciones gráficas realizadas con el ratón sobre la pantalla. Es también un programa basado en objetos, aunque no orientado a objetos como C++ o Java. La diferencia está en que Visual Basic 6.0 utiliza objetos con propiedades y métodos, pero carece de los mecanismos de herencia y polimorfismo propios de los verdaderos lenguajes orientados a objetos como Java y C++.

Visual Basic 6.0 está orientado a la realización de programas para Windows, pudiendo incorporartodos los elementos de este entorno informático: ventanas, botones, cajas de diálogo y de texto,botones de opción y de selección, barras de desplazamiento, gráficos, menús, etc.Prácticamente todos los elementos de interacción con el usuario de los que dispone Windows en cualquiera de sus versiones pueden ser programados en Visual Basic 6.0 de un modo muy sencillo. En ocasionesbastan unas pocas operaciones con el ratón y la introducción a través del teclado de algunassentencias para disponer de aplicaciones con todas las características de Windows. En lossiguientes apartados se introducirán algunos conceptos de este tipo de programación.

3.1. Módulos y formularios.

Cada uno de los elementos gráficos que pueden formar parte de una aplicación típica de Windowses un tipo de control: los botones, las cajas de diálogo y de texto, las cajas de seleccióndesplegables, los botones de opción y de selección, las barras de desplazamiento horizontales yverticales, los gráficos, los menús, y muchos otros tipos de elementos son controles para VisualBasic 6.0. Cada control debe tener un nombre a través del cual se puede hacer referencia a él en elprograma. Visual Basic 6.0 proporciona nombres por defecto que el usuario puede modificar. En la terminología de Visual Basic 6.0 se llama formulario (form) a una ventana. Unformulario puede ser considerado como una especie de contenedor para los controles. Unaaplicación puede tener varios formularios, pero un único formulario puede ser suficiente para lasaplicaciones más sencillas. Los formularios deben también tener un nombre, que puede crearsesiguiendo las mismas reglas que para los controles.

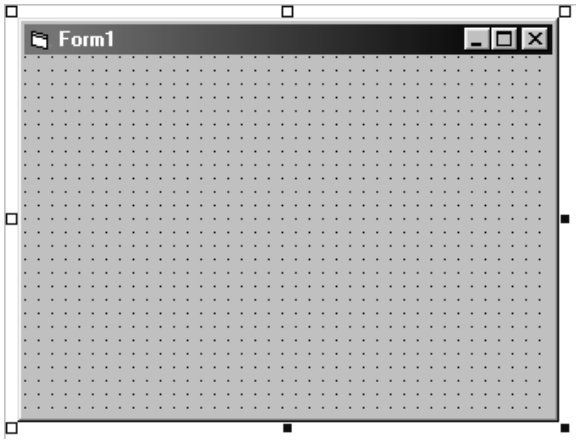


Figura 3.1. Formulario

El formulario es una de las partes más importantes, ya que es donde se diseña lapantalla o pantallas que formarán parte de nuestro programa, como se muestra en la figura 3.1. A estas pantallas lellamaremos formularios. Aquí iremos agregando y modificando los diferentes elementosde nuestra aplicación, como puedan ser botones, cuadros de texto, etc. Si no viéramosla pantalla del formulario podríamos activarla desde Ver. Objeto o pulsar Mayúsculas + F7.

El diseño de una pantalla es tan simple como arrastrar los objetos que deseamos,desde el cuadro de herramientas hasta el formulario. Para modificar el tamaño de cualquier objeto, incluso del formulario solo es necesario situarse en cualquierade las esquinas del objeto o en el centro de uno de sus lados marcados con un cuadrado,esperar que el ratón se convierta en una fecha de desplazamiento, pulsar el botónizquierdo del ratón y mientras se mantiene pulsado movernos hasta que el objeto tomeun nuevo tamaño. Si cambiamos el tamaño desde uno de los vértices podremos modificartanto el alto como el ancho, mientras que si arrastramos desde uno de los ladossolo podremos modificar el alto o el ancho dependiendo del lado en el que nos encontremos.

Los formularios y los distintos tipos de controles son entidades genéricas de las que puede habervarios ejemplares concretos en cada programa. En programación orientada a objetos se llama clase a estas entidades genéricas, mientras que sellama objeto a cada ejemplar de una clase determinada. Por ejemplo, en un programa puede habervarios botones, cada uno de los cuales es un objeto del tipo de control command button, que sería laclase.

Cada formulario y cada tipo de control tienen un conjunto de propiedades que definen su aspecto gráfico como tamaño, color, posición en la ventana, tipo y tamaño de letra, etc. y su forma deresponder a las acciones del usuario. Cada propiedad tiene unnombre que viene ya definido por el lenguaje.Por lo general, las propiedades de un objeto son datos que tienen valores lógicos (True, False)o numéricos concretos, propios de ese objeto y distintos de las de otros objetos de su clase. Así pues, cada clase, tipo de objeto o control tiene su conjunto de propiedades, y cada objeto o controlconcreto tiene unos valores determinados para las propiedades de su clase.Casi todas las propiedades de los objetos pueden establecerse en tiempo de diseño y tambiénen tiempo de ejecución. En este segundo caso se accede a sus valores por medio delas sentencias del programa, en forma análoga a como se accede a cualquier variable en un lenguajede programación. Para ciertas propiedades ésta es la única forma de acceder a ellas.

Por supuestoVisual Basic 6.0 permite crear distintos tipos de variables. Se puede acceder a una propiedad de un objeto por medio del nombre del objeto a quepertenece, seguido de un punto y el nombre de la propiedad, como por ejemplo Treeview.objName.

El módulo VBA de AutoCAD es ligeramente distinto al entorno Visual Basic habitual, sies que estamos familiarizados con él. En principio, por defecto el Explorador de proyectos y laVentana de Propiedades aparecen anclados a la izquierda, y no a la derecha. El Cuadro deherramientas no aparece por ningún lado, ya que sólo se pone de manifiesto cuando estamostrabajando con un formulario.En general la estructura de menús y barra de herramientas es muy parecida, si bienexisten ausencias —sobre todo— en VBA con respecto a Visual Basic, como por ejemplo lacapacidad de incluir formularios MDI, de compilar proyectos o de abrir proyectos, entre otras.Esto último se realiza desde el menú desplegable de AutoCAD, y sólo desde ahí. Decir que elmódulo VBA únicamente puede ser abierto desde una sesión de AutoCAD, y noindependientemente.Todas la nuevas características se irán aprendiendo a lo largo de este MÓDULO, por loque no debemos preocuparnos si dominamos Visual Basic y creemos no sentirnos a gusto enVBA.Dado que, como hemos dicho, se suponen conocimientos previos por parte del lector,nos meteremos de lleno ya mismo con la programación exclusivamente dirigida a AutoCAD.

3.2. Objetos y comandos.

En este apartado se describen los objetos y comandos útiles para este trabajo, porque a consideración no es necesario describir todo el lenguaje de programación que Visual Basic utiliza para AutoCAD. Los objetos ActiveX que proporciona AutoCADpara su manejo desde programas VBA están divididos según una jerarquía que deberemos seguir a la hora de llamarlos o referirnos a ellos. La figura 3.2 será muy útil a la hora de programar, ya que establece dicha jerarquía.

En Visual Basic es factible añadir al entorno nuevos objetos creados por nosotros para luego ser utilizados. Lo que se ha hecho en VBA es precisamente eso. Estos objetos tienen sus propiedades y métodos, al igual que los demás. Existen objetos de entidades individuales de dibujo (líneas, círculos, arcos...) con sus propiedades (color, capa, tipo de línea...) y métodos (copiar, mover, escalar).

También se han definido otros objetos no gráficos como son el Espacio Modelo, el Espacio Papel y los bloques. Estos se consideran una colección de objetos de entidades individuales de dibujo y tienen también sus propiedades para, por ejemplo, saber cuántas entidades simples contienen, y sus métodos para, por ejemplo, añadir nuevas entidades a la colección.

El propio documento actual de AutoCADestá definido como un objeto y tiene sus propiedades (camino de acceso, límites...) y métodos (guardar, regenerar...). Dentro de él se encuentran los mencionados anteriormente, además de otras colecciones como el conjunto de capas, de estilos de texto, etcétera, cada una con propiedades y métodos.

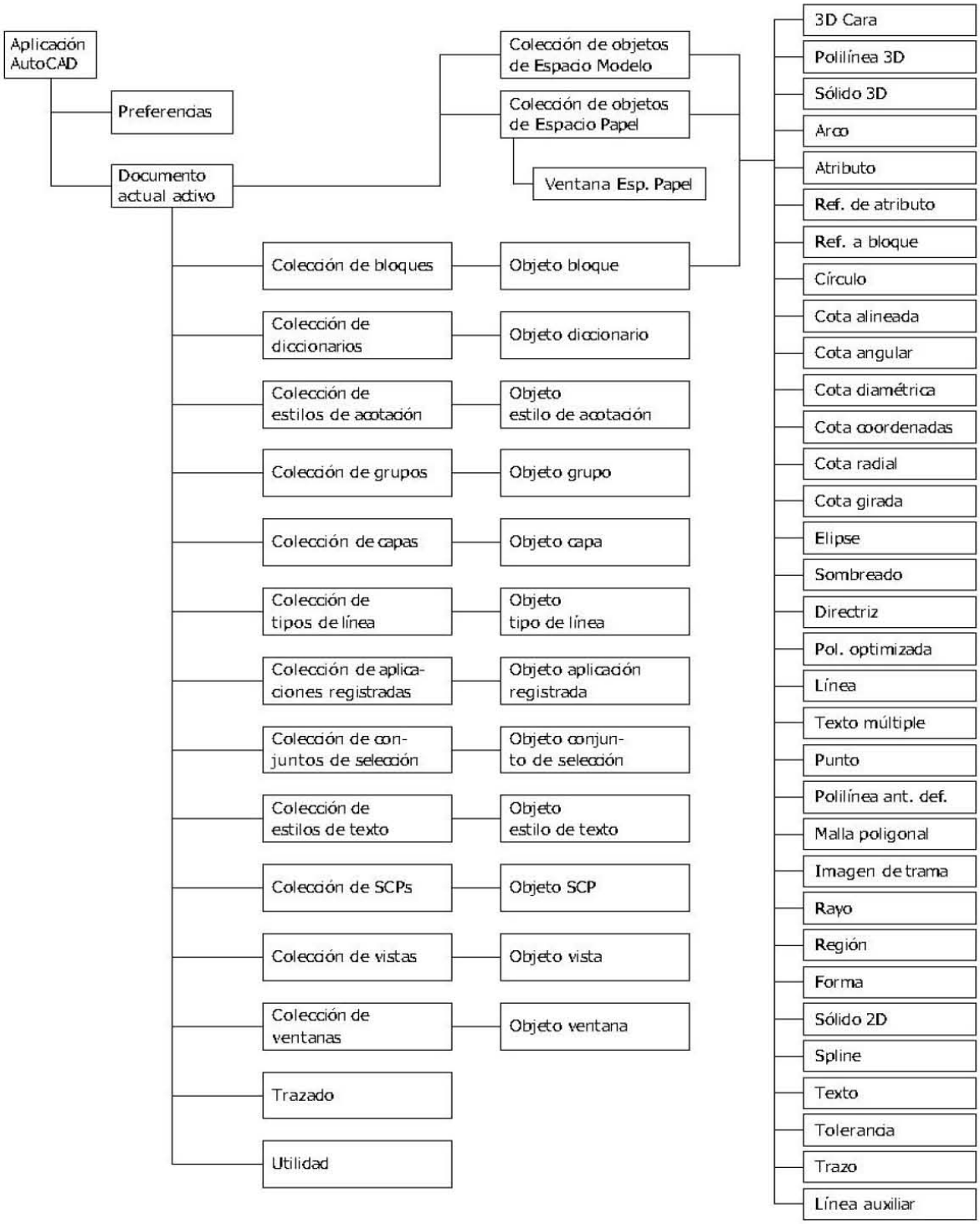


Figura 3.2. Propiedades y objetos de Autocad

A continuación, bajo esta sección de dibujo y representación de entidades, se muestra los diferentes métodos que tenemos deañadir a las entidades de dibujo mediante rutinas de programación. La manera de dibujar entidades dice relación a métodos que pertenecen a lascolecciones de Espacio Modelo, Espacio Papel y bloques (como se ve en la plantilla de objetos).

Línea

La sintaxis que se utiliza en este módulo es: instrucciones, funciones, métodos, propiedades y demás términos reservados como parecen en el editor; los textos en cursiva son mnemotécnicos[8] que han de ser sustituidos por su valor; los listados de programas se muestran como aparecen en el editor teniendo previamente la declaración de las variables.

La sintaxis del método **AddLine**para dibujar líneas es la que sigue:

Set ObjLínea = ObjColección.AddLine(DblPtoInicial, DblPtoFinal)

Propiedades de los objetos de línea:

Application, Color, EndPoint, EntityName, EntityType, Handle, Layer, Linetype, LinetypeScale, ObjectID, StartPoint, Thickness, Visible

Métodos de los objetos de línea:

ArrayPolar, ArrayRectangular, Copy, Erase, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Offset, Rotate, Rotate3D, ScaleEntity, SetXData, TransfomBy, Update

Círculo

El método **AddCircle**permite dibujar círculos en la colección de objetos de EspacioModelo, Espacio Papel o formando parte de un bloque. Hay que indicar las coordenadas delpunto del centro, que será un matriz de tres elementos (coordenadas X, Y y Z) de tipoDouble, y el radio del círculo, que también será Double. La sintaxis de AddCircle es:

Set ObjCirculo = ObjColección.AddCircle(DblCentro, DblRadio)

Propiedades de los objetos de círculo:

Application, Area, Center, Color, EntityName, EntityType, Handle, Layer, Linetype, LinetypeScale, Normal, ObjectID, Radius, Thickness, Visible,

Métodos de los objetos de círculo:

[8]Procedimiento de asociación mental para facilitar el recuerdo de algo

ArrayPolar, ArrayRectangular, Copy, Erase, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Offset, Rotate, Rotate3D

Elipse

El método AddEllipse permite dibujar elipses o arcos de elipse en la colección deobjetos de Espacio Modelo, Espacio Papel o en la colección de bloques. La sintaxis deAddEllipse es:

Set ObjElipse = ObjColección.AddElipse(DblCentro, DblPtoEjeMayor, DblRelación)

El centro ha de indicarse como una matriz de elementos Double (DblCentro);DblPtoEjeMayor es un punto (matriz de tres valores Double) en uno de los extremos del ejemayor, considerando las coordenadas en el SCO y relativas al centro; DblRelación es unvalor Double que es la relación entre la medida del eje menor y del eje mayor (valor máximoigual a 1 se corresponde con un círculo).

Propiedades de los objetos de elipse:

Application, Area, Center, Color,EndAngle, EndParameter,EndPoint, EntityName, EntityType, Handle, Layer, LineType, LinetypeScale,MajorAxis, MinorAxis, Normal, ObjectID,RadiusRatio, StartAngle, StartParameter, StartPoint, Visible

Métodos de los objetos de elipse:

ArrayPolar, ArrayRectangular, Copy, Erase, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Offset, Rotate, Rotate3D, ScaleEntity, SetXData, TransformBy, Update

Arco

El método AddArc permite crear un arco de circunferencia que se dibuja desde el puntoinicial al final en sentido contrario a las agujas del reloj. Estos puntos se calculan a partir de las propiedades StartAngle, EndAngle y Radius.Para crear un arco hay que indicar el centro (array de tres valores Double), el radio(Double), el ángulo inicial (Double) en radianes y el ángulo final (Double) también enradianes. La sintaxis de AddArc es:

Set ObjArco = ObjColección.AddArc(DblCentro, DblRadio, DblÁngInic, DblÁngFin)

Propiedades de los objetos de arco:

Application, Area, Center, Color, EndAngle, EndPoint, EntityName, EntityType, Handle Layer, LineType, LinetypeScale, Normal, ObjectID, Radius, StartAngle, StartPoint, Thickness, Visible

Métodos de los objetos de arco:

ArrayPolar, ArrayRectangular, Copy, Erase, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Offset, Rotate, Rotate3D, ScaleEntity, SetXData, TransformBy, Update

Texto en una línea

El método AddText permite añadir texto en una línea al dibujo. Hemos de indicar lacadena de texto en cuestión (variable String), el punto de inserción en el SCU (matriz Doublede tres valores) y la altura (Double). La sintaxis es pues que sigue:

Set ObjTexto = ObjColección.AddText(StrTexto, DblPtoIns, DblAltura)

Propiedades de los objetos de texto:

Application, Color, EntityName, EntityType, Handle,Heigth, HorizontalAlignment, InsertionPoint, Layer, LineType, LinetypeScale, Normal, ObjectID,ObliqueAngle, Rotation, ScaleFactor, StyleName, TextAlignmentPoint, TextGenerationFlag, TextString, Thickness,VerticalAlignment,Visible,

Métodos de los objetos de texto:

ArrayPolar, ArrayRectangular, Copy, Erase, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Rotate, Rotate3D, ScaleEntity, SetXData, TransformBy, Update

Polilínea

Para crear polilíneas se utiliza el métodoAddLightWeightPolyline, cuya sintaxis es la que sigue:

Set ObjPolilínea = ObjColección.AddLightWeightPolyline(DblMatrizVértices)

Propiedades de los objetos de polilínea optimizada:

Application, Area, Closed, Color, Coordinates, EntityName, EntityType, Handle, Layer, LineType, LinetypeScale, Normal, ObjectID, Thickness, Type, Visible,

Métodos de los objetos de polilínea optimizada:

AddVertex,ArrayPolar, ArrayRectangular, Copy, Erase, Explode, GetBoundingBox, GetBulge, GetWidth, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Offset, Rotate, Rotate3D, ScaleEntity, SetBulge, SetWidth, SetXData, TransformBy, Update

Texto múltiple

Para añadir texto múltiple a la colección de objetos de Espacio Modelo, Espacio Papel obloques, disponemos del método AddMText, cuya sintaxis es:

Set ObjTextoM = ObjColección.AddMText(DblPtoIns, DblAnchura, StrTexto)

Propiedades de los objetos de texto múltiple:

Application, AttachmentPoint, Color, DrawingDirection, EntityName, EntityType, Handle, Height, InsertionPoint, Layer, LineType, LinetypeScale, Normal, ObjectID, Rotation, StyleName, TextString, Visible, Width,

Métodos de los objetos de texto múltiple:

ArrayPolar, ArrayRectangular, Copy, Erase, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Rotate, Rotate3D, ScaleEntity, SetXData, TransformBy, Update

Así como en el dibujo de textos en una línea, DblPtoIns es una matriz o tabla de treselementos Double que son las coordenadas del punto de inserción, y StrTexto es la cadena literal de texto (String) que se imprimirá en pantalla. DblAnchura se refiere a la anchura dela caja de abarque del texto múltiple (es valor Double).

Región
El método para la obtención de regiones es AddRegion y su sintaxis es:

Set ObjRegión = ObjColección.AddRegion(ObjListaObjetos)

Propiedades de los objetos de región:

Application, Area, Centroid, Color, EntityName, EntityType, Handle, Layer, LineType, LinetypeScale, MomentOfInertia, Normal, ObjectID, Perimeter, PrincipalDirections, PrincipalMoments, ProductOfInertia, RadiiOfGyration, Visible,

Métodos de los objetos de región:

ArrayPolar, ArrayRectangular, Boolean, Copy, Erase, Explode, GetBoundingBox, GetXData, Highlight, IntersectWith, Mirror, Mirror3D, Move, Rotate, Rotate3D, ScaleEntity, SetXData, TransformBy, Update

Este método crea regiones a partir de una lista de objetos. Los objetos de esta listadeberán formar un área coplanar cerrada y sólo podrán ser líneas, arcos, círculos, arcoselípticos, polilíneas y/o splines.La lista de objetos será una matriz o array declarada como Object y que contendrátodas las entidades gráficas, de las cuales se dibujará la región.

3.3. Referencias.

Visual basic es la aplicación que administra y funcióna de manera central,tiene la facultad para administrar los diversos programas instalados de la PC en uso; para controlar las aplicaciones externas es necesario hacer "referencia" a las librerías de dicho programa; esta herramienta se localiza en el menú "Proyecto". Sólo se tiene que buscar

en la lista de librerías que aparecen y se selecciona la más adecuada; con esto podemos utilizar todas las funciones que contiene dicha librería. Únicamente tenemos que cuidar la compatibilidad de versión que estamos usando para que no tengamos conflictos internos del programa.

Para este trabajo la librería que se requiere; de Excel es Microsoft Excel 14.0 Object Library y de Autocad es Autocad 2012 Type Library.

3.4. Cuadro de herramientas y componentes.

Cuadro de herramientas. En este cuadro encontramos las herramientas que podemos utilizar para diseñar nuestro proyecto. El cuadro de herramientas que presentamos a continuación es el estándar, el cual contiene los elementos básicos. A continuación vamos a nombrar las herramientas básicas, de izquierda a derecha según la figura 3.3.Para visualizar el cuadro de herramientas podremos ir a la opción Cuadro de herramientas dentro de la opción Ver.

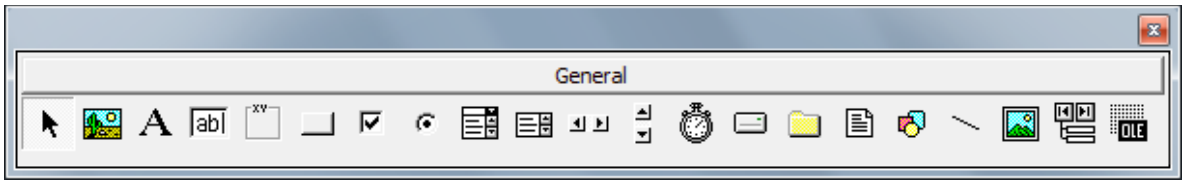


Figura 3.3. Cuadro de herramientas de Visual Basic

Puntero. Utilizaremos este control para poder mover, cambiar el tamaño o seleccionar los diferentes elementos que insertemos en el formulario.

PictureBox. Lo podemos utilizar para mostrar imágenessueltas, aunque suele utilizarse como contenedor de otros elementos. Esto es que dentro de un cuadro de imagen pueden existir otros elementos quedependen de él. Si nosotros movemos el cuadro de imagen con elementos en el interior,todos ellos se moverán junto con él. Si mirásemos la propiedad Top y Left de cualquierelemento que está insertado dentro de un cuadro de imagen veríamos que están enrelación con el borde de este y no con el borde del formulario como en la gran parte delos objetos.

Label. En las etiquetas o labels la propiedad más importante es Caption, que contiene el texto que aparece sobre este control. Esta propiedad puede ser modificada desde programa, pero no interactivamente clicando sobre ella (a diferencia de las cajas de texto, que se verán a continuación). Puede controlarse su tamaño, posición, color de fondo y una especie de borde 3-D. Habitualmente las labels no suelen recibir eventos ni contener código. Utilizaremos este control para escribir etiquetas donde aparecerá texto que elusuario no podrá cambiar.

TextBox. La propiedad más importante de las cajas de texto es Text, que almacena el texto contenido en ellas. También se suelen controlar las que hacen referencia a su tamaño, posición y apariencia. En algún momento se puede desear impedir el acceso a la caja de texto, por lo que se establecerá su propiedad Enabled como False. La propiedad Locked como True hace que la caja de texto sea de sólo lectura. La propiedad MultiLine, que sólo se aplica a las cajas de texto, determina si en una de ellas se pueden incluir más de una línea o si se ignoran los saltos de línea. La justificación o centrado del texto se controla con la propiedad Alignment.Son cuadros de texto que el usuario podrá cambiar.

Frame. Para agrupar varios controles, cuando se cambia de lugar el frame, todo su contenido lo hará.

CommandButton. Utilizaremos este control para crear botones sobre los cuales podráactuar el usuario.

CheckBox. Casilla que el usuario podrá utilizar para marcar dos posibles opciones.Verdadero o falso, sí o no, activado, desactivado. El usuario podrá marcar la cantidadde casillas de verificación que desee dentro de una aplicación.

OptionButton. Muy parecida al control anterior, pero el usuario solo podrá marcar unade las opciones. Si tenemos dos controles de este tipo, en el momento de seleccionaruno automáticamente se quitará la selección el otro.

ComboBox tiene características comunes de un TextBox y de un ListBox. Un TextBox ya que se puede escribir texto en el recuadro de texto que aparece y de un ListBox ya que podemos seleccionar uno de los elementos que aparecen en la lista desplegable de dicho control.

ListBox es un elemento que nos muestra una lista de elementos de los queel usuario de la aplicación puede escoger cualquiera de ellos. Normalmente si el númerode elementos que hay dentro de la lista excede del espacio que hemos reservado parala visión del contenido de esta aparecen unas barras de desplazamiento para podernosmover con facilidad sobre la lista.Cada elemento de la lista tiene un número que nos indica el lugar que ocupa. Esmuy importante tener en cuenta que el primer elemento de la lista tiene como índice 0.La propiedad que nos indica el índice de cada elemento es ListIndex. Si no hay ningúnelemento en la lista el valor de esta propiedad es -1.

ScrollBars. En este tipo de control las propiedades más importantes son Max y Min, que determinan el rango en el que está incluido su valor, LargeChange y SmallChange que determinan lo que se modifica su valor al dar clic en la barra o en el botón con la flecha respectivamente y Value que determina el valor actual de la barra de desplazamiento. Las barras de desplazamiento no tienen propiedad Caption. El evento que se programa habitualmente es Change, que se activa cuando la barra de desplazamiento modifica su valor. Todo lo comentado es para las barras de desplazamiento vertical y horizontal.

Timer. Si se desea que una acción suceda con cierta periodicidad se puede utilizar un control Timer. Este control produce de modo automático un evento cada cierto número de milisegundos y es de fundamental importancia para crear animaciones o aplicaciones con movimiento de objetos. La propiedad más importante de un objeto de este tipo es Interval, que determina, precisamente, el intervalo en milisegundos entre eventos consecutivos.

DriveListBox. Tiene una propiedad llamada Drive que recoge la unidad seleccionada por el usuario, puede ser una unidad física como el disco c:\ o una unidad lógica asignada por el usuario a otro disco o directorio en un servidor o en otro ordenador de la red.

DirListBox. La propiedad pathdetermina el drive seleccionado y sus directorios se muestran en dicha una ventana.

FileListBox. Muestra los archivos del la carpeta seleccionada.

Shape. Inserta un dibujo en el formulario indicado, puede ser rectángulo, elipse, círculo o cuadro, tiene propiedades modificables en el borde y en el relleno.

Line. Inserta una línea en el formulario indicado, tiene propiedades modificables en el borde y en el relleno.

Image. Inserta un objeto y muestra las imágenes, se cargan con mayor rapidez y consume menos memoria.

Data.Permite crear una conexión o enlace a las bases de datos, para leer, almacenar y modificar registros, por ejemplo hacer un reporte.

Ole.Permite enlazar un archivo externo a VB a un programa creado por nosotros.

3.5. Base de datos.

Podemos indicar que una base de datos se define como un conjunto de información relacionada que se encuentra agrupada o estructurada; el archivo por sí mismo, no constituye una base de datos, sino más bien la forma en que está organizada la información es la que da origen a la base de datos.

Una base de datos es un sistema formado por un conjunto de datos almacenados, estructurados, fiables y homogéneos,que permiten el acceso directo a ellos y un conjunto de programas que manipulan ese conjunto de datos.

En la terminología propia de las bases de datos hay tres conceptos claves dentro de las tablas: campo, registro y dato.Un campo es cada uno de los tipos de datos que se van a usar.Se hace referencia a los campos por su nombre.Un registro está formado por el conjunto de información en particular. Un dato es la intersección entre un campo y un registro.

Tablas

Las tablas con el componente básico o elemental de las bases de datos. O lo que es lo mismo, una base de datos está principalmente compuesta por varias tablas relacionadas. Las tablas contienen datos sobre algo o alguien, proveedores, clientes, libros en una biblioteca, compras, ventas, etc.

Consultas

Las consultas son preguntas que un usuario hace a la base de datos. Con ellas puede obtener información de varias tablas y con la estructura que más le interese. Además, las consultas pueden archivarse de forma que la próxima vez que se quiera hacer la misma pregunta no tendrá que volver a plantearla, será suficiente con llamar a la consulta previamente creada. La importancia de las consultas es enorme, de hecho es la potencia de esta herramienta la que permite que los gestores de base de datos sean casi imprescindibles en nuestro trabajo diario.

Formularios

Los formularios son un mecanismo que facilita enormemente la operatoria general con tablas, principalmente a la hora de mostrar, introducir y modificar datos. Un uso adecuado de éstos redunda bastante en el nivel de manejabilidad de una aplicación o de un sistema de información desarrollado con Access.

Informes

Los informes permiten presentar la información con una apariencia altamente profesional a la hora de imprimir nuestros datos.

Páginas de acceso a datos

Una página de acceso a datos es una página Web que se puede utilizar para agregar, modificar, ver o manipular datos actuales en una base de datos. Se pueden crear páginas que se utilizarán para especificar y modificar datos, de manera similar a los formularios de Access. También se pueden crear páginas que muestren registros agrupados jerárquicamente, de manera similar a los informes de Access.

CAPITULO 4. DESARROLLO DEL PROGRAMA "CACHO"

Objetivo

Diseño de la estructura del programa; descripción de ventanas, botones, rutinas de programación y procesos propios para su funcionamiento.

4.1. Diseño y descripción de ventanas.

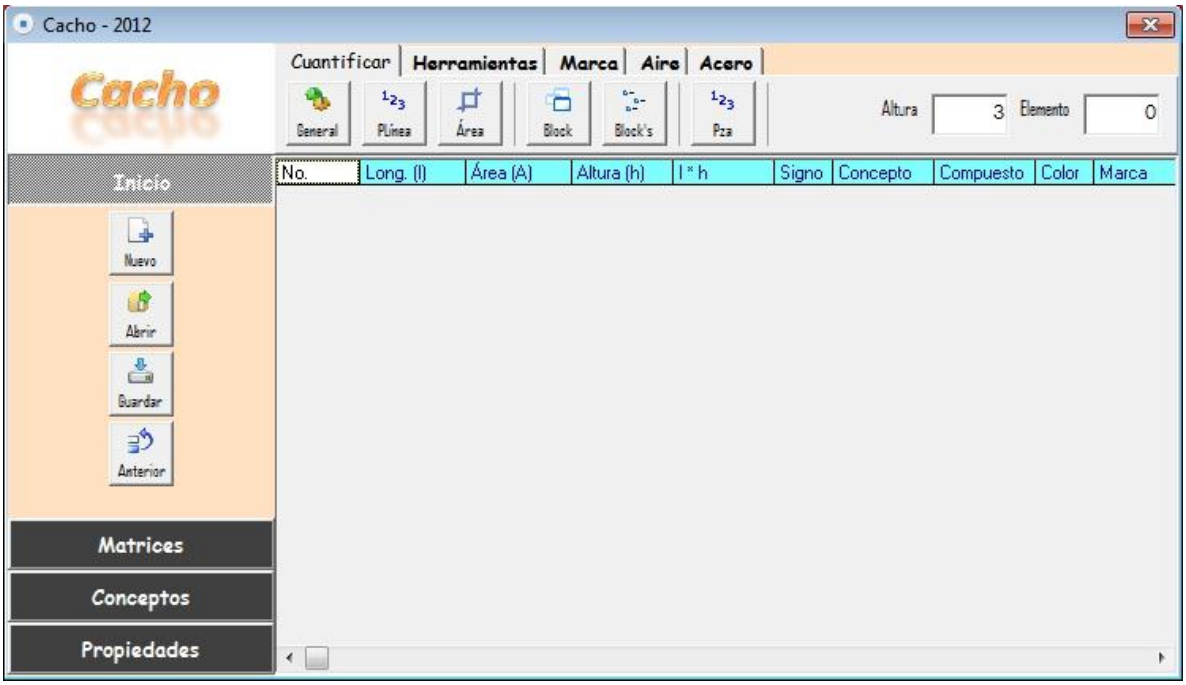


Figura 4.1. Ventana principal

Esta ventana, figura 4.1., cuenta con tres principales elementos, en la parte superior el menú para cuantificar en sus diversas modalidades; ***Cuantificar, Herramientas, Marca,*** las partidas especiales ***Aire*** *y* ***Acero*** (las partidas especiales pueden variar dependiendo de las necesidades del cliente). Del lado izquierdo tenemos un menú donde administramos los archivos de trabajo y son las funciones más recurrentes al cuantificar; ***Inicio, Matrices, Conceptos, Propiedades***, las cuales se describen más adelante. Al centro de la ventana una tabla donde se visualizan los datos cuantificados con sus dimensiones y diversas características.

Tabla

Elemento donde se muestran y se ordenan los datos de los objetos cuantificados. Sus columnas son de izquierda a derecha ***No.*** (Número consecutivo de los elementos cuantificados y asocia a la marca que se deja en Autocad) ***Longitud***(es la dimensión del objeto medido como línea, círculo, polilínea, arco; los cuales pueden representar tramos de tubería, cable, tramos de elementos estructurales, etc.)***, Área***(es la dimensión del objeto medido como el círculo, la polilínea de más de dos puntos, el arco, etc.)***, Altura*** (dimensión de apoyo para obtener área vertical como pueden ser los metros cuadrados de un muro)***, Lxh (área vertical)*** (en esta columna está el resultado de multiplicar la columna longitud (L) y la columna altura (h) y es para representar áreas de forma vertical)***, Signo*** (por default tenemos que el signo aparece positivo pero podemos cambiar dándole doble clic con el botón del mouse, porque en ocasiones podemos

encontrarnos con superficies que les debemos descontar otras áreas)**, *Compuesto*** (si aparece el valor cero nos indica que nuestro concepto es simple y si el valor es uno nos indica que nuestro concepto es una matriz)**, *Color*** (es la característica que se ha asignado en la pestaña de propiedades y se aplica a los elementos cuantificados)**, *Marca*** (es la característica que se ha asignado en la pestaña de propiedades y se aplica a los elementos cuantificados)**, *Objeto(***esel tipo de elemento cuantificado) ***y Id. Objeto***(nombre único del elemento cuantificado).

Sobre la tabla con el botón derecho del mouse podemos visualizar un ***menú contextual*** donde aparecen varias funciones rápidas.

Al dar doble clic sobre algún valor útil de la tabla, dicho valor se almacena en el portapapeles y se puede utilizar en cualquier aplicación.

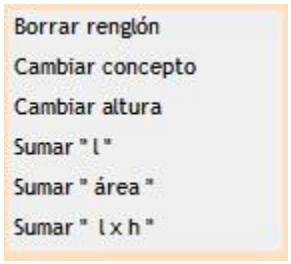


Figura 4.2. Menú contextual

Borrar renglón. Este comando lo utilizamos cuando deseamos eliminar de la tabla de objetos cuantificados que ya no son de utilidad y los cuales no se almacenarán al momento de guardar. Para borrar conceptos debemos seleccionar de la tabla el rango de conceptos, estos indicaran en color, presionamos el botón derecho del mouse y seleccionamos ***Borrar renglón***, se muestra una pantalla para confirmar el comando; ahora en la primera columna notaremos que ya no existen los objetos seleccionados. Después de este proceso debemos dar guardar para que tengan efecto los cambios.

Cambiar concepto. Cuando hemos cuantificado algún elemento y no es correcta la asignación, podemos cambiar dicho concepto seleccionando de la tabla la lista a modificar, presionamos el botón derecho del mouse y seleccionamos ***Cambiar concepto,*** aparecerá una lista, seleccionamos el nuevo concepto y le damos Asignar; después de este proceso debemos dar guardar para que tengan efecto los cambios.

Cambiar altura. Para conceptos que son de área vertical, como lo pueden ser muros, pintura y acabados, podemos cambiar la altura ya establecida, seleccionamos de la tabla la lista a modificar, presionamos el botón derecho del mouse y seleccionamos ***Cambiar altura***, aparecerá una solicitud para asignar la nueva dimensión a modificar y aceptar se cambia automáticamente la columna de ***Altura (h)*** y ***Lxh***

Sumar L, Sumar área y Sumar Lxh. Cuandonecesitamos saber la suma de algunos conceptos, seleccionamos horizontalmente el rango a operar, posteriormente

presionamos el botón derecho del mouse y seleccionamos la columna que deseamos sumar; el programa nos muestra el resultado y se almacena en el portapapeles quedando disponible para pegar en cualquier aplicación.

Menú Inicio

En esta barra vamos a encontrar botones para administrar los archivos de trabajo. Antes de iniciar cualquier proceso de cuantificación debemos de tener un archivo activo, ya sea un existente o nuevo.

Figura 4.3. Menú de Inicio

Nuevo. Creamos un archivo de trabajo; donde al dar clic sobre el botón se abre una ventana en la que debemos de definir una carpeta para almacenar el archivo, además de asignarle un hombre, al darle aceptar en cuestión de segundos hemos creado una base de datos y en la parte superior de la ventana principal aparecerá la ruta completa del archivo creado.

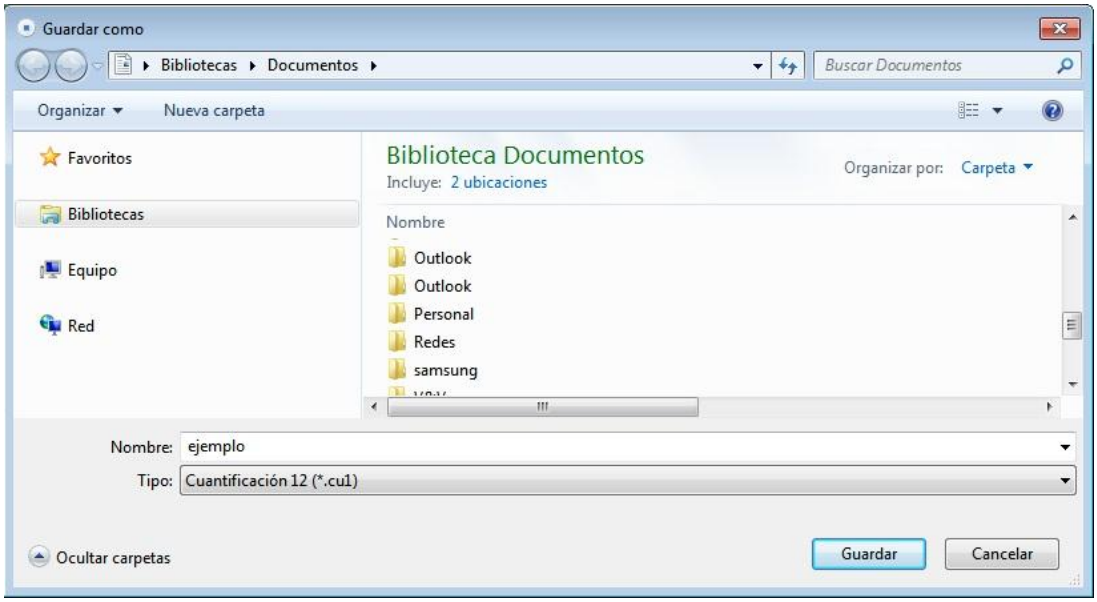


Figura 4.4. Nuevo

Abrir. Si deseamos trabajar en un archivo existente, para completarlo u obtener información; al dar sobre el botón se abre una ventana donde debemos de definir la ruta del archivo y presionamos el botón abrir.

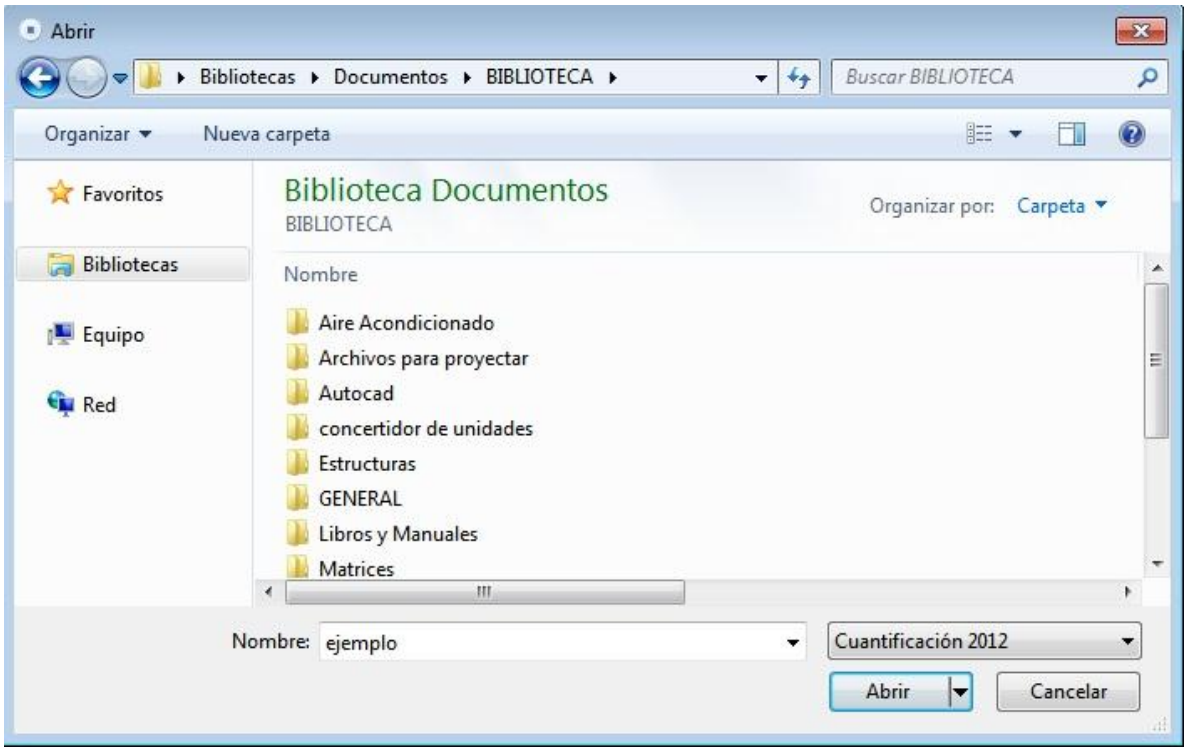


Figura 4.5. Abrir

Guardar. Botón para almacenar todas las modificaciones de la edición manual, cabe mencionar que todos los procesos de cuantificación al final se guardan automáticamente por lo cual no es necesario emplear este botón en estos pasos.

Anterior. Al abrir el programa podemos llamar archivo con el que estábamos trabajando, al colocar el cursor sobre el botón nos aparecerá la ruta completa y en nombre del último archivo que utilizábamos; al dar clic sobre el botón encargará automáticamente el archivo sin tener que buscarlo en las carpetas.

Menú Matrices

En esta barra podemos encontrar botones los cuales abrirán una ventana secundaria donde se administrará más información.

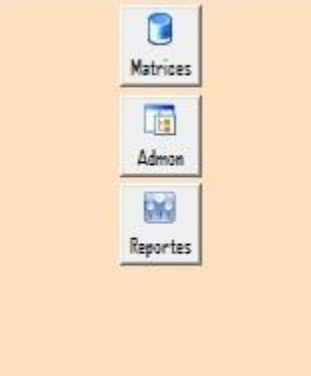


Figura 4.6.

Matrices. Con este botón abrimos la ventana de matrices, donde podemos crear conceptos con diversos componentes tal y como se hace en un precio unitario.

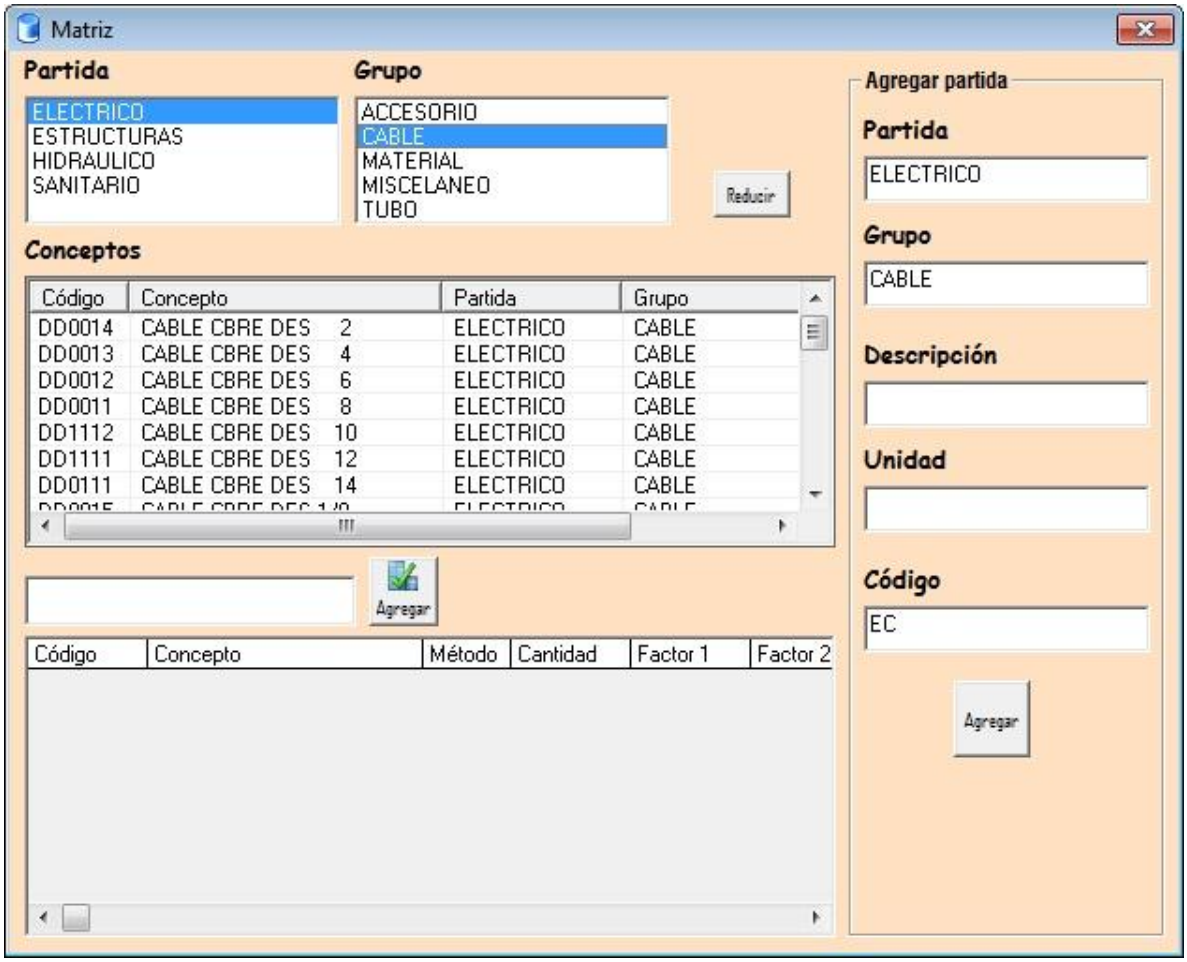


Figura 4.7. Ventana de administración de Matrices

Partida. Lista de principales actividades a donde pertenece el concepto. Para agregar una nueva partida sobre el cuadro de lista debemos de dar clic con el botón derecho del mouse, nos aparecerá una ventana preguntando por el nombre de la nueva partida; para actualizar los datos debemos cerrar la ventana volver a abrir.

Grupo. Parte o subdivisión de la partida. Para agregar un nuevo grupo sobre el cuadro de lista debemos de dar clic con el botón derecho del mouse con lo cual nos aparecerá una ventana preguntando por el nombre del nuevo grupo; para actualizar la lista de grupo se selecciona nuevamente la partida.

Conceptos. Lista de los insumos que pertenece a la partida y grupo seleccionado. Para agregar un concepto se da clic sobre el botón de ampliar se hace más ancha la ventana nos muestra los campos que debemos llenar para el nuevo concepto. La partida y el grupo tienen el contenido de lo seleccionado en las listas correspondientes, la descripción y la unidad es definida por el usuario, el código se determina automáticamente, para completar el proceso dar clic en Agregar; para actualizar la lista de concepto se selecciona nuevamente el grupo.

Para crear nuestro concepto con formato de matriz, debemos agregar todos los insumos que sean necesarios para poder conformarlo; para agregar un insumo a la tabla que se encuentra en la parte inferior que es el cuerpo de la matriz, se hace doble clic sobre el concepto; para eliminar un concepto de la matriz se le da doble clic al concepto en la columna de Código y confirmar si en realidad se desea eliminar.

En el cuerpo de la tabla de la matriz existen tres columnas que se deben editar. Que conforman la siguiente ecuación:

(Tramo medido + Longitud) * Factor 1 * Factor 2

Donde:

Tramo medido. Es la dimensión, longitud, área y cantidad; obtenida directamente de los objetos cuantificados en el plano. El programa calculan las tres dimensiones que se tienen, en el formato de reportes el usuario deberá seleccionar el necesario.

Cantidad. Es un valor definido por el usuario el cual se va a agregar al concepto indistintamente de su valor obtenido de la cuantificación del plano. Para editar se da doble clic sobre el valor.

Factor 1. Es un valor definido por el usuario, donde podremos representar la cantidad de elementos contenidos en el tramo medioPara editar se da doble clic sobre el valor.

Factor 2. Es un valor definido por el usuario, donde podremos representar el desperdicio. Para editar se da doble clic sobre el valor.

Para la fórmula, si hablamos en el terreno eléctrico podemos definir una cédula de cable la cual contendrá tres tramos de cable y que en nuestro criterio por cada tramo vamos a dejar 2m de "coca" para los trabajos de conexiones indistintamente además vamos a considerar un desperdicio del 10%; nuestra tabla de matriz quedan de la siguiente manera como se muestra en la tabla 4.1.

CED01	Código	Concepto	Método	Factor
1	DC2036	CABLE THW 8	Longitud	(Longitud + 2) x 3 x 1.1

Tabla 4.1. Ejemplo de Matriz de concepto eléctrico

A continuación se muestra la abla 4.2. de los resultados. La columna de longitud es el tramo medio del plano y la columna lxf es el resultado de la multiplicación por el factor.

ID	No.	Concepto	Clave	Material		Longitud (l)	Factor	l x f
1	26	CED01	DC2036	CABLE THW	8	12	(Longitud+2)x3x1.1	46.2
2	27	CED01	DC2036	CABLE THW	8	8	(Longitud+2)x3x1.1	33

Tabla 4.2. Tabla de Resultados de conepto eléctrico

En el área de estructuras, un concepto usual es una sección, por ejemplo, una trabe de 20x40 cm que contienen seis varillas del número 4 estribos del número a cada 20 cm. Nuestra matriz queda de la siguiente manera.

T-3	Código	Concepto	Método	Factor
1	CFC200	CONCRETO FC 200	Longitud	(Longitud + 0) x .08 x 1.05
2	VARNO4	VARILLA NO. 4	Longitud	(Longitud + 0) x 6 x 1.03
3	VARNO2	VARILLA NO. 2	Longitud	(Longitud + 0.20) x 0.20 x 1.03

Tabla 4.3. Ejemplo de matriz de concepto de estructuras

A continuación se muestra la tabla 4.4. de resultados.

ID	No.	Concepto	Clave	Material	Longitud (l)	Factor	l x f
1	30	T-3	CFC200	CONCRETO FC 200	6.3	(Longitud + 0) x .08 x 1.05	0.529
1	30	T-3	VARNO4	VARILLA NO. 4	6.3	(Longitud + 0) x 6 x 1.03	38.934
1	30	T-3	VARNO2	VARILLA NO. 2	6.3	(Longitud + 0.20) x 5 x 1.03	33.475

Tabla 4.4. Tabla de Resultados de concepto de estructuras

Se describe cada factor.

Para el concepto CONCRETO FC 200, (Longitud + 0) x .08 x 1.05; el valor **.08**, necesitamos obtener el volumen del material por lo cual debemos multiplicar las dimensiones 0.20m de

base por 0.40m de altura por 1m de largo, (0.2x0.4x1) lo que resulta 0.08, esto quiere decir que por cada metro medido en el plano, tenemos 0.08m3 de concreto; el valor **1.05** es el desperdicio representado en decimales, lo que significa que el volumen por cada metro obtenido ahora será **0.084**.

Para VARILLA NO. 4, (Longitud + 0) x 6 x 1.03; **6** es la cantidad de varillas de la sección y el **1.03** es el desperdicio para acero, aunque falta el porcentaje de dobleces y traslapes este no lo considero en el ejemplo; entonces por cada metro obtenido del plano, se obtiene 6.18m.

Para VARILLA NO. 2, (Longitud + 0.20) x 5 x 1.03; para obtener de manera tradicional la cantidad de estribos la fórmula es (L/ sep estribos)+1; **0.20** es un tramo adicional en relación a la separación de los estribos para suplir el "+1" de la fórmula tradicional y no falte un estribo por cada tramo; 5 es resultado de la relación 1/0.20, lo cual significa que por cada metro tenemos 5 estribos; 1.03 es el desperdicio asignado al concepto. En ejemplo con la fórmula tradicional tenemos ((6.3/0.2)+1)*1.03, lo que resulta 33.475, comparando con la fórmula aplicada en la matriz es igual.

Para concluir asignamos un nombre a la matriz y damos clic en Agregar, con esto se ha agregado en la lista de conceptos localizado en la ventana principal en el menú de Conceptos. Sin cerrar la ventana podemos agregar los conceptos con matriz que necesitemos.

Admon. Con este botón abre la ventana Administrador, ver figura 4.8., donde se administra los conceptos tipo matriz.

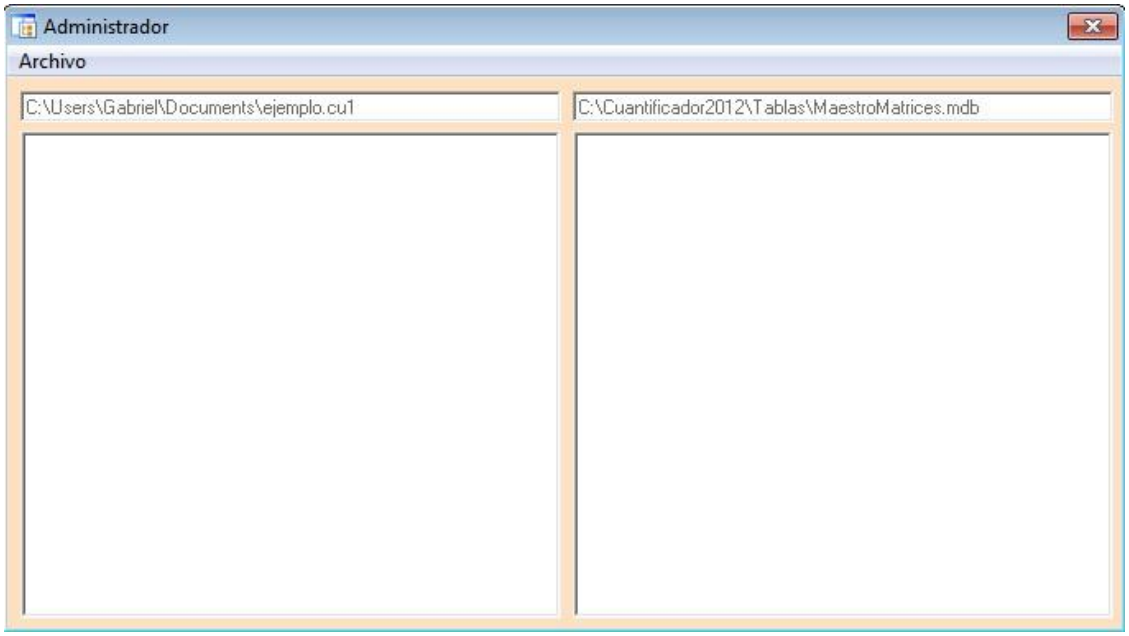


Figura 4.8. Ventana de administración de conceptos

Del lado izquierdo se muestra el archivo actual y del lado derecho el archivo de referencia donde podemos almacenar u obtener matrices; la ruta de ambos archivos se muestra en la parte superior. Podemos cambiar el archivo de referencia dando clic en **Abrir archivo base**en el **Menú Archivo**, donde aparecerá una ventana para localizar el archivo; de forma predeterminada el archivo es **MaestroMatrices.**

Ahora con sólo seleccionar y arrastrar llevamos de una ventana a otra la información; de un archivo a otro. En la ventana aparecen los nombres de los conceptos y para poder conocer la información que contiene, ver figura 4.9., seleccionamos el concepto y mantenemos presionado sobre el icono el botón derecho del mouse. Así se sabe si es el concepto que necesitamos.

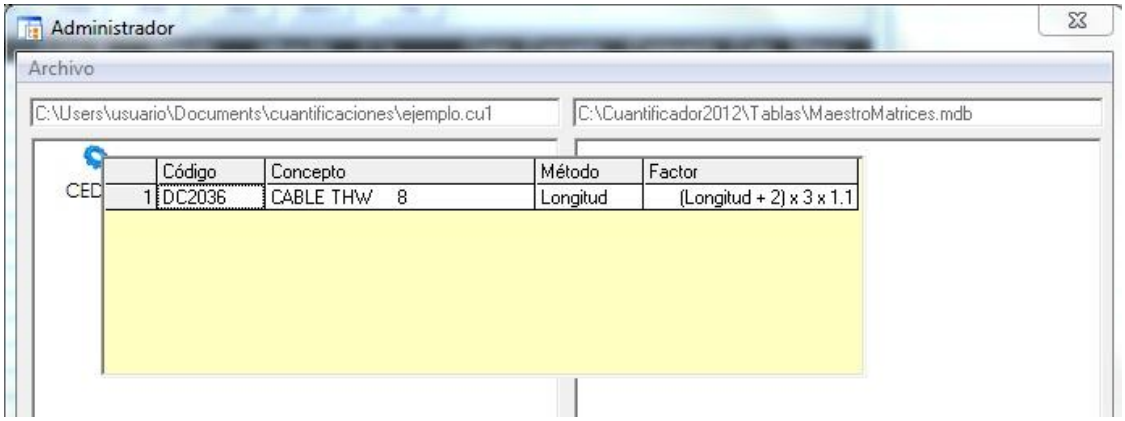


Figura 4.9. Contenido de la matriz

Reportes. Con este botón abrimos la ventana reportes, como se muestra en la figura 4.10. Y administramos la información obtenida en el plano.

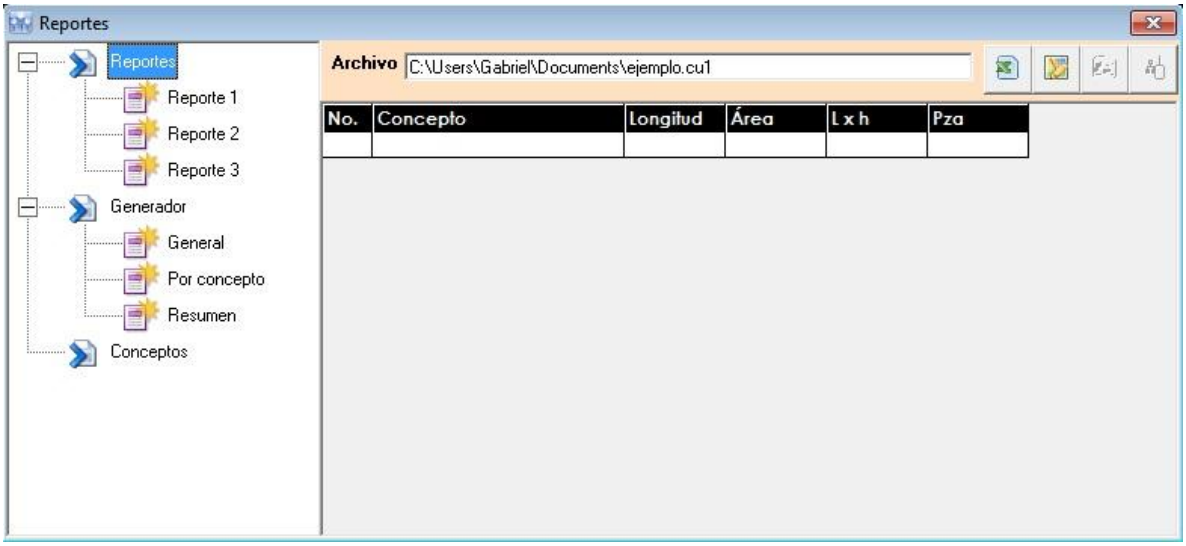


Figura 4.10. Ventana de Reportes

La ventana **Reportes** en la parte superior muestra la ruta del archivo de trabajo; una tabla donde muestra toda la información generada; cuatro botones en la parte superior derecha, donde escribimos la información de la tabla en Excel, en Autocad, en Excel con formato personalizado y un botón para exportar la imagen de Autocad a Excel; del lado izquierdo contiene una lista para generar la información en la tabla; Reportes, se calculan los valores obtenidos de la cuantificación combinados con los conceptos en formato de Matriz, es decir, se generará la información de los insumos; Generador, se muestra la información de todos los elementos medidos en el plano, de manera general, por concepto, o en resumen; conceptos, se enlistan los conceptos que han sido creados con una matriz. Para crear la información en la tabla es la doble clic en el icono.

Se recomienda conocer y hacer ejercicio de la tabla de resultados, ya que la tabla es completa y muestra todos los valores longitud, área, área vertical, piezas, además de las multiplicaciones por su factor; y dependiendo del tipo del concepto de cuantificación es el valor que debemos tomar. Por ejemplo cuando medimos el área para un piso, en una columna nos muestra área y el otro no muestra la longitud que podemos considerarla como el zoclo.

Menú Conceptos

Es el menú para trabajar con conceptos que debemos asignar a la hora de cuantificar, figura 4.11.; en la parte superior aparece una lista donde se muestra todos los conceptos que hemos agregado de manera simple o como matriz, si se encuentra la lista vacía aparecerá Concepto. El botón agregar, sirve para crear un concepto de forma simple; el botón eliminar borra el concepto seleccionado en la lista; botón borrar todo, elimina todos los conceptos contenidos en la lista; en la parte inferior aparece un comando que cuando está habilitado en la tabla principal sólo se muestra el concepto que muestra la lista de este menú.

Figura 4.11. Menú conceptos

Menú Propiedades

Este menú está dividido en dos grupos, como se muestra en la figura 4.12 la marca y el color; Marca, existen diferentes tipos de figuras para utilizar el momento de cuantificar en el plano, también contiene una casilla para determinar el tamaño de esta marca figura; color, existe una lista de botones para definir el que deseamos utilizar en nuestro plano.

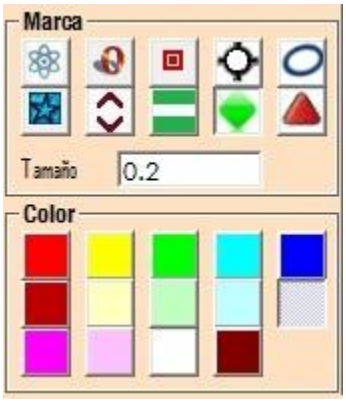


Figura 4.12 Menú propiedades

Menú Cuantificar

Contiene seis botones y dos parámetros para cuantificar de forma rápida; como se ejemplifica en la figura 4.13. Cuando le hemos definido los criterios anteriores el concepto, la marca y el color, ahora podemos utilizar este menú.

Figura 4.13. Menú principal

Descripción de botones

General, al dar clic en el botón se oculta la ventana principal se dirige el control hacia Autocad, donde podemos seleccionar por objeto o por ventana; por objeto tenemos que ir eligiendo 1 × 1 de los elementos a cuantificar, por ventana, se elige el área donde todos son elementos contenidos en esta serán los elementos a cuantificar. Al terminar de seleccionar los objetos debemos dar clic al botón derecho del mouse para completar el proceso; aparecerá una barra donde nos indica el porcentaje de avance y cuando ésta termina regresamos a nuestra ventana principal.

PLínea, este botón utilizamos cuando necesitamos saber distancias o áreas que existen pero no están dibujadas en el plano; al dar clic en el botón se oculta la ventana principal se dirige el control hacia Autocad, se indica con el mouse, con clic en el botón izquierdo, todos los puntos que conforman la dimensión que deseamos obtener; cuando hemos marcado todos los puntos terminamos el comando con el botón derecho del mouse, regresamos a nuestra ventana principal.

Área, este comando lo utilizamos cuando necesitamos definir en planta una área en forma rectangular, al dar clic sobre el botón la ventana principal se oculta y de control se dirige a Autocad; para este comando sólo se necesitan indicar dos puntos, se indica la esquina del área y su contra esquina.

Block, este comando se utiliza cuando se necesita cuantificar, principalmente piezas que están dibujadas en el plano como bloque; al dar clic en el botón se oculta la ventana principal y nos muestra una ventana de instrucciones para recordar cómo hacer el procedimiento; al cerrar dicho mensaje se dirige el control hacia Autocad, este comando es en dos pasos, el primero es seleccionar la lista de bloques que deseamos cuantificar, por ejemplo seleccionarlos en la simbología del plano, y damos Enter con el botón derecho del mouse; el segundo paso es seleccionar el área general donde queremos cuantificar, damos Enter con el botón derecho del mouse, aparecerá una barra donde nos indica el porcentaje de avance y cuando ésta termina regresamos a nuestra ventana principal.

Block's, este comando se utiliza cuando se necesita cuantificar todas las piezas dibujadas en bloque del plano; al dar clic en el botón se oculta la ventana principal y se dirige el control hacia Autocad, se selecciona toda el área que deseamos obtener y le damos Enter con el botón derecho del mouse, aparecerá una barra donde nos indica el porcentaje de avance y cuando ésta termina regresamos a nuestra ventana principal.

Pza, este comando se utiliza para cuantificar piezas, al dar clic en el botón se oculta la ventana principal y se dirige el control hacia Autocad, con el botón derecho del mouse se indica todos los objetos que deseamos cuantificar, se recomienda para este paso esperar hasta que la barra de comandos nos indique el siguiente punto, para terminar este comando damos clic en el botón derecho del mouse.

Altura, el parámetro nos apoya para obtener área vertical, al multiplicarlo por la longitud obtenida en planta de los objetos cuantificados; se modifica con tan sólo entrar a la etiqueta y modificar la nueva dimensión.

Elemento, este parámetro nos indica el número de los objetos cuantificados y asigna el número consecutivo a cada elemento; se puede modificar si es necesario iniciar la numeración en algún número en especial.

Menú Herramientas

Contiene dos parámetros para mostrar los datos en la tabla y la representación en el plano.

Figura 4.14. Menú Herramientas

Precisión, es una lista de datos donde podemos elegir el parámetro para redondear nuestros valores en las operaciones, como por ejemplo al multiplicar la longitud por la altura, al tener nuestra precisión en 0.01, el redondeo se realizará a dos décimas.

Grosor, la cuantificación de polilíneas además de indicar una figura, color y número, sólo para estos objetos se hará la representación gráfica con el ancho de la línea.

Menú Marca

Contiene cinco parámetros, los cuales cuanto son habilitados se mostrarán en los elementos cuantificados del plano, según figura 4.15.

Figura 4.15. Menú marca

La representación gráfica del plano es como muestra la figura 4.16. Los parámetros son representados de arriba hacia abajo tal y como parece de izquierda a derecha.

Figura 4.16. Representación gráfica de elementos cuantificados

4.2. Sub rutinas referidas a los objetos del programa.

Ahora para cada ventana del programa, se agrega su rutina de programación, para que funcione cada uno de los componentes establecidos y diseñados. Con una breve explicación al principio de la rutina, además que en el cuerpo.

Ventana principal

A continuación se presentan las rutinas para la ventana principal.

Rutina que se efectúa cuando se carga la ventana, abre el archivo "opciones.mdb" de donde obtiene los últimos parámetros utilizados y configura la ventana principal.

```
Private Sub Form_Load

' *** lectura de opciones

    Dim db As Database
    Dim Fd As Field
    Dim Tb As New TableDef
    Dim i As Integer

sbase = LCase("C:\Cuantificador2012\System\Opciones.mdb")
Set db = OpenDatabase(sbase)

    Set Rs = db.OpenRecordset("Opciones", dbOpenDynaset)
        MarcaX(Rs.Fields("Marca")).Value = True: Ma = Rs.Fields("marca")
        colorX(Rs.Fields("color")).Value = True: co = Rs.Fields("color")
TamanoMarca = Rs.Fields("tamaño")
        Altura = Rs.Fields("Altura")
Presicion.ListIndex = Rs.Fields("precision")
        Etiqueta = Rs.Fields("etiqueta")

    Set Rs = db.OpenRecordset("Impresion", dbOpenDynaset)
        ImpresionMarca(0).Value = Rs.Fields("no")
        ImpresionMarca(1).Value = Rs.Fields("longitud")
        ImpresionMarca(2).Value = Rs.Fields("area")
        ImpresionMarca(3).Value = Rs.Fields("lxh")
        ImpresionMarca(4).Value = Rs.Fields("marca")

    Set Rs = db.OpenRecordset("Varios", dbOpenDynaset)
        ArchAnt = Rs.Fields("anterior")
        Boton(4).ToolTipText = ArchAnt
db.Close

' Cargar el menú de Acero
Perfil.AddItem "CE"
    Perfil.AddItem "IE"
    Perfil.AddItem "IR"
    Perfil.AddItem "IS"
    Perfil.AddItem "LD"
    Perfil.AddItem "LI"
    Perfil.AddItem "OC"
    Perfil.AddItem "OR"
    Perfil.AddItem "OS"
    Perfil.AddItem "TR"
Perfil.ListIndex = 0
    CargarPerfiles ("CE")
    cte2 = CargarPESO("CE", "76X6.10")
    MenuPrincipal.Tab = 0

End Sub
```

Rutina que se ejecutan al dar doble clic sobre la tabla, donde copia los valores de la celda seleccionada al portapapeles.

```
Private Sub tabla_DblClick
If tabla.Rows = 1 Then Exit Sub
X = tabla.MouseCol
Y = tabla.MouseRow

' Copia valores de la tabla

If X = 1 Or X = 2 Or X = 4 Then
Clipboard.Clear
Clipboard.SetText tabla.TextMatrix(Y, X)
End If

' Cambia de signo los valores de la tabla
If X = 5 Then

If tabla.TextMatrix(Y, 5) = "+" Then tabla.TextMatrix(Y, 5) = "-":
tabla2.TextMatrix(Y, 5) = "-": GoTo 20
If tabla.TextMatrix(Y, 5) = "-" Then tabla.TextMatrix(Y, 5) = "+":
tabla2.TextMatrix(Y, 5) = "+"
20

For h = 1 To 4
    tabla.TextMatrix(Y, h) = tabla.TextMatrix(Y, h) * -1
    tabla2.TextMatrix(Y, h) = tabla2.TextMatrix(Y, h) * -1
Next

End If

' Ajustar los valores seleccionados en la tabla
Dim pres As Variant
pres = Tabla1.Presicion.ListIndex

If X = 3 Then
cte02 = InputBox("Nuevo Valor", "Valor Anterior " & tabla.TextMatrix(Y, X))
If cte02 = "" Then GoTo 50
tabla.TextMatrix(Y, X) = cte02
tabla.TextMatrix(Y, 4) = Round(tabla.TextMatrix(Y, 1) * tabla.TextMatrix(Y, 3),
pres)
tabla2.TextMatrix(Y, X) = cte02
tabla2.TextMatrix(Y, 4) = tabla.TextMatrix(Y, 4)
50
End If

End Sub
```

Rutina que ejecuta con el botón derecho del mouse el menú contextua. Abre el menú contextual donde permite hacer operaciones con los renglones seleccionados.

```
Private Sub tabla_MouseDown
If Tabla1.tabla.Rows <= 1 Then GoTo 100
If Button = 2 Then
Unload Menu01: Menu01.Show
End If
100
End Sub
```

Rutina de control del menú del lado izquierdo, controlando la apariencia y despliegue de botones principales, que se ejecutan cada que se da clic sobre los botones principales.

```
Private Sub MenuBotones_Click

Boton(0).Visible = False
Boton(1).Visible = False
Boton(2).Visible = False
Boton(3).Visible = False
Boton(4).Visible = False
Boton(5).Visible = False
Boton(6).Visible = False
Boton(7).Visible = False
Boton(8).Visible = False
Boton(9).Visible = False
Conceptos.Visible = False
Visualizar.Visible = False
Frame2.Visible = False
Frame3.Visible = False


Select Case Index

    Case Is = 1
        MenuBotones(1).Top = 0
        Boton(1).Visible = True: Boton(1).Left = 960: Boton(1).Top = 545
        Boton(2).Visible = True: Boton(2).Left = 960: Boton(2).Top = 1210
        Boton(3).Visible = True: Boton(3).Left = 960: Boton(3).Top = 1875
        Boton(4).Visible = True: Boton(4).Left = 960: Boton(4).Top = 2540
        MenuBotones(2).Top = 3455
        MenuBotones(3).Top = 3950
        MenuBotones(4).Top = 4445
    Case Is = 2
        MenuBotones(1).Top = 0
        MenuBotones(2).Top = 495
        Boton(5).Visible = True: Boton(5).Left = 960: Boton(5).Top = 1040
        Boton(0).Visible = True: Boton(0).Left = 960: Boton(0).Top = 1705
        Boton(6).Visible = True: Boton(6).Left = 960: Boton(6).Top = 2370
        MenuBotones(3).Top = 3950
        MenuBotones(4).Top = 4445

    Case Is = 3
        MenuBotones(1).Top = 0
        MenuBotones(2).Top = 495
        MenuBotones(3).Top = 990
        Conceptos.Visible = True: Conceptos.Left = 0: Conceptos.Top = 1535
        Boton(7).Visible = True: Boton(7).Left = 960: Boton(7).Top = 1960
        Boton(8).Visible = True: Boton(8).Left = 960: Boton(8).Top = 2625
        Boton(9).Visible = True: Boton(9).Left = 960: Boton(9).Top = 3290
Visualizar.Visible = True: Visualizar.Left = 500: Visualizar.Top = 4000
MenuBotones(4).Top = 4445

    Case Is = 4
        MenuBotones(1).Top = 0
MenuBotones(2).Top = 495
```

```
        MenuBotones(3).Top = 990
        MenuBotones(4).Top = 1485
        Frame2.Visible = True: Frame2.Left = 10: Frame2.Top = 1990
        Frame3.Visible = True: Frame3.Left = 10: Frame3.Top = 3440
End Select

End Sub
```

Menú Inicio

Se describen a continuación los botones contenidos en este menú

Rutina que se ejecutan con los botones del menú, reconocido con el Index del botón, ya que todos los botones del menú están contenidos en una matriz de botones.

```
Private Sub Boton_Click

Dim Qo1 As Variant
Dim s As Variant

' botón Admon
If Index = 0 Then

        If Dir("C:\Cuantificador2012\Tablas\MaestroMatrices.mdb", vbArchive) = ""
Then CrearMaestroMatrices

Administrador.ListView1.ListItems.Clear
        Administrador.ListView2.ListItems.Clear

        Dim db As Database
        Dim Fd As Field
        Dim Tb As New TableDef
        Dim i As Integer
        Dim SubElemento As ListItem

        sbase = LCase(txtnombreArch)
        Set db = OpenDatabase(sbase)

        Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM COMPUESTO
")
                Do Until Rs.EOF
                    r = r + 1
                    Administrador.ListView1.ListItems.Add , , Rs("clave"), 1
Rs.MoveNext
                Loop
        db.Close

        sbase = LCase("C:\Cuantificador2012\Tablas\MaestroMatrices.mdb")
Set db = OpenDatabase(sbase)
        Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM COMPUESTO
")
                Do Until Rs.EOF
                    r = r + 1
                    Administrador.ListView2.ListItems.Add , , Rs("clave"), 2
```

```
                    Rs.MoveNext
                Loop
        db.Close
    Administrador.ruta1 = txtnombreArch
    Administrador.Show
End If


' Nuevo
If Index = 1 Then

    On Error Resume Next
        With CommonDialog1
.CancelError = True
            .FileName = Tabla1.txtnombreArch
.Filter = "Cuantificación 12 (*.cu1)|*.cu1"
.FilterIndex = 1
.Flags = cdlOFNFileMustExist + cdlOFNHideReadOnly
            .ShowSave

            If Err = 0 Then
s = .FileTitle

                If Dir(.FileName) <> "" Then Qo1 = MsgBox("Existe un archivo " &
_
"con el mismo nombre" & vbCr & "¿Desea Continuar?", vbYesNo + vbInformation)
If Qo1 = vbNo Then Exit Sub

                Screen.MousePointer = vbHourglass: Tabla1.tabla.MousePointer =
flexHourglass
                If Dir(.FileName) <> "" Then Kill (.FileName)
                CrearTabla (.FileName)
                ArchAnt = .FileName
                txtnombreArch = .FileName
                Boton(4).ToolTipText = .FileName
                tabla.Rows = 1: tabla2.Rows = 1: tablaC.Rows = 1
EncabezadoDeTabla
                Conceptos.Clear: Conceptos.Text = "Concepto"
Screen.MousePointer = vbDefault: Tabla1.tabla.MousePointer = flexDefault
Tabla1.Etiqueta = 0
                Tabla1.Conceptos = "Concepto"
Tabla1.Caption = "Cacho - 2012 || " & txtnombreArch

                MenuBotones(2).Enabled = True
                MenuBotones(3).Enabled = True
                MenuBotones(4).Enabled = True
            End If
        End With
End If


' Abrir
If Index = 2 Then

    On Error Resume Next
        With CommonDialog1
            .CancelError = True
```

```
            .FileName = Tabla1.txtnombreArch
            .Filter = "Cuantificación 2012|*.cu1"
            .FilterIndex = 1
            .Flags = cdlOFNFileMustExist + cdlOFNHideReadOnly
            .ShowOpen

            If Err = 0 Then
                Tabla1.txtnombreArch = .FileName
                Tabla1.ArchAnt = .FileName
                s = .FileTitle
                Boton(4).ToolTipText = .FileName
            End If

            Screen.MousePointer   =   vbHourglass:   Tabla1.tabla.MousePointer   =
flexHourglass
            If .FileName <> "" Then
                tabla.Rows = 1: tabla2.Rows = 1: tablaC.Rows = 1
Conceptos.Clear
                CargarTabla (.FileName)
                EncabezadoDeTabla
                Tabla1.Caption = "Cacho - 2012 || " & txtnombreArch
MenuBotones(2).Enabled = True
                MenuBotones(3).Enabled = True
                MenuBotones(4).Enabled = True
            End If
            Screen.MousePointer   =   vbDefault:   Tabla1.tabla.MousePointer   =
flexDefault
        End With
End If


' Guardar
If Index = 3 Then

If txtnombreArch = "" Or Dir(txtnombreArch) = "" Then MsgBox "No hay archivo
abierto": GoTo 70

GuardarTabla (txtnombreArch)
        GuardarOpciones
70
End If


' Anterior
If Index = 4 Then

sbase = LCase("C:\Cuantificador2012\System\Opciones.mdb")
Set db = OpenDatabase(sbase)
    Set Rs = db.OpenRecordset("Varios", dbOpenDynaset)
        ArchAnt = Rs.Fields("anterior")
    db.Close

If Dir(ArchAnt, vbArchive) = "" Then MsgBox "No se encuentra el archivo":    Exit
Sub
txtnombreArch = ArchAnt

    Boton(4).ToolTipText = ArchAnt
```

```
    tabla.Rows = 1: tabla2.Rows = 1: tablaC.Rows = 1
Conceptos.Clear: Conceptos.Text = "Concepto"
    CargarTabla (ArchAnt): EncabezadoDeTabla
    Tabla1.Caption = "Cacho - 2012 || " & txtnombreArch
MenuBotones(2).Enabled = True
        MenuBotones(3).Enabled = True
        MenuBotones(4).Enabled = True
End If

' Matrices
If Index = 5 Then

    Unload Matrices: Matrices.Show

End If


' Reporteador
If Index = 6 Then
            On Error GoTo 150

            On Error Resume Next
            Set AApp = GetObject(, "AutoCAD.Application")
If Err Then
                MsgBox Err.Description, vbCritical, "GRICC || Autocad"
                GoTo 150
            End If

            Dim xlApp As Excel.Application
            Dim xlWB As Excel.Workbook
            Dim xlWS As Excel.Worksheet

            On Error Resume Next
            Set xlApp = GetObject(, "Excel.Application")
If Err Then
                MsgBox Err.Description, vbCritical, "GRICC || Excel"
                GoTo 150
            End If

            AppActivate "Autocad"
AppActivate "Excel"
            AppActivate "Cacho"

            Unload     Reporteador:     Reporteador.ruta     =     txtnombreArch:
Reporteador.Show: Exit Sub
150       MsgBox "Verificar que este abierto Excel y Autocad": Exit Sub
End If


' Botón para agregar conceptos
If Index = 7 Then
X = UCase(SinEspacios(InputBox("Concepto a agregar", "GRICC")))
If Trim(X) = "" Then GoTo 30

        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
            If Conceptos.Text = X Then
```

```
MsgBox "Ya existe " & X, vbInformation
                Conceptos = X
GoTo 30
            End If
        Next

    Conceptos.AddItem X
r = Conceptos.ListCount
Conceptos.ListIndex = r - 1
    LCompuesto = 0
30
FiltrarTabla (txtnombreArch)
End If


' Botón para eliminar conceptos
If Index = 8 Then
    n = Conceptos.ListIndex
    If Conceptos.ListCount <= 0 Then GoTo 50
    Conceptos.RemoveItem n
    If Conceptos.ListCount <= 0 Then Conceptos.Text = "Concepto" Else
Conceptos.ListIndex = 0
50
End If


' Botón para borrar lista
If Index = 9 Then
Conceptos.Clear: Conceptos.Text = "Concepto"
End If

End Sub
```

Menú Cuantificar

Este menú contiene los botones los más importantes de todo el programa ya que obtienen la información directamente del plano de Autocad y registrar dichos valores en las la ventana principal.

```
Private Sub General_Click

If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub
Dim m(0 To 2) As Double
Dim n(0 To 5) As Double

Tabla1.Hide

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
```

```
Set AM = AD.ModelSpace

AppActivate "Autocad"
X = Seleccion()

Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

ht = TamanoMarca * 0.8
ATP = 0
i = 0
radio_min = TamanoMarca

vc = 0
progreso.Show
AppActivate "Cacho"
progreso.ProgressBar1.Min = 0
progreso.ProgressBar1.Max = g1.Count

‘ evalúa todos los objetos contenidos en la selección
For Each entry In g1

    vc = vc + 1
    progreso.ProgressBar1 = vc
    DoEvents

    If entry.Layer = "Cuant_NUM" Then GoTo 90
    If entry.ObjectName = "AcDbText" Then GoTo 90
    If entry.ObjectName = "AcDbMText" Then GoTo 90
    If entry.ObjectName = "AcDbEllipse" Then GoTo 90

    ' -- Hatch
    If entry.ObjectName = "AcDbHatch" Then
        A = entry.Area
        varA = entry.basePoint
        entry.GetBoundingBox minExt, maxExt
        m(0) = (minExt(0) + maxExt(0)) / 2
        m(1) = (minExt(1) + maxExt(1)) / 2
        GoTo 60
    End If

    ' -- Arco
    If entry.ObjectName = "AcDbArc" Then
        P = entry.ArcLength
        A = entry.Area
varA = entry.center

        'MsgBox entry.Radius
Pi = 3.14159265
        Sangle = entry.StartAngle * (180 / Pi)
        tangle = entry.TotalAngle * (180 / Pi)
        Mangle = Sangle + (tangle / 2)
cteY = entry.radius * Sin(Mangle * (Pi / 180))
        cteX = entry.radius * Cos(Mangle * (Pi / 180))

        m(0) = varA(0) + cteX: m(1) = varA(1) + cteY: m(2) = 0
        'm(0) = varA(0): m(1) = varA(1): m(2) = 0
```

```
GoTo 60
    End If

    ' -- Block
    If entry.ObjectName = "AcDbBlockReference" Then
bloque = SinEspacios(UCase(entry.Name))
varA = entry.insertionPoint
        m(0) = varA(0): m(1) = varA(1): m(2) = 0
        A = 0: P = 0
        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
            If bloque = Conceptos.Text Then GoTo 55
        Next
        Conceptos.AddItem bloque
55
        GoTo 60
    End If

    ' -- Círculo
    If entry.ObjectName = "AcDbCircle" Then
        P = 3.141592654 * ((entry.radius * 2))
        A = entry.Area
        entry.GetBoundingBox minExt, maxExt
        m(0) = (minExt(0) + maxExt(0)) / 2
        m(1) = (minExt(1) + maxExt(1)) / 2
        GoTo 60
    End If

    ' -- Polilínea
    If entry.ObjectName = "AcDbPolyline" Or entry.ObjectName = "AcDb2dPolyline"
Then
        P = entry.Length
        A = entry.Area

        vv = entry.Coordinates

            Dim e(0 To 3) As Double

        m(0) = (vv(2) + vv(0)) / 2
        m(1) = (vv(3) + vv(1)) / 2

        GoTo 60
    End If

    ' -- Línea
    If entry.ObjectName = "AcDbLine" Then
        A = 0
        P = entry.Length
        startPoint = entry.startPoint
        endPoint = entry.endPoint
        n(0) = startPoint(0): n(1) = startPoint(1): n(2) = startPoint(2)
        n(3) = endPoint(0): n(4) = endPoint(1): n(5) = endPoint(2)
        m(0) = (n(0) + n(3)) / 2
        m(1) = (n(1) + n(4)) / 2
        m(2) = 0
        GoTo 60
    End If
```

```
60
    co = SeleccionDeColor(colorX)
    entry.Color = co
    Etiqueta = Etiqueta + 1

'asigna propiedades
    entry.Highlight (False)
If borde.Value = True Then entry.Color = co: entry.ConstantWidth = Grosor
If Conceptos = "" Then Conceptos = "Ejemplo"

'captura valores en tabla
    pres = Presicion.ListIndex
r = tabla.Rows
tabla.Rows = r + 1

    If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta
    tabla.TextMatrix(r, 1) = Round(P, pres)
    tabla.TextMatrix(r, 2) = Round(A, pres)
    tabla.TextMatrix(r, 3) = Round(Altura, pres)
    tabla.TextMatrix(r, 4) = Round(P * Altura, pres)
    tabla.TextMatrix(r, 5) = "+"
    tabla.TextMatrix(r, 6) = Conceptos
    tabla.TextMatrix(r, 7) = LCompuesto
tabla.TextMatrix(r, 8) = co
    tabla.TextMatrix(r, 9) = SeleccionDeMarca
tabla.TextMatrix(r, 10) = Mid(entry.ObjectName, 5, Len(entry.ObjectName))
    tabla.TextMatrix(r, 11) = Trim(UCase(entry.Handle))

'dibuja la marca

'borde
If borde.Value = True Then
Set El = MS.AddPolyline(n)
El.Color = co
El.ConstantWidth = LIN
End If

X = dibujar(m(0), m(1), m(2), radio_min, co, Ma)
If tabla.Rows > 1 Then tabla.FixedRows = 1
' termina de dibujar la marca

450
90

Next

If tabla.Rows > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla

Unload progreso
Update

200

End Sub
```

Boton que permite, a partir de indicar "n" puntos con el ratón, determinar área y perímetro.

```
Private Sub Pline_Click

If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub

Dim AApp As Object, l As Object, MS As Object, U As Object
Dim El As Object
Dim m(0 To 2) As Double
Dim varx As Double
Dim bloque As Variant
Dim insertionPoint(0 To 2) As Double, Alignmentpoint(0 To 2) As Double
Dim plineObj As AcadLWPolyline
Dim radio_min As Double
Dim ht As Double
Dim po As Object
Dim X1 As Variant, X2 As Variant, d As Double
Dim px1(0 To 2) As Double
Dim px2(0 To 2) As Double
Dim py1(0 To 2) As Double
Dim py2(0 To 2) As Double

radio_min = TamanoMarca

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility


Tabla1.Hide

On Error Resume Next
X = 0: dt = 0: d = 0

Dim db As Database
Dim Fd As Field
Dim Tb As New TableDef
Dim i As Integer

sbase = LCase("C:\Cuantificador2012\System\polilinea.mdb")
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("Puntos", dbOpenDynaset)
Dim returnPnt As Variant

Do Until Rs.EOF And Rs.BOF
Rs.MoveFirst
Rs.Delete
Rs.Update
Loop

c = 0
tablaP.Rows = 1
Delta = radio_min * 2
```

```
ReDim n(0 To 5) As Double

xc = Seleccion()
Dim ssetObj As AcadSelectionSet
Set ssetObj = AD.SelectionSets.Add("prueba" & xc)

ReDim ssobjs(0 To 100) As AcadEntity

        Do Until X = 2

            returnPnt = l.Utility.GetPoint(, "Punto " & c & " [ESC con botón
secundario]: ")
            X = MouseButton
            r = tablaP.Rows

            If X = 1 Then

                c = c + 1

                tablaP.Rows = r + 1
                tablaP.TextMatrix(r, 0) = returnPnt(0)
                tablaP.TextMatrix(r, 1) = returnPnt(1)
                tablaP.TextMatrix(r, 2) = returnPnt(2)

                n(0) = returnPnt(0) - Delta
                n(1) = returnPnt(1)
                n(2) = 0
                n(3) = returnPnt(0) + Delta
                n(4) = returnPnt(1)
                n(5) = 0
                Set pl = MS.AddPolyline(n)
                pl.Color = co
                pl.Layer = "cuant_mca"
                Set ssobjs(((c - 1) * 2)) = pl

                n(0) = returnPnt(0)
                n(1) = returnPnt(1) - Delta
                n(2) = 0
                n(3) = returnPnt(0)
                n(4) = returnPnt(1) + Delta
                n(5) = 0
                Set pl = MS.AddPolyline(n)
                pl.Color = co
                pl.Layer = "cuant_mca"
                Set ssobjs(((c - 1) * 2) + 1) = pl

                Update

                If Rs.EOF And Rs.BOF Then Rs.AddNew Else Rs.MoveLast: Rs.AddNew
                Rs.Fields("x") = returnPnt(0)
                Rs.Fields("y") = returnPnt(1)
                Rs.Fields("z") = returnPnt(2)
                Rs.Update
            End If
        Loop
```

```
ssetObj.AddItems ssobjs

For i = 0 To ssetObj.Count - 1
     ssetObj.Item(i).Color = 6
Next

ssetObj.Erase

    Rs.MoveFirst
    ReDim n(0 To (c * 3) - 1) As Double

    For i = 0 To c - 1
        n(i * 3) = Rs.Fields("x")
        n((i * 3) + 1) = Rs.Fields("y")
        n((i * 3) + 2) = Rs.Fields("z")
Rs.MoveNext
    Next

Dim minExt As Variant
Dim maxExt As Variant

If c - 1 > 0 Then
    Dim Pl3 As AcadPolyline
        Set Pl3 = MS.AddPolyline(n)
Pl3.ConstantWidth = Grosor
        Pl3.Color = co
        dt = Pl3.Length
at = Pl3.Area
        bloque = Pl3.Handle

       Pl3.GetBoundingBox minExt, maxExt
       m(0) = (minExt(0) + maxExt(0)) / 2
       m(1) = (minExt(1) + maxExt(1)) / 2

co = SeleccionDeColor(colorX)
        entry.Color = co
        Etiqueta = Etiqueta + 1

cteN = Rs.RecordCount
cteM = Round(cteN * 0.5, 0)

If cteN > 2 Then
    cteI = 0
    Rs.MoveFirst
        Do Until Rs.EOF
            m(0) = Rs.Fields("x")
            m(1) = Rs.Fields("y")
cteI = cteI + 1
            If cteI = cteM Then GoTo 85
Rs.MoveNext
        Loop
85
End If

    'asigna propiedades
        entry.Highlight (False)
        If borde.Value = True Then entry.Color = co: entry.ConstantWidth = Grosor
```

```
If Conceptos = "" Then Conceptos = "Concepto"

    'captura valores en tabla
        pres = Presicion.ListIndex

tabla.Rows = tabla.Rows + 1
r = tabla.Rows - 1

        If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta
        tabla.TextMatrix(r, 1) = Round(dt, pres)
        tabla.TextMatrix(r, 2) = Round(at, pres)
tabla.TextMatrix(r, 3) = Round(Altura, pres)
        tabla.TextMatrix(r, 4) = Round(Round(dt, pres) * Altura, pres)
tabla.TextMatrix(r, 5) = "+"
        tabla.TextMatrix(r, 6) = Conceptos
        tabla.TextMatrix(r, 7) = LCompuesto
        tabla.TextMatrix(r, 8) = co
        tabla.TextMatrix(r, 9) = SeleccionDeMarca
        tabla.TextMatrix(r, 10) = "PolyLine"
        tabla.TextMatrix(r, 11) = UCase(bloque)

        'dibuja la marca
        X = dibujar(m(0), m(1), m(2), radio_min, co, Ma)
If tabla.Rows > 1 Then tabla.FixedRows = 1
' termina de dibujar la marca

If tabla.Rows > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla
        Unload progreso
        Update

End If

Update
db.Close

End Sub
```

Botón para obtener área en forma rectangular, solicita y genera un cuadro a partir de dos puntos para obtener área y perímetro.

```
Private Sub Cuadrante_Click
If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub
On Error GoTo 900

Dim empresa As Variant
Dim ptoI1 As Variant
Dim ptoI2 As Variant
Dim ptoi3 As Variant
Dim ptoi4 As Variant
Dim ptoi5 As Variant
Dim ptoi As Variant

Dim AApp As Object
Dim ModelSpace As Object
```

```
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object
Dim points(0 To 9) As Double
'Dim coorx As Double
'Dim coory As Double

Dim DIF As Double
Dim ht As Double
Dim radio_min As Double

radio_min = TamanoMarca

ht = t_marca
DIF = (t_marca * 1.1)

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace

AppActivate "Autocad"
Tabla1.Hide
ptoI1 = l.Utility.GetPoint(, vbCr & "Primer Punto del Área")
ptoI2 = l.Utility.GetPoint(, vbCr & "Segundo Punto del Área")

points(0) = ptoI1(0): points(1) = ptoI1(1)
    points(2) = ptoI1(0): points(3) = ptoI2(1)
    points(4) = ptoI2(0): points(5) = ptoI2(1)
    points(6) = ptoI2(0): points(7) = ptoI1(1)
    points(8) = ptoI1(0): points(9) = ptoI1(1)

Dim dt As Double, at As Double, bloque As Variant
Dim minExt As Variant, maxExt As Variant
Dim n(0 To 5) As Double
Dim m(0 To 2) As Double

Set El = MS.AddLightWeightPolyline(points)
El.ConstantWidth = Grosor
El.Color = co
dt = El.Length
at = El.Area
bloque = El.Handle
El.GetBoundingBox minExt, maxExt
Update

        n(0) = minExt(0): n(1) = minExt(1): n(2) = minExt(2)
        n(3) = maxExt(0): n(4) = maxExt(1): n(5) = maxExt(2)
m(0) = (n(0) + n(3)) / 2
        m(1) = (n(1) + n(4)) / 2
        m(2) = 0
        m(0) = ptoI1(0)
m(1) = ptoI1(1)

Dim r As Variant, pres As Variant, xm As Variant
Dim axa As Variant
Dim entry As AcadEntity
```

```
Etiqueta = Etiqueta + 1
'captura valores en tabla
axa = 0
        If axa = Empty Then axa = 0
        pres = Presicion.ListIndex

        tabla.Rows = tabla.Rows + 1
        r = tabla.Rows - 1

        If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta
        tabla.TextMatrix(r, 1) = Round(dt, pres)
        tabla.TextMatrix(r, 2) = Round(at, pres)
        tabla.TextMatrix(r, 3) = Round(Altura, pres)
        tabla.TextMatrix(r, 4) = Round(Round(dt, pres) * Altura, pres)
tabla.TextMatrix(r, 5) = "+"
        tabla.TextMatrix(r, 6) = Conceptos
        tabla.TextMatrix(r, 7) = LCompuesto
        tabla.TextMatrix(r, 8) = co
        tabla.TextMatrix(r, 9) = SeleccionDeMarca
        tabla.TextMatrix(r, 10) = "ByRectangle"
        tabla.TextMatrix(r, 11) = UCase(bloque)

'dibuja la marca
X = dibujar(m(0), m(1), m(2), radio_min, co, Ma)
' termina de dibujar la marca

GoTo 900

900
Update

If tabla.Rows > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla

AppActivate "Cacho"
GuardarOpciones
GuardarTabla (txtnombreArch)
FiltrarTabla (txtnombreArch)
Tabla1.Show
1000

End Sub
```

Rutina para cuantificar los bloques respecto a una lista inicial, se realizan dos selecciones de datos, la primera, es lista de bloques únicos que serán ubicados en el plano, y la segunda serlección es de todo el plano o área de búsqueda.

```
Private Sub BlockUnitario_Click

If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub

Dim AApp As Object
Dim ModelSpace As Object
Dim El As Object
```

```
Dim MS As Object
Dim l As Object
Dim U As Object

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
    If Err Then
        Err.Clear
        Set AApp = CreateObject("AutoCAD.Application")
        AApp.Visible = True

        If Err Then
            MsgBox Err.Description, vbCritical, "GRICC"
            Exit Sub
        End If
    End If

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

Dim insertionPoint(0 To 2) As Double
Dim Alignmentpoint(0 To 2) As Double
Dim plineObj As AcadLWPolyline
Dim dib As Variant
Dim ht As Double
Dim entry As AcadEntity
Dim nom1 As Variant, nom2 As Variant
Dim Contador As Double
Dim pza As Double
Dim acdbAcadBlockReference As String
Dim varA As Variant
Dim m(0 To 2) As Double
Dim radio_min As Double
Dim circ As AcadCircle
Dim points(0 To 9) As Double
Dim textobj As AcadText
Dim r As Variant, pres As Variant, axa As Variant, xm As Variant
Dim vc As Double

Tabla1.Hide
ht = TamanoMarca * 0.8
radio_min = TamanoMarca

X = Seleccion()

MsgBox "Seleccionar Los Block's Muestra >> Enter >> Todo el Plano",
vbInformation
AppActivate "Autocad"

Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

' * * * * * Modif001
```

```
ReDim Lista1(0 To 100) As Variant
Contador = 0
For Each entry In g1
    'If Contador <> 0 Then GoTo 120

    If entry.ObjectName <> "AcDbBlockReference" Then GoTo 30
nom1 = UCase(entry.Name)
    Contador = Contador + 1

    Lista1(Contador - 1) = nom1

    ' agrega concepto
X = nom1
    If Trim(X) = "" Then GoTo 30

        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
            If Conceptos.Text = X Then
                Conceptos = X
                GoTo 30
            End If
        Next

    Conceptos.AddItem X
    r = Conceptos.ListCount
Conceptos.ListIndex = r - 1
30
Next

If Contador = 0 Then MsgBox "La lista no contiene Bloques", vbInformation,
"GRICC": GoTo 500

120

X = Seleccion()

Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

pza = 0
vc = 0

progreso.Show
AppActivate "Cacho"
progreso.ProgressBar1.Min = 0
progreso.ProgressBar1.Max = g1.Count

For Each entry In g1

    vc = vc + 1
    progreso.ProgressBar1 = vc
    progreso.Caption = Round((vc / progreso.ProgressBar1.Max) * 100, 2) & "%"
    progreso.com = progreso.Caption
    DoEvents

    If entry.ObjectName = "AcDbBlockReference" Then
        nom2 = UCase(entry.Name)
```

```
        progreso.com = LCase(nom2)

' Ajuste para Ver. 2012
            For k = 0 To Contador - 1
nom1 = UCase(Lista1(k))

                        If nom1 = nom2 Then
                            If Contador = 1 Then pza = pza + 1
                            varA = entry.insertionPoint

                            m(0) = varA(0)
m(1) = varA(1)
                            m(2) = 0
                            Tabla1.Conceptos = nom1

                            'dibuja la marca
                            dib = dibujar(m(0), m(1), m(2), radio_min, Tabla1.co,
Tabla1.Ma)
                            ' termina de dibujar la marca

                            'captura valores en tabla
                            pres = Presicion.ListIndex
                            axa = 0

                            ' escritura en tabla
If r > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla
tabla.Rows = tabla.Rows + 1
                            r = tabla.Rows - 1
tabla.TextMatrix(r, 0) = Etiqueta
                            tabla.TextMatrix(r, 1) = 1
                            tabla.TextMatrix(r, 2) = 0
                            tabla.TextMatrix(r, 3) = 0
                            tabla.TextMatrix(r, 4) = Altura
                            tabla.TextMatrix(r, 5) = "+"
                            tabla.TextMatrix(r, 6) = nom1
                            tabla.TextMatrix(r, 7) = LCompuesto
                            tabla.TextMatrix(r, 8) = co
                            tabla.TextMatrix(r, 9) = SeleccionDeMarca
tabla.TextMatrix(r, 10) = Mid(entry.ObjectName, 5, Len(entry.ObjectName))
                            tabla.TextMatrix(r, 11) = sg(entry.Handle)

                        End If

            Next

    End If

Next

Unload progreso
Update

Clipboard.Clear
If Contador = 1 Then Clipboard.SetText pza: MsgBox "Total " & nom1 & ": " & pza &
vbCr & vbCr & "Valor disponible en el portapapeles" &
vbCr & "Utilice Ctrl+V", vbInformation, "Suma de Texto"
```

```
500

End Sub
```

Rutina para determinar todos los bloques dela selección única en el plano.

```
Private Sub BlockVarios_Click
If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub
Dim AApp As Object
Dim ModelSpace As Object
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object
Dim bloque As Variant
Dim insertionPoint(0 To 2) As Double, Alignmentpoint(0 To 2) As Double
Dim m(0 To 2) As Double
Dim radio_min As Double
Dim ht As Double
Dim entry As AcadEntity
Dim nom1 As Variant, nom2 As Variant
Dim i As Variant
Dim r As Variant
Dim pres As Variant
Dim nombre As Variant
Dim pza As Double
Dim vc As Double

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
    If Err Then
        Err.Clear
        Set AApp = CreateObject("AutoCAD.Application")
        AApp.Visible = True
        If Err Then
            MsgBox Err.Description, vbCritical, "GRICC"
            Exit Sub
        End If
    End If

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

radio_min = TamanoMarca
ht = t_marca * 0.8


Tabla1.Hide
AppActivate "Autocad"
'opciones
```

```
X = Seleccion()

Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

pza = 0
vc = 0
progreso.Show
AppActivate "Cacho"
progreso.ProgressBar1.Min = 0
progreso.ProgressBar1.Max = g1.Count

For Each entry In g1
    vc = vc + 1
    progreso.ProgressBar1 = vc
    DoEvents

        If entry.ObjectName = "AcDbBlockReference" Then
            bloque = SinEspacios(UCase(entry.Name))
            progreso.com = LCase(bloque)
            progreso.Caption = Round((vc / progreso.ProgressBar1.Max) * 100, 2) &
"%"
varA = entry.insertionPoint
            nombre = bloque
            m(0) = varA(0)
            m(1) = varA(1)
            m(2) = 0


            ' agrega concepto
X = bloque
            If Trim(X) = "" Then GoTo 30

                For i = 0 To Conceptos.ListCount - 1
                    Conceptos.ListIndex = i
                    If Conceptos.Text = X Then
                        Conceptos = X
                        GoTo 30
                    End If
                Next

            Conceptos.AddItem X
            r = Conceptos.ListCount
Conceptos.ListIndex = r - 1
30

            'dibuja la marca
            X = dibujar(m(0), m(1), m(2), radio_min, co, Ma)


            'captura valores en tabla

            tabla.Rows = tabla.Rows + 1
            r = tabla.Rows - 1
            pres = Presicion.ListIndex

            If r > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla
```

```
            tabla.TextMatrix(r, 0) = r
            tabla.TextMatrix(r, 1) = 1
            tabla.TextMatrix(r, 2) = 0
            tabla.TextMatrix(r, 3) = 0
            tabla.TextMatrix(r, 4) = 0
            tabla.TextMatrix(r, 5) = "+"
            tabla.TextMatrix(r, 6) = bloque
            tabla.TextMatrix(r, 7) = LCompuesto
            tabla.TextMatrix(r, 8) = co
            tabla.TextMatrix(r, 9) = SeleccionDeMarca
tabla.TextMatrix(r, 10) = Mid(entry.ObjectName, 5, Len(entry.ObjectName))
            tabla.TextMatrix(r, 11) = UCase(entry.Handle)

        End If
    pza = pza + 1
Next

Unload progreso
Clipboard.SetText nom1

Update
AppActivate "Cacho"
GuardarOpciones
GuardarTabla (txtnombreArch)
FiltrarTabla (txtnombreArch)
Tabla1.Show

End Sub
```

Rutina para cuantificar piezas indicándolas con el mouse

```
Private Sub Piezas_Click

If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub

'On Error GoTo 1000
Dim i As Variant, h As Variant

Dim ht As Double
ht = TamanoMarca * 0.8

Dim textstring As Variant
Dim insertionPoint(0 To 2) As Double, Alignmentpoint(0 To 2) As Double
Dim textobj As AcadText
Dim AApp As Object
Dim ModelSpace As Object
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object
Dim po As Object
Dim entry As AcadEntity
Dim Handle As AcadEntity
Dim m(0 To 2) As Double
```

```
Dim n(0 To 5) As Double

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
If Err Then
Err.Clear
Set AApp = CreateObject("AutoCAD.Application")
AApp.Visible = True
If Err Then
MsgBox Err.Description, vbCritical, "GRICC"
Exit Sub
End If
End If

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility

Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

g01 = " GRICC "

Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

Dim points(0 To 9) As Double
Dim startPoint As Variant
Dim radio_min As Double
Dim circ As AcadCircle
Dim Altura As Double
Dim c As Variant, r As Variant
Dim PX As Double, X1, Y1, H1, H2, X2, Y2 As Double

Tabla1.Hide

    On Error Resume Next
    X = 0:

    For h = 1 To 1000

    Dim keywordList As String
    keywordList = "C F"

    ' Call InitializeUserInput to setup the keywords
    l.Utility.InitializeUserInput 128, keywordList

    ' Get the user input
    Dim returnPnt As Variant
    returnPnt = l.Utility.GetPoint(, "Punto " & h & " [ESC = Terminar]: ")

    m(0) = returnPnt(0)
    m(1) = returnPnt(1)
    m(2) = returnPnt(2)

    radio_min = TamanoMarca
```

```
    If Err Then
             Err.Clear
             GoTo 100
    Else

Dim dib As Variant
'dibuja la marca
dib = dibujar(m(0), m(1), m(2), radio_min, co, Ma)
' termina de dibujar la marca
Etiqueta = Etiqueta + 1
        'captura valores en tabla
Dim xm As String

       tabla.Rows = tabla.Rows + 1
tabla2.Rows = tabla2.Rows + 1
        r = tabla.Rows - 1

pres = Presicion.ListIndex

        If r > 1 Then tabla.FixedRows = 1

tabla.TextMatrix(r, 0) = Etiqueta
        tabla.TextMatrix(r, 1) = 1
        tabla.TextMatrix(r, 2) = 0
        tabla.TextMatrix(r, 3) = 0
        tabla.TextMatrix(r, 4) = Round(Tabla1.Altura, pres)
        tabla.TextMatrix(r, 5) = "+"
        tabla.TextMatrix(r, 6) = Conceptos
        tabla.TextMatrix(r, 7) = LCompuesto
        tabla.TextMatrix(r, 8) = co
        tabla.TextMatrix(r, 9) = SeleccionDeMarca
        tabla.TextMatrix(r, 10) = sg("_PZA")
tabla.TextMatrix(r, 11) = "0"

Update
End If
Next
100

GuardarOpciones
GuardarTabla (txtnombreArch)
FiltrarTabla (txtnombreArch)
AppActivate "Cacho"
Tabla1.Show
1100

End Sub
```

Menú Herramientas y Marca

En estos dos menús sólo se tienen controles de lectura los cuales no ejecuta ningún procedimiento sólo se hace referencia a estos. La lista de precisión, nos apoya para

redondear los valores de los objetos capturados y de las operaciones que se realizan. La lista de grosor, determina el ancho cuando se marca una polilínea. Todos los componentes del menú Marca son controles CheckBox, que nos ayudan a seleccionar la información que se va a escribir cuando se indica una marca a los objetos cuantificados.

Menú Aire

Menú especial para cuantificar la lámina de los ductos de aire acondicionado.

Botón para obtener los datos de la sección del ducto.

```
Private Sub Ducto_Click

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
    If Err Then
        Err.Clear
        Set AApp = CreateObject("AutoCAD.Application")
        AApp.Visible = True
            If Err Then
                MsgBox Err.Description, vbCritical, "GRICC"
            Exit Sub
    End If
End If

'Tabla1.Hide

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

AppActivate "Autocad"
X = Seleccion()

Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

'Tabla1.Show

For Each entry In g1

   DuctoName = FormatoDucto(entry.textstring)

Next

    X = DuctoName
    If Trim(X) = "" Then GoTo 30

        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
```

```
            If Conceptos.Text = X Then
                Conceptos = X
                GoTo 30
            End If
        Next

Conceptos.AddItem X
    r = Conceptos.ListCount
Conceptos.ListIndex = r - 1
    LCompuesto = 0
30

AppActivate "Cacho"

End Sub
```

Botón con el que se selecciona el tramo del ducto en el plano y de obtiene mediante una ecuación los Kg de lámina.

```
Private Sub GeneralAire_Click

' Botón de aire

Dim xlApp As Excel.Application
Dim xlWB As Excel.Workbook
Dim xlWS As Excel.Worksheet

On Error Resume Next
Set xlApp = GetObject(, "Excel.Application")
If Err Then
    MsgBox Err.Description, vbCritical, "GRICC || Excel"
    Exit Sub
End If

If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub
Dim m(0 To 2) As Double
Dim n(0 To 5) As Double

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
    If Err Then
        Err.Clear
        Set AApp = CreateObject("AutoCAD.Application")
        AApp.Visible = True
            If Err Then
                MsgBox Err.Description, vbCritical, "GRICC"
            Exit Sub
    End If
End If

'Tabla1.Hide

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
```

```
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

AppActivate "Autocad"
X = Seleccion()


Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

ht = TamanoMarca * 0.8
ATP = 0
i = 0
radio_min = TamanoMarca

vc = 0
progreso.Show
AppActivate "Cacho"
progreso.ProgressBar1.Min = 0
progreso.ProgressBar1.Max = g1.Count

For Each entry In g1

    vc = vc + 1
    progreso.ProgressBar1 = vc
    DoEvents

    If entry.Layer = "Cuant_NUM" Then GoTo 90
    If entry.ObjectName = "AcDbText" Then GoTo 90
    If entry.ObjectName = "AcDbMText" Then GoTo 90
    If entry.ObjectName = "AcDbEllipse" Then GoTo 90

    ' -- Hatch
    If entry.ObjectName = "AcDbHatch" Then
        A = entry.Area
        varA = entry.basePoint
        entry.GetBoundingBox minExt, maxExt
        m(0) = (minExt(0) + maxExt(0)) / 2
        m(1) = (minExt(1) + maxExt(1)) / 2
        GoTo 60
    End If

    ' -- Arco
    If entry.ObjectName = "AcDbArc" Then
        P = entry.ArcLength
        A = entry.Area
varA = entry.center

        'MsgBox entry.Radius
Pi = 3.14159265
        Sangle = entry.StartAngle * (180 / Pi)
        tangle = entry.TotalAngle * (180 / Pi)
        Mangle = Sangle + (tangle / 2)
cteY = entry.radius * Sin(Mangle * (Pi / 180))
        cteX = entry.radius * Cos(Mangle * (Pi / 180))
```

```
        m(0) = varA(0) + cteX: m(1) = varA(1) + cteY: m(2) = 0
        'm(0) = varA(0): m(1) = varA(1): m(2) = 0
GoTo 60
    End If

    ' -- Block
    If entry.ObjectName = "AcDbBlockReference" Then
bloque = SinEspacios(UCase(entry.Name))
varA = entry.insertionPoint
        m(0) = varA(0): m(1) = varA(1): m(2) = 0
        A = 0: P = 0
        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
            If bloque = Conceptos.Text Then GoTo 55
        Next
        Conceptos.AddItem bloque
55
        GoTo 60
    End If

    ' -- Círculo
    If entry.ObjectName = "AcDbCircle" Then
        P = 3.141592654 * ((entry.radius * 2))
        A = entry.Area
        entry.GetBoundingBox minExt, maxExt
        m(0) = (minExt(0) + maxExt(0)) / 2
        m(1) = (minExt(1) + maxExt(1)) / 2
        GoTo 60
    End If

    ' -- Polilínea
    If entry.ObjectName = "AcDbPolyline" Or entry.ObjectName = "AcDb2dPolyline"
Then
        P = entry.Length
        A = entry.Area

        vv = entry.Coordinates

            Dim e(0 To 3) As Double

        m(0) = (vv(2) + vv(0)) / 2
        m(1) = (vv(3) + vv(1)) / 2

        GoTo 60
    End If

    ' -- Línea
    If entry.ObjectName = "AcDbLine" Then
        A = 0
        P = entry.Length
        startPoint = entry.startPoint
        endPoint = entry.endPoint
        n(0) = startPoint(0): n(1) = startPoint(1): n(2) = startPoint(2)
        n(3) = endPoint(0): n(4) = endPoint(1): n(5) = endPoint(2)
        m(0) = (n(0) + n(3)) / 2
        m(1) = (n(1) + n(4)) / 2
        m(2) = 0
```

```
        GoTo 60
    End If
60
    co = SeleccionDeColor(colorX)
    entry.Color = co
    Etiqueta = Etiqueta + 1

'asigna propiedades
    entry.Highlight (False)
If borde.Value = True Then entry.Color = co: entry.ConstantWidth = Grosor
If Conceptos = "" Then Conceptos = "Ejemplo"

'captura valores en tabla
    pres = Presicion.ListIndex
r = tabla.Rows
tabla.Rows = r + 1

If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta

' Cálculo de lámina

        Dim SemiP, SemiP2, FactorU, Calibre, PesoU, PesoN, PesoT
        Dim DLargo, DAncho, DAlto, DAncho2, DAlto2

    DLargo = Round(P, pres)
    DAncho = separador1(DuctoName) ' ancho del ducto en in
    DAlto = separador2(DuctoName) ' alto del ducto en in

    DAncho2 = Round(DAncho * 25.4, 0) ' ancho del ducto en cm
    DAlto2 = Round(DAlto * 25.4, 0) ' alto del ducto en cm

    SemiP = Val(DAncho) + Val(DAlto) 'semiperímetro en in
    SemiP2 = Val(DAncho2) + Val(DAlto2) 'semiperímetro en in F+G

        peso26 = 4.04 ' T2
        peso24 = 4.65 ' T3
        peso22 = 6.49 ' T4
        peso20 = 7.71 ' T5
        peso18 = 10.15 ' T6

    ' ** ** ** ** ** Determinación del calibre de la lámina
If DAncho2 = 0 Then Calibre = 0: PesoU = 0
    If DAncho2 > 0 And DAncho2 <= 700 Then Calibre = 24: PesoU = peso24
    If DAncho2 > 700 And DAncho2 <= 1300 Then Calibre = 22: PesoU = peso22
    If DAncho2 > 1300 And DAncho2 <= 2000 Then Calibre = 20: PesoU = peso20
    If DAncho2 > 2000 Then Calibre = 18: PesoU = peso18
    ' ** ** ** ** **

FactorU = (1500 - SemiP2) / 1200 'celda K

If SemiP2 < 1500 Then PesoN = 2.63875 * (SemiP2 / 1000) * (1.12 ^ FactorU) * (1.3
- (0.01 * Calibre)) * PesoU
If SemiP2 >= 1500 Then PesoN = 2.63875 * (SemiP2 / 1000) * (1.3 - (0.01 *
Calibre)) * PesoU
PesoT = PesoN * DLargo
```

```
PesoDucto = Round(PesoT, pres)

'asigna propiedades
entry.Highlight (False)
If borde.Value = True Then entry.Color = co: entry.ConstantWidth = Grosor
If Conceptos = "" Then Conceptos = "Ejemplo"


    If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta
    tabla.TextMatrix(r, 1) = Round(DLargo, pres)
    tabla.TextMatrix(r, 2) = 0
tabla.TextMatrix(r, 3) = Round(PesoN, pres)
tabla.TextMatrix(r, 4) = Round(PesoDucto, pres)
    tabla.TextMatrix(r, 5) = "+"
    tabla.TextMatrix(r, 6) = DuctoName
    tabla.TextMatrix(r, 7) = Calibre
    tabla.TextMatrix(r, 8) = co
    tabla.TextMatrix(r, 9) = SeleccionDeMarca
tabla.TextMatrix(r, 10) = Mid(entry.ObjectName, 5, Len(entry.ObjectName))
    tabla.TextMatrix(r, 11) = Trim(UCase(entry.Handle))


'dibuja la marca

'borde
If borde.Value = True Then
Set El = MS.AddPolyline(n)
El.Color = co
El.ConstantWidth = LIN
End If

X = dibujar(m(0), m(1), m(2), radio_min, co, Ma)
If tabla.Rows > 1 Then tabla.FixedRows = 1
' termina de dibujar la marca

450
90

Next

If tabla.Rows > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla

Unload progreso
Update

200

AppActivate "Autocad"
'Tabla1.Show
GuardarOpciones

CLI = Conceptos.ListIndex

End Sub
```

Menú Acero

Menú especial con el cual mediante la selección de los perfiles de una lista, obtenemos el peso nominal de perfil, con el que obtenemos el total por elemento cuantificado.

```
Private Sub Perfil_Validate

CargarPerfiles (Perfil)
cte2 = CargarPESO(Perfil, Perfil2)

End Sub

Private Sub Perfil2_Validate
cte2 = CargarPESO(Perfil, Perfil2)
End Sub


Private Sub GeneralAcero_Click

' Botón de aire

Dim xlApp As Excel.Application
Dim xlWB As Excel.Workbook
Dim xlWS As Excel.Worksheet

On Error Resume Next
Set xlApp = GetObject(, "Excel.Application")
If Err Then
    MsgBox Err.Description, vbCritical, "GRICC || Excel"
    Exit Sub
End If

If Conceptos.Text = "" Then MsgBox "Definir concepto": Exit Sub
Dim m(0 To 2) As Double
Dim n(0 To 5) As Double

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
    If Err Then
        Err.Clear
        Set AApp = CreateObject("AutoCAD.Application")
        AApp.Visible = True
            If Err Then
                MsgBox Err.Description, vbCritical, "GRICC"
            Exit Sub
    End If
End If

'Tabla1.Hide

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
```

```
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

AppActivate "Autocad"
X = Seleccion()

Set g1 = AD.SelectionSets.Add("prueba" & X)
Call g1.SelectOnScreen

ht = TamanoMarca * 0.8
ATP = 0
i = 0
radio_min = TamanoMarca

vc = 0
progreso.Show
AppActivate "Cacho"
progreso.ProgressBar1.Min = 0
progreso.ProgressBar1.Max = g1.Count


    xc = Perfil & Perfil2
    If Trim(xc) = "" Then GoTo 30

        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
            If Conceptos.Text = xc Then
                Conceptos = xc
                GoTo 30
            End If
        Next

    Conceptos.AddItem xc
    r = Conceptos.ListCount
    Conceptos.ListIndex = r - 1
    LCompuesto = 0
30

For Each entry In g1

    vc = vc + 1
    progreso.ProgressBar1 = vc
    DoEvents

    If entry.Layer = "Cuant_NUM" Then GoTo 90
    If entry.ObjectName = "AcDbText" Then GoTo 90
    If entry.ObjectName = "AcDbMText" Then GoTo 90
    If entry.ObjectName = "AcDbEllipse" Then GoTo 90

    ' -- Hatch
    If entry.ObjectName = "AcDbHatch" Then
        A = entry.Area
        varA = entry.basePoint
        entry.GetBoundingBox minExt, maxExt
        m(0) = (minExt(0) + maxExt(0)) / 2
        m(1) = (minExt(1) + maxExt(1)) / 2
        GoTo 60
```

```
    End If

    ' -- Arco
    If entry.ObjectName = "AcDbArc" Then
        P = entry.ArcLength
        A = entry.Area
varA = entry.center

        'MsgBox entry.Radius
Pi = 3.14159265
        Sangle = entry.StartAngle * (180 / Pi)
        tangle = entry.TotalAngle * (180 / Pi)
        Mangle = Sangle + (tangle / 2)
cteY = entry.radius * Sin(Mangle * (Pi / 180))
        cteX = entry.radius * Cos(Mangle * (Pi / 180))

        m(0) = varA(0) + cteX: m(1) = varA(1) + cteY: m(2) = 0
        'm(0) = varA(0): m(1) = varA(1): m(2) = 0
GoTo 60
    End If

    ' -- Block
    If entry.ObjectName = "AcDbBlockReference" Then
bloque = SinEspacios(UCase(entry.Name))
varA = entry.insertionPoint
        m(0) = varA(0): m(1) = varA(1): m(2) = 0
        A = 0: P = 0
        For i = 0 To Conceptos.ListCount - 1
            Conceptos.ListIndex = i
            If bloque = Conceptos.Text Then GoTo 55
        Next
        Conceptos.AddItem bloque
55
        GoTo 60
    End If

    ' -- Círculo
    If entry.ObjectName = "AcDbCircle" Then
        P = 3.141592654 * ((entry.radius * 2))
        A = entry.Area
        entry.GetBoundingBox minExt, maxExt
        m(0) = (minExt(0) + maxExt(0)) / 2
        m(1) = (minExt(1) + maxExt(1)) / 2
        GoTo 60
    End If

    ' -- Polilínea
    If entry.ObjectName = "AcDbPolyline" Or entry.ObjectName = "AcDb2dPolyline"
Then
        P = entry.Length
        A = entry.Area

        vv = entry.Coordinates

            Dim e(0 To 3) As Double

        m(0) = (vv(2) + vv(0)) / 2
```

```
        m(1) = (vv(3) + vv(1)) / 2

        GoTo 60
    End If

    ' -- Línea
    If entry.ObjectName = "AcDbLine" Then
        A = 0
        P = entry.Length
        startPoint = entry.startPoint
        endPoint = entry.endPoint
        n(0) = startPoint(0): n(1) = startPoint(1): n(2) = startPoint(2)
        n(3) = endPoint(0): n(4) = endPoint(1): n(5) = endPoint(2)
        m(0) = (n(0) + n(3)) / 2
        m(1) = (n(1) + n(4)) / 2
        m(2) = 0
        GoTo 60
    End If
60
    co = SeleccionDeColor(colorX)
    entry.Color = co
    Etiqueta = Etiqueta + 1

'asigna propiedades
    entry.Highlight (False)
If borde.Value = True Then entry.Color = co: entry.ConstantWidth = Grosor
If Conceptos = "" Then Conceptos = "Ejemplo"

'captura valores en tabla
    pres = Presicion.ListIndex
r = tabla.Rows
    tabla.Rows = r + 1

If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta

' Cálculo de acero

        Dim SemiP, SemiP2, FactorU, Calibre, PesoU, PesoN, PesoT
        Dim DLargo, DAncho, DAlto, DAncho2, DAlto2

    DLargo = Round(P, pres)

    ' ** ** ** ** ** Determinación del calibre de la lámina

PesoN = PesoPerfil
PesoT = PesoN * DLargo
PesoDucto = Round(PesoT, pres)


'asigna propiedades
    entry.Highlight (False)
If borde.Value = True Then entry.Color = co: entry.ConstantWidth = Grosor
If Conceptos = "" Then Conceptos = "Ejemplo"

    If Trim(PREFIJO) = "" Then tabla.TextMatrix(r, 0) = Etiqueta Else
tabla.TextMatrix(r, 0) = PREFIJO & "-" & Etiqueta
```

```
    tabla.TextMatrix(r, 1) = Round(DLargo, pres)
    tabla.TextMatrix(r, 2) = 0
    tabla.TextMatrix(r, 3) = Round(PesoN, pres)
    tabla.TextMatrix(r, 4) = Round(PesoDucto, pres)
    tabla.TextMatrix(r, 5) = "+"
    tabla.TextMatrix(r, 6) = Perfil & Perfil2
    tabla.TextMatrix(r, 7) = 0
    tabla.TextMatrix(r, 8) = co
    tabla.TextMatrix(r, 9) = SeleccionDeMarca
tabla.TextMatrix(r, 10) = Mid(entry.ObjectName, 5, Len(entry.ObjectName))
    tabla.TextMatrix(r, 11) = Trim(UCase(entry.Handle))

'dibuja la marca

'borde
If borde.Value = True Then
Set El = MS.AddPolyline(n)
El.Color = co
El.ConstantWidth = LIN
End If

X = dibujar(m(0), m(1), m(2), radio_min, co, Ma)
If tabla.Rows > 1 Then tabla.FixedRows = 1
' termina de dibujar la marca

450
90

Next

If tabla.Rows > 1 Then tabla.FixedRows = 1: EncabezadoDeTabla

Unload progreso
Update

200

AppActivate "Autocad"
'Tabla1.Show
GuardarOpciones

CLI = Conceptos.ListIndex
GuardarTabla (txtnombreArch)
Conceptos.ListIndex = CLI
FiltrarTabla (txtnombreArch)

AppActivate "Cacho"

End Sub
```

Ventana Matrices

```
Private Sub Form_Load
```

```
' Se definen el encabezado de la tabla

tablaM.TextMatrix(0, 0) = "Código"
tablaM.TextMatrix(0, 1) = "Concepto"
tablaM.TextMatrix(0, 2) = "Método"
tablaM.TextMatrix(0, 3) = "Cantidad"
tablaM.TextMatrix(0, 4) = "Factor 1"
tablaM.TextMatrix(0, 5) = "Factor 2"
tablaM.TextMatrix(0, 6) = "Partida"
tablaM.TextMatrix(0, 7) = "Grupo"
tablaM.TextMatrix(0, 8) = "Unidad"

tablaM.ColWidth(1) = 2200
tablaM.ColWidth(2) = 700

' Agrega cuatro columnas
With ListView1
        .ColumnHeaders.Add , , "Código", 800
.ColumnHeaders.Add , , "Concepto", 2500
        .ColumnHeaders.Add , , "Partida"
        .ColumnHeaders.Add , , "Grupo"
        .ColumnHeaders.Add , , "Unidad"
    End With

CargarBase01

Matrices.Width = 6780

End Sub
```

Los procedimientos predecesores son de los controles de esta ventana

```
Private Sub List1_Click

Dim db As Database
Dim Fd As Field
Dim Tb As New TableDef
Dim i As Integer

sbase = LCase("C:\Cuantificador2012\Tablas\conceptos.mdb")
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * FROM grupos WHERE partida = '" & List1 & "'")
    List2.Clear
        Do Until Rs.EOF
            List2.AddItem UCase(Rs.Fields("grupo"))
            Rs.MoveNext
        Loop
Set Rs = Nothing
db.Close
CAgregar(0) = UCase(List1)
CAgregar(1) = UCase(List2)
End Sub
```

```
Private Sub List1_MouseDown
If Button = 2 Then
    Dim db As Database
    Dim Fd As Field
    Dim Tb As New TableDef
Dim i As Integer
    sbase = LCase("C:\Cuantificador2012\Tablas\conceptos.mdb")
    Set db = OpenDatabase(sbase)

    Z = UCase(Trim(InputBox("Nombre de la nueva partida")))
    If Z = "" Then MsgBox "No se agrego información": Exit Sub

Set Rs = db.OpenRecordset("SELECT * FROM partidas WHERE partida = '" & Z & "'")
    If Rs.EOF = False Then MsgBox "ya existe " & Z: Exit Sub

    ans01 = MsgBox("Agregar partida " & Z & "?", vbYesNo + vbInformation)
    If ans01 = vbNo Then MsgBox "No se agrego información": Exit Sub
        Set Rs = db.OpenRecordset("partidas")
            Rs.AddNew
            Rs.Fields("partida") = Z
            Rs.Update
        db.Close
    CargarBase01
End If

End Sub


Private Sub List2_Click

ListView1.ListItems.Clear
ListView1.Enabled = True


Dim db As Database
Dim Fd As Field
Dim Tb As New TableDef
Dim i As Integer
Dim SubElemento As ListItem

sbase = LCase("C:\Cuantificador2012\Tablas\conceptos.mdb")
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * FROM conceptos WHERE partida = '" & List1 &
"' and grupo = '" & List2 & "'")
        Do Until Rs.EOF
            If      UCase(Rs.Fields("partida"))      =      UCase(List1)      And
UCase(Rs.Fields("grupo")) = UCase(List2) Then
                r = r + 1
                    Set     SubElemento     =     ListView1.ListItems.Add(,     ,
UCase(Rs.Fields("codigo")))
                    SubElemento.SubItems(1) = Rs.Fields("descripcion")
                    SubElemento.SubItems(2) = UCase(Rs("partida"))
                    SubElemento.SubItems(3) = UCase(Rs("grupo"))
                    SubElemento.SubItems(4) = UCase(Rs.Fields("unidad"))
            End If
            Rs.MoveNext
```

```
        Loop
Set Rs = Nothing
db.Close
CAgregar(0) = UCase(List1)
CAgregar(1) = UCase(List2)
End Sub
```

Private Sub List2_MouseDown

```
If Button = 2 Then
    Dim db As Database
    Dim Fd As Field
    Dim Tb As New TableDef
Dim i As Integer
    sbase = LCase("C:\Cuantificador2012\Tablas\conceptos.mdb")
    Set db = OpenDatabase(sbase)

    Z = UCase(Trim(InputBox("Nombre del nuevo grupo")))
    If Z = "" Then MsgBox "No se agrego información": Exit Sub
    If List1 = "" Then MsgBox "Seleccionar una partida": Exit Sub

If List2.ListCount = 0 Then
        GoTo 50
    Else
        Set Rs = db.OpenRecordset("SELECT * FROM GRUPOs WHERE partida = '" &
List1 & "'")
    End If

    Do Until Rs.EOF
        If UCase(Rs("GRUPO")) = Z Then Exit Sub
        Rs.MoveNext
    Loop

    If Rs.EOF = False Then MsgBox "ya existe " & Z: Exit Sub

50
    ans01 = MsgBox("Agregar grupo " & UCase(Z) & "?", vbYesNo + vbInformation)
    If ans01 = vbNo Then MsgBox "No se agrego información": Exit Sub

        Set Rs = db.OpenRecordset("grupos")
            Rs.AddNew
            Rs.Fields("partida") = List1
            Rs.Fields("grupo") = Z
            Rs.Update
         db.Close
End If

End Sub
```

Private Sub ListView1_DblClick

```
BotAgregar.Enabled = True
r = tablaM.Rows

tablaM.Rows = r + 1
```

```
tablaM.TextMatrix(r, 0) = "Concepto"
tablaM.TextMatrix(r, 1) = "Concepto"
tablaM.TextMatrix(r, 2) = "Concepto"
tablaM.TextMatrix(r, 3) = "Concepto"

tablaM.TextMatrix(r, 0) = ListView1.SelectedItem
tablaM.TextMatrix(r, 1) = ListView1.SelectedItem.SubItems(1)
tablaM.TextMatrix(r, 2) = "Longitud"
tablaM.TextMatrix(r, 3) = 0
tablaM.TextMatrix(r, 4) = 1
tablaM.TextMatrix(r, 5) = 1
tablaM.TextMatrix(r, 6) = ListView1.SelectedItem.SubItems(2)
tablaM.TextMatrix(r, 7) = ListView1.SelectedItem.SubItems(3)
tablaM.TextMatrix(r, 8) = ListView1.SelectedItem.SubItems(4)

If tablaM.Rows > 1 Then tablaM.FixedRows = 1

End Sub
```

Private Sub BotAgregar_Click

```
X = UCase(SinEspacios(NombreConcepto))
If Trim(X) = "" Then GoTo 300

    For i = 0 To Tabla1.Conceptos.ListCount - 1
Tabla1.Conceptos.ListIndex = i
        If Tabla1.Conceptos.Text = X Then
MsgBox "Ya existe " & X, vbInformation
            Tabla1.Conceptos = X
GoTo 300
        End If
    Next

Tabla1.Conceptos.AddItem X
r = Tabla1.Conceptos.ListCount
Tabla1.Conceptos.ListIndex = r - 1


nombre = Tabla1.txtnombreArch
'nombre = "C:\Users\usuario\Documents\ejemplo3.mdb"

'MsgBox nombre

    Dim db As Database
Dim Fd As Field
    Dim Tb As New TableDef
    'Dim i As Integer

    sbase = LCase(nombre)
Set db = OpenDatabase(sbase)

        ' Guarda Tabla Conceptos
Set Rs = db.OpenRecordset("conceptos", dbOpenDynaset)

            If Rs.EOF And Rs.BOF Then Rs.AddNew Else Rs.MoveLast: Rs.AddNew
```

```
                Rs.Fields("Clave") = 0
                Rs.Fields("Codigo") = 0
Rs.Fields("Concepto") = UCase(NombreConcepto)
                Rs.Fields("Compuesto") = 1
            Rs.Update

100
    'Guarda la Tabla Compuesto
Set Rs = db.OpenRecordset("Compuesto", dbOpenDynaset)
    n = tablaM.Rows
        For i = 1 To n - 1
            If Rs.EOF And Rs.BOF Then Rs.AddNew Else Rs.MoveLast: Rs.AddNew

Rs.Fields("Clave") = UCase(NombreConcepto)
Rs.Fields("Codigo") = tablaM.TextMatrix(i, 0)
                Rs.Fields("Concepto") = tablaM.TextMatrix(i, 1)
                Rs.Fields("Metodo") = tablaM.TextMatrix(i, 2)
                Rs.Fields("Cantidad") = tablaM.TextMatrix(i, 3)
                Rs.Fields("Factor1") = tablaM.TextMatrix(i, 4)
                Rs.Fields("Factor2") = tablaM.TextMatrix(i, 5)
                Rs.Fields("Partida") = tablaM.TextMatrix(i, 6)
                Rs.Fields("Grupo") = tablaM.TextMatrix(i, 7)
                Rs.Fields("Unidad") = tablaM.TextMatrix(i, 8)
Rs.Update
        Next
200

' Importa de matrices a General
Tabla1.LCompuesto = 1

m = Tabla1.tablaC.Rows
    For i = 1 To n - 1
Tabla1.tablaC.Rows = m + i
        r = Tabla1.tablaC.Rows - 1

        Tabla1.tablaC.TextMatrix(r, 0) = UCase(NombreConcepto)
        Tabla1.tablaC.TextMatrix(r, 1) = tablaM.TextMatrix(i, 0)
        Tabla1.tablaC.TextMatrix(r, 2) = tablaM.TextMatrix(i, 1)
        Tabla1.tablaC.TextMatrix(r, 3) = tablaM.TextMatrix(i, 2)
        Tabla1.tablaC.TextMatrix(r, 4) = tablaM.TextMatrix(i, 3)
        Tabla1.tablaC.TextMatrix(r, 5) = tablaM.TextMatrix(i, 4)
        Tabla1.tablaC.TextMatrix(r, 6) = tablaM.TextMatrix(i, 5)
        Tabla1.tablaC.TextMatrix(r, 7) = tablaM.TextMatrix(i, 6)
        Tabla1.tablaC.TextMatrix(r, 8) = tablaM.TextMatrix(i, 7)

Next
db.Close
300
End Sub

Private Sub tablaM_DblClick

c = tablaM.Col: r = tablaM.Row
cte1 = tablaM.TextMatrix(0, c)

    If c = 2 Then
```

```
            If     UCase(tablaM.TextMatrix(r,     c))     =     "LONGITUD"     Then
tablaM.TextMatrix(r, c) = "AREA": GoTo 30
            If UCase(tablaM.TextMatrix(r, c)) = "AREA" Then tablaM.TextMatrix(r,
c) = "LXH": GoTo 30
            If UCase(tablaM.TextMatrix(r, c)) = "LXH" Then tablaM.TextMatrix(r,
c) = "PZA": GoTo 30
            If UCase(tablaM.TextMatrix(r, c)) = "PZA" Then tablaM.TextMatrix(r,
c) = "LONGITUD": GoTo 30
30
    End If

    If c >= 3 And c <= 5 And r > 0 Then
tablaM.TextMatrix(r, c) = InputBox("Modificar " & cte1, , tablaM.TextMatrix(r,
c))
End If

    If c = 0 And r > 0 Then

    varx = MsgBox("¿Desea Borrar " & tablaM.TextMatrix(r, 0) & " - " &
tablaM.TextMatrix(r, 1) & "?", vbInformation + vbYesNo)
    If varx = vbNo Then GoTo 200
    Screen.MousePointer = vbHourglass: tablaM.MousePointer = flexHourglass

    rc = tablaM.RowSel

    ReDim da(0 To tablaM.Rows, 0 To tablaM.Cols) As Variant

    For i = 1 To tablaM.Rows - 1
For j = 0 To tablaM.Cols - 1 'cambiar el valor sagún las columnas
da(i, j) = tablaM.TextMatrix(i, j)
        Next
    Next

    For i = 1 To tablaM.Rows - 1
For j = 0 To tablaM.Cols - 1 'cambiar el valor sagún las columnas
If i < rc Then tablaM.TextMatrix(i, j) = da(i, j)
            If i >= rc Then tablaM.TextMatrix(i, j) = da(i + 1, j)
        Next
    Next

    tablaM.Rows = tablaM.Rows - 1

    Screen.MousePointer = vbDefault: tablaM.MousePointer = flexDefault
200
    End If

    BotAgregar.Enabled = True

End Sub
```

Private Sub CAgregar_Change

```
If Index = 0 Then CAgregar(4) = Mid(CAgregar(0), 1, 1): GoTo 50
If Index = 1 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1):
GoTo 50
```

```
If Index = 2 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1): GoTo 50
If Index = 3 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1) & "0" & Round(Rnd() * 100, 0) & Round(Rnd() * 100, 0):
GoTo 50

50
End Sub
```

Private Sub CAgregar_Change

```
If Index = 0 Then CAgregar(4) = Mid(CAgregar(0), 1, 1): GoTo 50
If Index = 1 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1):
GoTo 50
If Index = 2 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1): GoTo 50
If Index = 3 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1) & "0" & Round(Rnd() * 100, 0) & Round(Rnd() * 100, 0):
GoTo 50

50
End Sub
```

Private Sub CAgregar_Change

```
If Index = 0 Then CAgregar(4) = Mid(CAgregar(0), 1, 1): GoTo 50
If Index = 1 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1):
GoTo 50
If Index = 2 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1): GoTo 50
If Index = 3 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1) & "0" & Round(Rnd() * 100, 0) & Round(Rnd() * 100, 0):
GoTo 50

50
End Sub
```

Private Sub CAgregar_Change

```
If Index = 0 Then CAgregar(4) = Mid(CAgregar(0), 1, 1): GoTo 50
If Index = 1 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1):
GoTo 50
If Index = 2 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1): GoTo 50
If Index = 3 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1) & "0" & Round(Rnd() * 100, 0) & Round(Rnd() * 100, 0):
GoTo 50

50
End Sub
```

Private Sub CAgregar_Change

```
If Index = 0 Then CAgregar(4) = Mid(CAgregar(0), 1, 1): GoTo 50
If Index = 1 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1):
GoTo 50
If Index = 2 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1): GoTo 50
If Index = 3 Then CAgregar(4) = Mid(CAgregar(0), 1, 1) & Mid(CAgregar(1), 1, 1) &
Mid(CAgregar(2), 1, 1) & "0" & Round(Rnd() * 100, 0) & Round(Rnd() * 100, 0):
GoTo 50

50
End Sub
```

Private Sub AgregarPartida_Click

```
For j = 0 To 4
    If CAgregar.Item(j) = "" Then MsgBox "Información incompleta": Exit Sub
Next

Dim db As Database
Dim Fd As Field
Dim Tb As New TableDef
Dim i As Integer
Dim cteValor, cteConcepto, cteMetodo, cteCantidad

sbase = LCase("C:\Cuantificador2012\Tablas\conceptos.mdb")
Set db = OpenDatabase(sbase)

Set Rs = db.OpenRecordset("conceptos", dbOpenDynaset)
    Do Until Rs.EOF
        If UCase(CAgregar(2)) = UCase(Rs.Fields("descripcion")) And _
         UCase(CAgregar(3)) = UCase(Rs.Fields("unidad")) And _
          UCase(CAgregar(4)) = UCase(Rs.Fields("codigo")) Then MsgBox "Exite el
concepto " & UCase(CAgregar(4)): Exit Sub
        Rs.MoveNext
    Loop
        Rs.AddNew
        Rs.Fields("partida") = UCase(CAgregar(0))
        Rs.Fields("grupo") = UCase(CAgregar(1))
        Rs.Fields("descripcion") = UCase(CAgregar(2))
        Rs.Fields("unidad") = UCase(CAgregar(3))
        Rs.Fields("codigo") = UCase(CAgregar(4))
Rs.Update

MsgBox "Se agregó la información!!"

db.Close

End Sub
```

Ventana Administrador

En esta ventana se administran los conceptos en formato matriz, exportar e Importar información en diferentes.

```
Private Sub BotArchBase_Click

On Error Resume Next

Dim s As Variant
    With CommonDialog1
.CancelError = True
        .FileName = Ruta2
        .Filter = "Cuantificación 2012|*.cu1"
        .InitDir = "C:\Cuantificador2012\Tablas"
        .FilterIndex = 1
.Flags = cdlOFNFileMustExist + cdlOFNHideReadOnly
        .ShowOpen

        If Err = 0 Then
            Ruta2 = .FileName
            s = .FileTitle
        End If

        Administrador.ListView1.ListItems.Clear
        Administrador.ListView2.ListItems.Clear

        Dim db As Database
        Dim Fd As Field
        Dim Tb As New TableDef
        Dim i As Integer
        Dim SubElemento As ListItem

        sbase = LCase(ruta1)
        Set db = OpenDatabase(sbase)

        Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM COMPUESTO
")
                Do Until Rs.EOF
                    r = r + 1
                    Administrador.ListView1.ListItems.Add , , Rs("clave"), 1
                    Rs.MoveNext
                Loop
        db.Close

        sbase = LCase(Ruta2)
        Set db = OpenDatabase(sbase)
        Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM COMPUESTO
")
                Do Until Rs.EOF
                    r = r + 1
                    Administrador.ListView2.ListItems.Add , , Rs("clave"), 2
                    Rs.MoveNext
                Loop
        db.Close
```

```
End With

End Sub
```

Private Sub ListView1_DblClick

```
If ListView1.ListItems.Count = 0 Then Exit Sub

sbase = LCase(ruta1)
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * from compuesto where Clave = '" &
ListView1.SelectedItem & "'")
clave = Rs("clave")
       Do Until Rs.EOF
                Z = Z & vbCr & Rs("codigo") & ": " & Rs("concepto") & ": " &
Rs("concepto")
Rs.MoveNext
        Loop
db.Close

MsgBox Z, vbInformation, clave

End Sub
```

Private Sub ListView1_MouseDown

```
If ListView1.ListItems.Count = 0 Then Exit Sub

If Button = 2 Then
    Unload Visor01


sbase = LCase(ruta1)
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * from compuesto where Clave = '" &
ListView1.SelectedItem & "'")
        clave = Rs("clave")
    r = 0
       Do Until Rs.EOF
                r = r + 1
                Visor01.tabla.Rows = r + 1
                Visor01.tabla.TextMatrix(r, 0) = r
                Visor01.tabla.TextMatrix(r, 1) = Rs("codigo")
                Visor01.tabla.TextMatrix(r, 2) = Rs("concepto")
                Visor01.tabla.TextMatrix(r, 3) = Rs("metodo")
                Visor01.tabla.TextMatrix(r, 4) = "(" & Rs("metodo") & " + " &
Rs("cantidad") & ") x " & Rs("factor1") & " x " & Rs("factor2")

                Rs.MoveNext
        Loop
db.Close

Visor01.tabla.TextMatrix(0, 1) = "Código"
    Visor01.tabla.TextMatrix(0, 2) = "Concepto"
```

```
    Visor01.tabla.TextMatrix(0, 3) = "Método"
    Visor01.tabla.TextMatrix(0, 4) = "Factor"
Visor01.tabla.ColWidth(0) = 500
    Visor01.tabla.ColWidth(1) = 1000
    Visor01.tabla.ColWidth(2) = 2500
    Visor01.tabla.ColWidth(3) = 1000
    Visor01.tabla.ColWidth(4) = 2000

    Visor01.Show
End If

End Sub
```

Private Sub ListView1_OLEDragDrop

```
Dim db As Database
Dim SubElemento As ListItem

' *** Rastrea Igual.
sbase = LCase(ruta1)
Set db = OpenDatabase(sbase)

h = ListView2.ListItems.Count
k = ListView1.ListItems.Count

For i = 1 To h
    For j = 1 To k
        If  ListView2.SelectedItem  =  ListView1.ListItems.Item(j)  Then  MsgBox
"Existe " & ListView2.SelectedItem, vbCritical: Exit Sub
    Next
Next

Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM COMPUESTO ")
        Do Until Rs.EOF
                If ListView2.SelectedItem = Rs("clave") Then MsgBox "Existe " &
ListView1.SelectedItem: Exit Sub
                Rs.MoveNext
        Loop
db.Close

    cteConcepto = UCase(SinEspacios(ListView2.SelectedItem))
    If Trim(cteConcepto) = "" Then Exit Sub

        For i = 0 To Tabla1.Conceptos.ListCount - 1
Tabla1.Conceptos.ListIndex = i
            If Tabla1.Conceptos.Text = cteConcepto Then
                MsgBox "Ya existe " & cteConcepto, vbInformation
                Tabla1.Conceptos = cteConcepto
                Exit Sub
            End If
        Next
    Tabla1.Conceptos.AddItem cteConcepto
    r = Tabla1.Conceptos.ListCount
    Tabla1.Conceptos.ListIndex = r - 1
```

```
    Tabla1.LCompuesto = 0

' *** Copiar

ListView1.ListItems.Add , , ListView2.SelectedItem, 2 ' copia el icono visual

sbase = LCase(Ruta2)
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * from compuesto where clave = '" &
ListView2.SelectedItem & "'")

   Set db2 = OpenDatabase(LCase(ruta1))
   Set Rs2 = db2.OpenRecordset("compuesto", dbOpenDynaset)

    r = Tabla1.tablaC.Rows
        Do Until Rs.EOF
If Rs2.EOF And Rs2.BOF Then Rs2.AddNew Else Rs2.MoveLast: Rs2.AddNew
Rs2("clave") = Rs("clave")
                        Rs2("codigo") = Rs("codigo")
                        Rs2("concepto") = Rs("concepto")
                        Rs2("metodo") = Rs("metodo")
                        Rs2("cantidad") = Rs("cantidad")
                        Rs2("factor1") = Rs("factor1")
                        Rs2("factor2") = Rs("factor2")
                        Rs2("partida") = Rs("partida")
                        Rs2("grupo") = Rs("grupo")
                        Rs2("unidad") = Rs("unidad")
                        Rs2.Update

                        Tabla1.tablaC.Rows = r + 1
                        Tabla1.tablaC.TextMatrix(r, 0) = Rs.Fields("CLAVE")
                        Tabla1.tablaC.TextMatrix(r, 1) = Rs.Fields("CODIGO")
                        Tabla1.tablaC.TextMatrix(r, 2) = Rs.Fields("CONCEPTO")
                        Tabla1.tablaC.TextMatrix(r, 3) = Rs.Fields("METODO")
                        Tabla1.tablaC.TextMatrix(r, 4) = Rs.Fields("CANTIDAD")
                        Tabla1.tablaC.TextMatrix(r, 5) = Rs.Fields("FACTOR1")
                        Tabla1.tablaC.TextMatrix(r, 6) = Rs.Fields("FACTOR2")
                        Tabla1.tablaC.TextMatrix(r, 7) = Rs.Fields("PARTIDA")
                        Tabla1.tablaC.TextMatrix(r, 8) = Rs.Fields("GRUPO")
r = r + 1
            Rs.MoveNext
            'Rs2.MoveNext
        Loop

db.Close
db2.Close

End Sub


Private Sub ListView2_MouseDown
If ListView2.ListItems.Count = 0 Then Exit Sub

If Button = 2 Then
    Unload Visor01
```

```
sbase = LCase(Ruta2)
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * from compuesto where Clave = '" &
ListView2.SelectedItem & "'")
        clave = Rs("clave")
    r = 0
       Do Until Rs.EOF
                r = r + 1
                Visor01.tabla.Rows = r + 1
                Visor01.tabla.TextMatrix(r, 0) = r
                Visor01.tabla.TextMatrix(r, 1) = Rs("codigo")
                Visor01.tabla.TextMatrix(r, 2) = Rs("concepto")
                Visor01.tabla.TextMatrix(r, 3) = Rs("metodo")
                Visor01.tabla.TextMatrix(r, 4) = "(" & Rs("metodo") & " + " &
Rs("cantidad") & ") x " & Rs("factor1") & " x " & Rs("factor2")

                Rs.MoveNext
        Loop
db.Close

Visor01.tabla.TextMatrix(0, 1) = "Código"
    Visor01.tabla.TextMatrix(0, 2) = "Concepto"
    Visor01.tabla.TextMatrix(0, 3) = "Método"
    Visor01.tabla.TextMatrix(0, 4) = "Factor"
Visor01.tabla.ColWidth(0) = 500
    Visor01.tabla.ColWidth(1) = 1000
    Visor01.tabla.ColWidth(2) = 2500
    Visor01.tabla.ColWidth(3) = 1000
    Visor01.tabla.ColWidth(4) = 2000

    Visor01.Show
End If

End Sub
```

Private Sub ListView2_OLEDragDrop

```
Dim db As Database
Dim SubElemento As ListItem

' *** Rastrea Igual.
sbase = LCase(Ruta2)
Set db = OpenDatabase(sbase)

h = ListView1.ListItems.Count
k = ListView2.ListItems.Count

For i = 1 To h
    For j = 1 To k
        If ListView1.SelectedItem = ListView2.ListItems.Item(j) Then MsgBox
"Existe " & ListView1.SelectedItem, vbCritical: Exit Sub
    Next
Next
```

```
Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM COMPUESTO ")
        Do Until Rs.EOF
                If ListView1.SelectedItem = Rs("clave") Then MsgBox "Existe " &
ListView1.SelectedItem: Exit Sub
                Rs.MoveNext
        Loop
db.Close

' *** Copiar

ListView2.ListItems.Add , , ListView1.SelectedItem, 1 ' copia el icono visual

sbase = LCase(ruta1)
Set db = OpenDatabase(sbase)
Set Rs = db.OpenRecordset("SELECT * from compuesto where clave = '" &
ListView1.SelectedItem & "'")

   Set db2 = OpenDatabase(LCase(Ruta2))
   Set Rs2 = db2.OpenRecordset("compuesto", dbOpenDynaset)

        Do Until Rs.EOF
If Rs2.EOF And Rs2.BOF Then Rs2.AddNew Else Rs2.MoveLast: Rs2.AddNew
Rs2("clave") = Rs("clave")
                        Rs2("codigo") = Rs("codigo")
                        Rs2("concepto") = Rs("concepto")
                        Rs2("metodo") = Rs("metodo")
                        Rs2("cantidad") = Rs("cantidad")
                        Rs2("factor1") = Rs("factor1")
                        Rs2("factor2") = Rs("factor2")
                        Rs2("partida") = Rs("partida")
Rs2("grupo") = Rs("grupo")
                        Rs2("unidad") = Rs("unidad")
Rs2.Update
Rs.MoveNext
        Loop

db.Close
db2.Close

End Sub
```

Ventana Reporteador

En esta ventana administramos toda la información recabada del plano de Autocad, y generan reportes hacia Excel o Autocad.

```
Private Sub Form_Load
ruta = Tabla1.txtnombreArch

Arbol.Nodes.Add , , "A1", "Reportes", 3
    Arbol.Nodes.Add "A1", tvwChild, "A11", "Reporte 1", 1
    Arbol.Nodes.Add "A1", tvwChild, "A12", "Reporte 2", 1
    Arbol.Nodes.Add "A1", tvwChild, "A13", "Reporte 3", 1

Arbol.Nodes.Add , , "A2", "Generador", 3
```

```
Arbol.Nodes.Add "A2", tvwChild, "A21", "General", 1
Arbol.Nodes.Add "A2", tvwChild, "A22", "Por concepto", 1
Arbol.Nodes.Add "A2", tvwChild, "A23", "Resumen", 1
'Arbol.Nodes.Add "A2", tvwChild, "A24", "Personzalizado", 6
    CargarConceptosA22
    'CargarConceptosA24

'Arbol.Nodes.Add , , "A3", "Exportar", 3
    'Arbol.Nodes.Add "A3", tvwChild, "A31", "Exportar a Excel", 6
    'Arbol.Nodes.Add "A3", tvwChild, "A32", "Exportar a Autocad", 6

Arbol.Nodes.Add , , "A4", "Conceptos", 3
    CargarConceptosA4

Set db = OpenDatabase(LCase(ruta))

Set Rs2 = db.OpenRecordset("SELECT DISTINCT valores.concepto FROM Valores")
    tabla.Cols = 6
        Do Until Rs2.EOF
acumuladol = 0
            acumulado1 = 0
            acumuladoA = 0
            acumulado2 = 0
            acumuladolxh = 0
            acumulado3 = 0
            acumuladoPZA = 0
            acumulado4 = 0

Set RsL = db.OpenRecordset("SELECT sum(longitud) as total FROM valores where
concepto = '" & Rs2("concepto") & "'")
Set RsA = db.OpenRecordset("SELECT sum(area) as total FROM valores where concepto
= '" & Rs2("concepto") & "'")
Set RsLxH = db.OpenRecordset("SELECT sum(lxh) as total FROM valores where
concepto = '" & Rs2("concepto") & "'")
Set RsPZA = db.OpenRecordset("SELECT COUNT(*) as total FROM valores where
concepto = '" & Rs2("concepto") & "'")
Tl = RsL("total")
                        TA = RsA("total")
TLxH = RsLxH("total")
                        Tpza = RsPZA("total")

                        Reporteador.tabla.Rows = Reporteador.tabla.Rows + 1
                        i = i + 1
                        tabla.TextMatrix(i, 0) = i
                        tabla.TextMatrix(i, 1) = Rs2("concepto")
                        tabla.TextMatrix(i, 2) = Round(Tl, pres)
                        tabla.TextMatrix(i, 3) = Round(TA, pres)
                        tabla.TextMatrix(i, 4) = Round(TLxH, pres)
                        tabla.TextMatrix(i, 5) = Round(Tpza, pres)

            Rs2.MoveNext
            Loop
        tabla.TextMatrix(0, 0) = "No."
tabla.TextMatrix(0, 1) = "Concepto"
        tabla.TextMatrix(0, 2) = "Longitud"
        tabla.TextMatrix(0, 3) = "Área"
        tabla.TextMatrix(0, 4) = "L x h"
```

```
        tabla.TextMatrix(0, 5) = "Pza"

        tabla.ColWidth(0) = 500
        tabla.ColWidth(1) = 2500
tabla.ColWidth(2) = 1000
        tabla.ColWidth(3) = 1000
        tabla.ColWidth(4) = 1000
        tabla.ColWidth(5) = 1000

        If tabla.Rows > 1 Then tabla.FixedRows = 1
    db.Close
Exit Sub

End Sub
```

Private Sub Arbol_DblClick

```
ImpresionPersonalizada.Enabled = False
CapturaImagen.Enabled = False

If Arbol.SelectedItem.Parent Is Nothing Then Exit Sub
resultado = UCase(Arbol.SelectedItem.Parent)

If resultado = "CONCEPTOS" Then Titulo = "m." & UCase(Arbol.SelectedItem)

If resultado <> "EXPORTAR" Then Titulo = UCase(Arbol.SelectedItem)
If resultado = "CONCEPTOS" Then Titulo = "m." & UCase(Arbol.SelectedItem)

If resultado = "EXPORTAR" And Arbol.SelectedItem = "Exportar a Excel" Then
    If tabla.Rows = 1 Or tabla.Cols = 1 Or tabla.TextMatrix(0, 0) = "" Then Exit
Sub
        c = tabla.Cols
        r = tabla.Rows

            Dim xlApp As Excel.Application
            Dim xlWB As Excel.Workbook
            Dim xlWS As Excel.Worksheet

            On Error Resume Next
            Set xlApp = GetObject(, "Excel.Application")
                If Err Then
                    Err.Clear
                    Set xlApp = CreateObject("Excel.Application")
                    xlApp.Workbooks.Add
                    AppActivate "Excel"
                    If Err Then
                        MsgBox Err.Description, vbCritical, "GRICC"
                        Exit Sub
                End If
            End If

    xlApp.Visible = True

    X = xlApp.ActiveWorkbook.Name
    Y = xlApp.ActiveSheet.Name
```

```
    Z = xlApp.Sheets.Count

For i = 1 To Z
    If UCase(xlApp.Sheets(i).Name) = Titulo Then GoTo 10
Next
    xlApp.Sheets.Add
    xlApp.ActiveSheet.Name = Titulo
10

   xlApp.Sheets(Titulo).Select


   xlApp.Range("a1:z10000").Clear

        For h = 1 To r
            For k = 1 To c
                xlApp.Cells(h, k) = tabla.TextMatrix(h - 1, k - 1)
            Next
        Next

        xlApp.Range(xlApp.Cells(1, 1), xlApp.Cells(1, c)).Interior.Color = RGB(0,
0, 0)
        xlApp.Range(xlApp.Cells(1, 1), xlApp.Cells(1, c)).Font.Color = RGB(255,
255, 255)
        xlApp.Range(xlApp.Cells(1, 1), xlApp.Cells(1, c)).HorizontalAlignment =
xlCenter
AppActivate "Cacho"
Exit Sub
End If

If resultado = "EXPORTAR" And Arbol.SelectedItem = "Exportar a Autocad" Then
    If tabla.Rows = 1 Or tabla.Cols = 1 Or tabla.TextMatrix(0, 0) = "" Then Exit
Sub
        c = tabla.Cols
        r = tabla.Rows

Dim ptoI1(0 To 2) As Double
Dim coorX As Double
Dim coorY As Double

Dim AApp As Object
Dim ModelSpace As Object
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object

Dim DIF As Double
Dim ht As Double

ht = Tabla1.Altura
DIF = (ht * 1.1)

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
```

```
AppActivate "Autocad"
ptoi = l.Utility.GetPoint(, vbCr & "Punto para insertar información")
coorX = Format(ptoi(0), "0")
coorY = Format(ptoi(1), "0")

Dim d As Double, sep As Double

sepY = ht * 1.1
sepX = ht * 2.2

For h = 0 To r - 1
    For k = 0 To c - 1
        ancho = 0
        For q = 1 To k
            ancho = ancho + (tabla.ColWidth(q - 1) * 0.01)
        Next

        ptoI1(0) = coorX + (sepX * k) + ancho
        ptoI1(1) = coorY - (sepY * h)

        Set El = MS.AddText(tabla.TextMatrix(h, k), ptoI1, ht)
ptoI1(0) = ptoI1(0) - (0.5 * sep)
        El.Color = Tabla1.co

Next

Next

AppActivate "Cacho"
Exit Sub
End If


Dim Rs
r = 0
tabla.Rows = 1
tabla.Cols = 1

Set db = OpenDatabase(LCase(ruta))

' ***** Visor por conceptos
If resultado = "CONCEPTOS" Then
Set Rs = db.OpenRecordset("select * from compuesto where clave = '" &
Arbol.SelectedItem & "'")

        tabla.Cols = 5
        If Rs.EOF And Rs.BOF Then GoTo 200
        Rs.MoveFirst

        Do Until Rs.EOF
            r = r + 1
            tabla.Rows = r + 1
            tabla.TextMatrix(0, 0) = Arbol.SelectedItem
            tabla.TextMatrix(r, 0) = r
tabla.TextMatrix(r, 1) = Rs("codigo")
            tabla.TextMatrix(r, 2) = Rs("concepto")
            tabla.TextMatrix(r, 3) = Rs("metodo")
```

```
            tabla.TextMatrix(r, 4) = "(" & Rs("metodo") & " + " & Rs("cantidad")
& ") x " & Rs("factor1") & " x " & Rs("factor2")
            Rs.MoveNext
        Loop

        tabla.TextMatrix(0, 1) = "Código"
        tabla.TextMatrix(0, 2) = "Concepto"
        tabla.TextMatrix(0, 3) = "Método"
        tabla.TextMatrix(0, 4) = "Factor"
tabla.ColWidth(0) = 1000
        tabla.ColWidth(1) = 1000
        tabla.ColWidth(2) = 2500
        tabla.ColWidth(3) = 1000
        tabla.ColWidth(4) = 2000

        If tabla.Rows > 1 Then tabla.FixedRows = 1
200
    db.Close
    Exit Sub

End If

' ***** Reporte 1
If resultado = "REPORTES" And Arbol.SelectedItem = "Reporte 1" Then
            tabla.Cols = 12
    Set Rs1 = db.OpenRecordset("conceptos")
        Do Until Rs1.EOF

            Set Rs2 = db.OpenRecordset("SELECT * FROM compuesto WHERE Clave = '"
& Rs1("concepto") & "'")

                Do Until Rs2.EOF

                    Set RsL1 = db.OpenRecordset("SELECT * FROM valores where
concepto = '" & Rs1("concepto") & "'")
                        If RsL1.EOF And RsL1.BOF Then
                            Tl = 0: TA = 0: TLxH = 0: Tpza = 0
                        Else
                            Set RsL = db.OpenRecordset("SELECT sum(longitud) as
total FROM valores where concepto = '" & Rs1("concepto") & "'")
                            Set RsA = db.OpenRecordset("SELECT sum(area) as total
FROM valores where concepto = '" & Rs1("concepto") & "'")
                            Set RsLxH = db.OpenRecordset("SELECT sum(lxh) as
total FROM valores where concepto = '" & Rs1("concepto") & "'")
                            Set RsPZA = db.OpenRecordset("SELECT COUNT(*) as
total FROM valores where concepto = '" & Rs1("concepto") & "'")
Tl = RsL("total") + (RsPZA("total") * Rs2("cantidad"))
                            TA = RsA("total") + (RsPZA("total") *
Rs2("cantidad"))
                            TLxH = RsLxH("total") + (RsPZA("total") *
Rs2("cantidad"))
                            Tpza = RsPZA("total") + (RsPZA("total") *
Rs2("cantidad"))
End If

                    Set Rs4 = db.OpenRecordset("SELECT * FROM compuesto where
clave = '" & Rs2("clave") & "' and concepto = '" & Rs2("concepto") & "'")
```

```
                    Set Rs5 = db.OpenRecordset("SELECT * FROM valores INNER JOIN
compuesto ON valores.concepto = compuesto.clave where compuesto.clave = '" &
Rs2("clave") & "' and compuesto.concepto = '" & Rs2("concepto") & "'")

                        Do Until Rs4.EOF
                            tabla.Rows = tabla.Rows + 1
                            i = i + 1
                            tabla.TextMatrix(i, 0) = i
                            tabla.TextMatrix(i, 1) = Rs1("concepto")
                            tabla.TextMatrix(i, 2) = Rs2("concepto")
                            tabla.TextMatrix(i, 3) = "(" & Rs4("metodo") & " + "
& Rs4("cantidad") & ") x " & Rs4("factor1") & " x " & Rs4("factor2") ' factor
                            tabla.TextMatrix(i, 4) = Round(Tl, pres)
                            tabla.TextMatrix(i, 5) = Round(tabla.TextMatrix(i, 4)
* Rs4("factor1") * Rs4("factor2"), pres)
                            tabla.TextMatrix(i, 6) = Round(TA, pres)
                            tabla.TextMatrix(i, 7) = Round(tabla.TextMatrix(i, 6)
* Rs4("factor1") * Rs4("factor2"), pres)
                            tabla.TextMatrix(i, 8) = Round(TLxH, pres)
                            tabla.TextMatrix(i, 9) = Round(tabla.TextMatrix(i, 8)
* Rs4("factor1") * Rs4("factor2"), pres)
                            tabla.TextMatrix(i, 10) = Round(Tpza, pres)
                            tabla.TextMatrix(i,  11)  =  Round(tabla.TextMatrix(i,
10) * Rs4("factor1") * Rs4("factor2"), pres)

                            Rs4.MoveNext
                        Loop

                Rs2.MoveNext
                Loop

        Rs1.MoveNext
        Loop

db.Close

tabla.TextMatrix(0, 0) = "No."
tabla.TextMatrix(0, 1) = "Concepto"
tabla.TextMatrix(0, 2) = "Compuesto"
tabla.TextMatrix(0, 3) = "Factor (f)"
tabla.TextMatrix(0, 4) = "Long (l)"
tabla.TextMatrix(0, 5) = "l x f"
tabla.TextMatrix(0, 6) = "Área (A)"
tabla.TextMatrix(0, 7) = "A x f"
tabla.TextMatrix(0, 8) = "l x h"
tabla.TextMatrix(0, 9) = "l x h x f"
tabla.TextMatrix(0, 10) = "Pza"
tabla.TextMatrix(0, 11) = "Pza x f"

tabla.ColWidth(0) = 500
tabla.ColWidth(1) = 1000
tabla.ColWidth(2) = 2300
tabla.ColWidth(3) = 2000
tabla.ColWidth(4) = 900
tabla.ColWidth(5) = 900
tabla.ColWidth(6) = 900
tabla.ColWidth(7) = 900
```

```
tabla.ColWidth(8) = 900
tabla.ColWidth(9) = 900
tabla.ColWidth(10) = 700
tabla.ColWidth(11) = 700

'tabla.BackColorSel = RGB(204, 236, 255)
tabla.BackColorSel = &HC0E0FF
tabla.BackColorFixed = RGB(0, 80, 150)

If tabla.Rows > 1 Then tabla.FixedRows = 1

Exit Sub

End If


' ***** Reporte 2
If resultado = "REPORTES" And Arbol.SelectedItem = "Reporte 2" Then

    Set Rs2 = db.OpenRecordset("SELECT DISTINCT compuesto.codigo FROM COMPUESTO
")
    tabla.Cols = 12
        Do Until Rs2.EOF
acumuladol = 0
            acumulado1 = 0
            acumuladoA = 0
            acumulado2 = 0
            acumuladolxh = 0
            acumulado3 = 0
            acumuladoPZA = 0
            acumulado4 = 0
                Set Rs1 = db.OpenRecordset("SELECT * FROM valores INNER JOIN
compuesto ON valores.concepto = compuesto.clave where compuesto.codigo = '" &
Rs2.Fields("codigo") & "'")
                    'MsgBox rs1("compuesto.concepto")
                    r = r + 1
tabla.Rows = r + 1
                    Do Until Rs1.EOF
                        If Rs2("codigo") = Rs1.Fields("codigo") Then

acumuladol = Round(Rs1("longitud") + acumuladol, pres)
                            acumulado1 = Round(acumulado1 + ((Rs1("longitud") +
Rs1("cantidad")) * Rs1("factor1") * Rs1("factor2")), pres)
                            acumuladoA = Round(Rs1("area") + acumuladoA, pres)
                            acumulado2 = Round(acumulado2 + ((Rs1("area") +
Rs1("cantidad")) * Rs1("factor1") * Rs1("factor2")), pres)
                            acumuladolxh = Round(Rs1("lxh") + acumuladolxh, pres)
                            acumulado3 = Round(acumulado3 + ((Rs1("lxh") +
Rs1("cantidad")) * Rs1("factor1") * Rs1("factor2")), pres)
                            acumuladoPZA = Round(1 + acumuladoPZA, pres)
                            acumulado4 = Round(acumulado4 + ((1 +
Rs1("cantidad")) * Rs1("factor1") * Rs1("factor2")), pres)

                                tabla.TextMatrix(r, 0) = r
                                tabla.TextMatrix(r, 1) = Rs2("codigo")
                                tabla.TextMatrix(r,             2)             =
Rs1("compuesto.concepto")
```

```
                                    tabla.TextMatrix(r, 3) = Rs1("metodo")
                                    tabla.TextMatrix(r, 4) = acumuladol
                                    tabla.TextMatrix(r, 5) = acumulado1
                                    tabla.TextMatrix(r, 6) = acumuladoA
                                    tabla.TextMatrix(r, 7) = acumulado2
                                    tabla.TextMatrix(r, 8) = acumuladolxh
                                    tabla.TextMatrix(r, 9) = acumulado3
                                    tabla.TextMatrix(r, 10) = acumuladoPZA
                                    tabla.TextMatrix(r, 11) = acumulado4

End If
                        Rs1.MoveNext
                    Loop


            Rs2.MoveNext
Loop
        tabla.TextMatrix(0, 0) = "No."
        tabla.TextMatrix(0, 1) = "Código"
        tabla.TextMatrix(0, 2) = "Concepto"
        tabla.TextMatrix(0, 3) = "Método"
        tabla.TextMatrix(0, 4) = "l"
        tabla.TextMatrix(0, 5) = "l x f"
        tabla.TextMatrix(0, 6) = "A"
        tabla.TextMatrix(0, 7) = "A x f"
        tabla.TextMatrix(0, 8) = "l x h"
        tabla.TextMatrix(0, 9) = "l x h x f"
        tabla.TextMatrix(0, 10) = "PZA"
        tabla.TextMatrix(0, 11) = "PZA x f"
tabla.ColWidth(0) = 500
        tabla.ColWidth(1) = 1000
        tabla.ColWidth(2) = 2500
        lx = 900
        tabla.ColWidth(3) = lx
        tabla.ColWidth(4) = lx
        tabla.ColWidth(5) = lx
        tabla.ColWidth(6) = lx
        tabla.ColWidth(7) = lx
        tabla.ColWidth(8) = lx
        tabla.ColWidth(9) = lx

        If tabla.Rows > 1 Then tabla.FixedRows = 1

    db.Close
    Exit Sub
End If


' ***** Reporte 3
If resultado = "REPORTES" And Arbol.SelectedItem = "Reporte 3" Then

        tabla.Cols = 13
        Set Rs = db.OpenRecordset("select * from valores order by concepto ASC,
ID ASC")
        Set Rs = db.OpenRecordset("SELECT * FROM valores INNER JOIN compuesto ON
valores.concepto = compuesto.clave  order by valores.id ASC")
            Do Until Rs.EOF
```

```
                                r = r + 1
tabla.Rows = r + 1
                                tabla.TextMatrix(r, 0) = r
                                tabla.TextMatrix(r, 1) = Rs("no")
                                tabla.TextMatrix(r, 2) = Rs("valores.concepto")
                                tabla.TextMatrix(r, 3) = Rs("codigo")
                                tabla.TextMatrix(r, 4) = Rs("compuesto.concepto")
tabla.TextMatrix(r, 5) = Rs("longitud")
                                tabla.TextMatrix(r, 6) = Rs("area")
tabla.TextMatrix(r, 7) = Rs("altura")
                                tabla.TextMatrix(r, 8) = Rs("lxh")
                                tabla.TextMatrix(r, 9) = "(" & Rs("metodo") & "+"
& Rs("cantidad") & ")x" & Rs("factor1") & "x" & Rs("factor2")

                                tabla.TextMatrix(r,   10)   =   (Rs("longitud")   +
Rs("cantidad")) * Rs("factor1") * Rs("factor2")
tabla.TextMatrix(r,  11)  =  (Rs("area")  +  Rs("cantidad"))  *  Rs("factor1")  *
Rs("factor2")
                                tabla.TextMatrix(r,    12)    =    (Rs("lxh")    +
Rs("cantidad")) * Rs("factor1") * Rs("factor2")
                Rs.MoveNext
            Loop

tabla.TextMatrix(0, 0) = "ID"
tabla.TextMatrix(0, 1) = "No."
tabla.TextMatrix(0, 2) = "Concepto"
tabla.TextMatrix(0, 3) = "Clave"
tabla.TextMatrix(0, 4) = "Material"
tabla.TextMatrix(0, 5) = "Longitud (l)"
tabla.TextMatrix(0, 6) = "Área (A)"
tabla.TextMatrix(0, 7) = "Altura (h)"
tabla.TextMatrix(0, 8) = "l * h"
tabla.TextMatrix(0, 9) = "Factor"
tabla.TextMatrix(0, 10) = "l x f"
tabla.TextMatrix(0, 11) = "A x f"
tabla.TextMatrix(0, 12) = "l x h x f"

tabla.ColWidth(0) = 500
tabla.ColWidth(1) = 500
tabla.ColWidth(2) = 1000
tabla.ColWidth(3) = 1000
tabla.ColWidth(4) = 2500
tabla.ColWidth(5) = 900
tabla.ColWidth(6) = 900
tabla.ColWidth(7) = 700
tabla.ColWidth(8) = 700
tabla.ColWidth(9) = 2000

If tabla.Rows > 1 Then tabla.FixedRows = 1
Exit Sub

End If

' ***** Generador General
If resultado = "GENERADOR" And Arbol.SelectedItem = "General" Then
        tabla.Cols = 9
```

```
        Set Rs = db.OpenRecordset("select * from valores order by concepto ASC,
ID ASC")
            Do Until Rs.EOF
                                r = r + 1
tabla.Rows = r + 1
tabla.TextMatrix(r, 0) = r
                                tabla.TextMatrix(r, 1) = Rs("no")
                                tabla.TextMatrix(r, 2) = Rs("concepto")
tabla.TextMatrix(r, 3) = Rs("longitud")
                                tabla.TextMatrix(r, 4) = Rs("area")
tabla.TextMatrix(r, 5) = Rs("altura")
                                tabla.TextMatrix(r, 6) = Rs("lxh")
                                tabla.TextMatrix(r, 7) = Rs("compuesto")
                                tabla.TextMatrix(r, 8) = Rs("valor")
Rs.MoveNext
            Loop

tabla.TextMatrix(0, 0) = "ID"
tabla.TextMatrix(0, 1) = "No."
tabla.TextMatrix(0, 2) = "Concepto"
tabla.TextMatrix(0, 3) = "Long. (l)"
tabla.TextMatrix(0, 4) = "Área (A)"
tabla.TextMatrix(0, 5) = "Altura (h)"
tabla.TextMatrix(0, 6) = "l * h"
tabla.TextMatrix(0, 7) = "Matriz"
tabla.TextMatrix(0, 8) = "Signo"

tabla.ColWidth(0) = 500
tabla.ColWidth(1) = 500
tabla.ColWidth(2) = 2000
tabla.ColWidth(3) = 900
tabla.ColWidth(4) = 900
tabla.ColWidth(5) = 900
tabla.ColWidth(6) = 900
tabla.ColWidth(7) = 700
tabla.ColWidth(8) = 700
If tabla.Rows > 1 Then tabla.FixedRows = 1
    Exit Sub
End If


' ***** Generador Por Concepto
If resultado = "POR CONCEPTO" Then
ImpresionPersonalizada.Enabled = True
        CapturaImagen.Enabled = True
        tabla.Cols = 9
        Set Rs = db.OpenRecordset("select * from valores where concepto = '" &
Arbol.SelectedItem & "' order by ID ASC")
            Do Until Rs.EOF
                                r = r + 1
tabla.Rows = r + 1
tabla.TextMatrix(r, 0) = r
                                tabla.TextMatrix(r, 1) = Rs("no")
                                tabla.TextMatrix(r, 2) = Rs("concepto")
tabla.TextMatrix(r, 3) = Rs("longitud")
                                tabla.TextMatrix(r, 4) = Rs("area")
tabla.TextMatrix(r, 5) = Rs("altura")
```

```
tabla.TextMatrix(r, 6) = Rs("lxh")
                                tabla.TextMatrix(r, 7) = Rs("compuesto")
                                tabla.TextMatrix(r, 8) = Rs("valor")
Rs.MoveNext
            Loop

tabla.TextMatrix(0, 0) = "ID"
tabla.TextMatrix(0, 1) = "No."
tabla.TextMatrix(0, 2) = "Concepto"
tabla.TextMatrix(0, 3) = "Long. (l)"
tabla.TextMatrix(0, 4) = "Área (A)"
tabla.TextMatrix(0, 5) = "Altura (h)"
tabla.TextMatrix(0, 6) = "l * h"
tabla.TextMatrix(0, 7) = "Matriz"
tabla.TextMatrix(0, 8) = "Signo"

tabla.ColWidth(0) = 500
tabla.ColWidth(1) = 500
tabla.ColWidth(2) = 2000
tabla.ColWidth(3) = 900
tabla.ColWidth(4) = 900
tabla.ColWidth(5) = 900
tabla.ColWidth(6) = 900
tabla.ColWidth(7) = 700
tabla.ColWidth(8) = 700
If tabla.Rows > 1 Then tabla.FixedRows = 1
    Exit Sub
End If


' ***** Resumen
If resultado = "GENERADOR" And Arbol.SelectedItem = "Resumen" Then

    Set Rs2 = db.OpenRecordset("SELECT DISTINCT valores.concepto FROM Valores")
    tabla.Cols = 6
        Do Until Rs2.EOF
acumulado1 = 0
            acumulado1 = 0
acumuladoA = 0
            acumulado2 = 0
            acumuladolxh = 0
            acumulado3 = 0
            acumuladoPZA = 0
            acumulado4 = 0

                        Set RsL = db.OpenRecordset("SELECT sum(longitud) as total
FROM valores where concepto = '" & Rs2("concepto") & "'")
Set RsA = db.OpenRecordset("SELECT sum(area) as total FROM valores where concepto
= '" & Rs2("concepto") & "'")
                        Set RsLxH = db.OpenRecordset("SELECT sum(lxh) as total
FROM valores where concepto = '" & Rs2("concepto") & "'")
                        Set RsPZA = db.OpenRecordset("SELECT COUNT(*) as total
FROM valores where concepto = '" & Rs2("concepto") & "'")
Tl = RsL("total")
                        TA = RsA("total")
TLxH = RsLxH("total")
                        Tpza = RsPZA("total")
```

```
                        tabla.Rows = tabla.Rows + 1
                        i = i + 1
                        tabla.TextMatrix(i, 0) = i
                        tabla.TextMatrix(i, 1) = Rs2("concepto")
                        tabla.TextMatrix(i, 2) = Round(Tl, pres)
                        tabla.TextMatrix(i, 3) = Round(TA, pres)
                        tabla.TextMatrix(i, 4) = Round(TLxH, pres)
                        tabla.TextMatrix(i, 5) = Round(Tpza, pres)

            Rs2.MoveNext
            Loop
        tabla.TextMatrix(0, 0) = "No."
tabla.TextMatrix(0, 1) = "Concepto"
        tabla.TextMatrix(0, 2) = "Longitud"
tabla.TextMatrix(0, 3) = "Área"
        tabla.TextMatrix(0, 4) = "L x h"
        tabla.TextMatrix(0, 5) = "Pza"

        tabla.ColWidth(0) = 500
        tabla.ColWidth(1) = 2500
tabla.ColWidth(2) = 1000
        tabla.ColWidth(3) = 1000
        tabla.ColWidth(4) = 1000
        tabla.ColWidth(5) = 1000

        If tabla.Rows > 1 Then tabla.FixedRows = 1
    db.Close
Exit Sub
End If

End Sub
```

Private Sub tabla_DblClick

```
pres = Tabla1.Presicion.ListIndex
Clipboard.Clear
c = tabla.Col
r = tabla.Row
varx = tabla.TextMatrix(r, c)

If IsNumeric(varx) Then
    Clipboard.SetText Round(varx, pres)
Else
    Clipboard.SetText varx
End If

End Sub
```

Private Sub ImpresionExcel_Click

```
If tabla.Rows = 1 Or tabla.Cols = 1 Or tabla.TextMatrix(0, 0) = "" Then Exit Sub
        c = tabla.Cols
        r = tabla.Rows
```

```
            Dim xlApp As Excel.Application
            Dim xlWB As Excel.Workbook
            Dim xlWS As Excel.Worksheet

            On Error Resume Next
            Set xlApp = GetObject(, "Excel.Application")
                If Err Then
                    Err.Clear
                    Set xlApp = CreateObject("Excel.Application")
                    xlApp.Workbooks.Add
                    AppActivate "Excel"
                    If Err Then
                        MsgBox Err.Description, vbCritical, "GRICC"
                        Exit Sub
                End If
            End If

    xlApp.Visible = True

    X = xlApp.ActiveWorkbook.Name
    Y = xlApp.ActiveSheet.Name
    Z = xlApp.Sheets.Count

For i = 1 To Z
    If UCase(xlApp.Sheets(i).Name) = Titulo Then GoTo 10
Next
    xlApp.Sheets.Add
    xlApp.ActiveSheet.Name = Titulo
10

   xlApp.Sheets(Titulo).Select

   xlApp.Range("a1:z10000").Clear

        For h = 1 To r
            For k = 1 To c
                xlApp.Cells(h, k) = tabla.TextMatrix(h - 1, k - 1)
            Next
        Next

        xlApp.Range(xlApp.Cells(1, 1), xlApp.Cells(1, c)).Interior.Color = RGB(0,
0, 0)
        xlApp.Range(xlApp.Cells(1, 1), xlApp.Cells(1, c)).Font.Color = RGB(255,
255, 255)
        xlApp.Range(xlApp.Cells(1, 1), xlApp.Cells(1, c)).HorizontalAlignment =
xlCenter
AppActivate "Cacho"
Exit Sub

End Sub


Private Sub ImpresionAutocad_Click

If tabla.Rows = 1 Or tabla.Cols = 1 Or tabla.TextMatrix(0, 0) = "" Then Exit Sub
        c = tabla.Cols
```

```
        r = tabla.Rows

Dim ptoI1(0 To 2) As Double
Dim coorX As Double
Dim coorY As Double
Dim AApp As Object
Dim ModelSpace As Object
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object
Dim DIF As Double
Dim d As Double, sep As Double
Dim ht As Double

ht = Tabla1.Altura
DIF = (ht * 1.1)

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace

AppActivate "Autocad"
ptoi = l.Utility.GetPoint(, vbCr & "Punto para insertar información")
coorX = Format(ptoi(0), "0")
coorY = Format(ptoi(1), "0")

sepY = ht * 1.1
sepX = ht * 2.2

For h = 0 To r - 1
    For k = 0 To c - 1
        ancho = 0
        For q = 1 To k
            ancho = ancho + (tabla.ColWidth(q - 1) * 0.01)
        Next

        ptoI1(0) = coorX + (sepX * k) + ancho
        ptoI1(1) = coorY - (sepY * h)

        Set El = MS.AddText(tabla.TextMatrix(h, k), ptoI1, ht)
ptoI1(0) = ptoI1(0) - (0.5 * sep)
        El.Color = Tabla1.co

Next

Next

Exit Sub
End Sub
```

Private Sub CapturaImagen_Click

```
On Error GoTo 200
Dim AApp As Object
Dim ModelSpace As Object
```

```
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object

On Error Resume Next
Set AApp = GetObject(, "AutoCAD.Application")
If Err Then
Err.Clear
Set AApp = CreateObject("AutoCAD.Application")
AApp.Visible = True
If Err Then
MsgBox Err.Description, vbCritical, "GRICC"
Exit Sub
End If
End If

Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility

Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

g01 = " GRICC "

Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

Dim xlApp As Excel.Application
Dim xlWB As Excel.Workbook
Dim xlWS As Excel.Worksheet
Set xlApp = GetObject(, "Excel.Application")
Dim marca As Double

Area.Hide
   l.SendCommand "_ai_selall" & vbCr
   l.SendCommand "_copyclip" & vbCr

renglon = 1
Do Until xlApp.Range("A" & renglon) = "REFERENCIA GRAFICA"
    renglon = renglon + 1
Loop
xlApp.Range("a" & renglon + 1).Activate

xlApp.ActiveSheet.PasteSpecial     Format:="Imagen     (metarchivo     mejorado)",
Link:=False, DisplayAsIcon:=False

xlApp.ActiveWindow.Zoom = 85

AApp.thisdrawing.SendCommand "ESC" & vbCr
Area.Show

GoTo 300
200 MsgBox "Excel NO Está Abierto", vbInformation, "GRICC"
300
```

```
End Sub
```

4.3. Funciones especiales.

Function cambia_propiedad

```
Dim w As Double

    If Tabla1.tabla.Row < Tabla1.tabla.RowSel Then
RI = Tabla1.tabla.Row: RF = Tabla1.tabla.RowSel
    Else
    RF = Tabla1.tabla.Row: RI = Tabla1.tabla.RowSel
End If


w = Label1

For i = RI To RF
Tabla1.tabla.TextMatrix(i, 6) = propiedad
Tabla1.tabla2.TextMatrix(i, 6) = propiedad
Next

cambio.Hide
Tabla1.Show
End Function
```

Function CrearLayer

```
Dim X As Double
Dim currLayer As AcadLayer
Dim newLayer As AcadLayer

Set acad = Nothing
On Error Resume Next
Set acad = GetObject(, "AutoCAD.Application")
If Err <> 0 Then
    Set acad = CreateObject("AutoCAD.Application")
    acad.Visible = True
    GoTo 1
End If

1
Set doc = acad.ActiveDocument
Set MSpace = doc.ModelSpace
Set l = acad.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

On Error GoTo 20
currLayer = UCase(Tabla1.empresa) & "-" & LCase(nombre)
GoTo 10
```

```
20
Set newLayer = AD.Layers.Add(UCase(Tabla1.empresa) & "-" & LCase(nombre))
newLayer.Color = clColor
AD.ActiveLayer = newLayer
10
End Function
```

Function CrearEstilodeTexto

```
Set acad = Nothing
On Error Resume Next
Set acad = GetObject(, "AutoCAD.Application")
If Err <> 0 Then
    Set acad = CreateObject("AutoCAD.Application")
    acad.Visible = True
    GoTo 1
End If
1
Set doc = acad.ActiveDocument
Set MSpace = doc.ModelSpace
Set l = acad.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AM = AD.ModelSpace

Set newtext = AD.TextStyles.Add(UCase(nombre))
newtext.Height = 0
newFontFile = "C:/Windows/Fonts/trebuc.ttf"
newtext.fontFile = newFontFile
AD.ActiveTextStyle = newtext

End Function
```

Function Dibujar

```
Dim AApp As Object
Dim ModelSpace As Object
Dim thisdrawing As Object
Dim El As Object
Dim MS As Object
Dim l As Object
Dim U As Object
Set MS = ActiveDocument.ModelSpace
Set AD = GetObject(, "autocad.application").ActiveDocument
Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument

Dim insertionPoint(0 To 2) As Double, Alignmentpoint(0 To 2) As Double

Dim points(0 To 9) As Double
Dim m(0 To 2) As Double
```

```
Dim empresa As Variant
Dim opcion As Variant
Dim varObject As AcadObject

' CREA NUEVAS LAYErs ---
Dim X As Double
Dim currLayer As AcadLayer
Dim newLayer As AcadLayer
Dim newtext As AcadTextStyle
Dim newFontFile

On Error GoTo 20
currLayer = "Cuantificar"
GoTo 10

20
Close #1

'crea layers 1
Set newtext = AD.TextStyles.Add("Cuantificar")
newtext.Height = 0
newFontFile = "C:/Windows/Fonts/trebuc.ttf"
newtext.fontFile = newFontFile
AD.ActiveTextStyle = newtext

Set newLayer = AD.Layers.Add(empresa & "Cuantificar")
newLayer.Color = acCyan
AD.ActiveLayer = newLayer
Set newLayer = AD.Layers.Add(empresa & "Cuant_NUM")
newLayer.Color = acYellow
Set newLayer = AD.Layers.Add(empresa & "Cuant_MCA")
newLayer.Color = acMagenta
'termina 1
10

Dim bytRojo As Byte
Dim bytVerde As Byte
Dim bytAzul As Byte
Dim strHex As String

strHex = Hex(co)
bytRojo = Val("&H" & Mid(strHex, 5, 2))
bytVerde = Val("&H" & Mid(strHex, 3, 2))
bytAzul = Val("&H" & Mid(strHex, 1, 2))

colorV = co

'MsgBox bytRojo & bytVerde & bytAzul

radio_min = Val(rm)
ht = rm * 0.8
m(0) = M1
m(1) = M2
m(2) = m3
opcion = mark
Pi = 3.141592
```

```
If Tabla1.ImpresionMarca(0).Value = 0 Then GoTo 350

'******* ***** ***** ***** ***** ***** ***** ***** ***** *****
' DIBUJOS ORDENADOS ALFABETICAMENTE
'******* ***** ***** ***** ***** ***** ***** ***** ***** *****

'Átomo
If opcion = 1 Then
Dim M13(0 To 2) As Double
M13(0) = 0.7071 * radio_min * 1.35
M13(1) = 0.7071 * radio_min * 1.35
Set El = MS.AddEllipse(m, M13, 0.55)
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

Dim M23(0 To 2) As Double
M23(0) = -0.7071 * radio_min * 1.35
M23(1) = 0.7071 * radio_min * 1.35
Set El = MS.AddEllipse(m, M23, 0.55)
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'círculo
If opcion = 2 Then
Set El = MS.AddCircle(m, radio_min * 1.05)
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'cuadro
If opcion = 3 Then
    points(0) = m(0) - radio_min: points(1) = m(1) - radio_min
    points(2) = m(0) + radio_min: points(3) = m(1) - radio_min
    points(4) = m(0) + radio_min: points(5) = m(1) + radio_min
    points(6) = m(0) - radio_min: points(7) = m(1) + radio_min
    points(8) = m(0) - radio_min: points(9) = m(1) - radio_min
Set El = MS.AddLightWeightPolyline(points)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'Eje
If opcion = 4 Then
Set El = MS.AddCircle(m, radio_min)
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

Dim e(0 To 3) As Double
e(0) = m(0): e(1) = m(1) + (0.8 * radio_min)
e(2) = m(0): e(3) = m(1) + (1.3 * radio_min)
Set El = MS.AddLightWeightPolyline(e)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
```

```
e(0) = m(0): e(1) = m(1) - (0.8 * radio_min)
e(2) = m(0): e(3) = m(1) - (1.3 * radio_min)
Set El = MS.AddLightWeightPolyline(e)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

e(0) = m(0) + (0.8 * radio_min): e(1) = m(1)
e(2) = m(0) + (1.3 * radio_min): e(3) = m(1)
Set El = MS.AddLightWeightPolyline(e)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

e(0) = m(0) - (0.8 * radio_min): e(1) = m(1)
e(2) = m(0) - (1.3 * radio_min): e(3) = m(1)
Set El = MS.AddLightWeightPolyline(e)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'Elipse
If opcion = 5 Then
Dim Maxis(0 To 2) As Double
Maxis(0) = radio_min * 1.3
Maxis(1) = 0

Set El = MS.AddEllipse(m, Maxis, 0.55)
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

End If

'Estrella
If opcion = 6 Then
d1 = (radio_min * 1.6)
d2 = (radio_min * 1.6) * 0.38

Dim pt(0 To 21) As Double
pt(0) = m(0): pt(1) = m(1) + d1
pt(2) = m(0) - (Sin(36 * (Pi / 180)) * d2)
pt(3) = m(1) + (Cos(36 * (Pi / 180)) * d2)
pt(4) = m(0) - (Cos(18 * (Pi / 180)) * d1)
pt(5) = m(1) + (Sin(18 * (Pi / 180)) * d1)
pt(6) = m(0) - (Cos(18 * (Pi / 180)) * d2)
pt(7) = m(1) - (Sin(18 * (Pi / 180)) * d2)
pt(8) = m(0) - (Sin(36 * (Pi / 180)) * d1)
pt(9) = m(1) - (Cos(36 * (Pi / 180)) * d1)
pt(10) = m(0): pt(11) = m(1) - d2
pt(12) = m(0) + (Sin(36 * (Pi / 180)) * d1)
pt(13) = m(1) - (Cos(36 * (Pi / 180)) * d1)
pt(14) = m(0) + (Cos(18 * (Pi / 180)) * d2)
pt(15) = m(1) - (Sin(18 * (Pi / 180)) * d2)
pt(16) = m(0) + (Cos(18 * (Pi / 180)) * d1)
pt(17) = m(1) + (Sin(18 * (Pi / 180)) * d1)
pt(18) = m(0) + (Sin(36 * (Pi / 180)) * d2)
```

```
pt(19) = m(1) + (Cos(36 * (Pi / 180)) * d2)
pt(20) = pt(0): pt(21) = pt(1)

Set El = MS.AddLightWeightPolyline(pt)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'Flecha
If opcion = 7 Then
d1 = radio_min * 0.55

Dim p6(0 To 5) As Double
p6(0) = m(0) - d1: p6(1) = m(1) + d1
p6(2) = m(0): p6(3) = m(1) + (d1 * 1.45)
p6(4) = m(0) + d1: p6(5) = p6(1)
Set El = MS.AddLightWeightPolyline(p6)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

p6(0) = m(0) - d1: p6(1) = m(1) - d1
p6(2) = m(0): p6(3) = m(1) - (d1 * 1.45)
p6(4) = m(0) + d1: p6(5) = p6(1)
Set El = MS.AddLightWeightPolyline(p6)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'Paralelas
If opcion = 8 Then
d1 = rm

Dim p7(0 To 3) As Double
p7(0) = m(0) - (Sin(45 * (Pi / 180)) * d1): p7(1) = m(1) + (d1 * 0.8)
p7(2) = m(0) + (Sin(45 * (Pi / 180)) * d1): p7(3) = p7(1)
Set El = MS.AddLightWeightPolyline(p7)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"

d1 = rm
Dim p71(0 To 3) As Double
p71(0) = m(0) - (Sin(45 * (Pi / 180)) * d1): p71(1) = m(1) - (d1 * 0.6)
p71(2) = m(0) + (Sin(45 * (Pi / 180)) * d1): p71(3) = p71(1)
Set El = MS.AddLightWeightPolyline(p71)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'rombo
If opcion = 9 Then
    points(0) = m(0): points(1) = m(1) - (radio_min * 1.2)
    points(2) = m(0) + (radio_min * 1.2): points(3) = m(1)
```

```
    points(4) = m(0): points(5) = m(1) + (radio_min * 1.2)
    points(6) = m(0) - (radio_min * 1.2): points(7) = m(1)
    points(8) = m(0): points(9) = m(1) - (radio_min * 1.2)
Set El = MS.AddLightWeightPolyline(points)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

'Triángulo
If opcion = 10 Then
d1 = (radio_min * 1.5)
Dim ptr(0 To 7) As Double
ptr(0) = m(0): ptr(1) = m(1) + d1
ptr(2) = m(0) - (Cos(30 * (Pi / 180)) * d1)
ptr(3) = m(1) - (Sin(30 * (Pi / 180)) * d1)
ptr(4) = m(0) + (Cos(30 * (Pi / 180)) * d1)
ptr(5) = ptr(3)
ptr(6) = ptr(0): ptr(7) = ptr(1)

Set El = MS.AddLightWeightPolyline(ptr)
El.ConstantWidth = ht * 0.14
El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
End If

350
Dim textom
Dim TextoCte

    If PREFIJO = "" Or PREFIJO = Empty Then textom = Tabla1.Etiqueta
    If PREFIJO <> "" Or PREFIJO <> Empty Then textom = PREFIJO & "-" &
Tabla1.Etiqueta

r = Tabla1.tabla.Rows - 1
If Tabla1.ImpresionMarca(0).Value = 0 Then cteConc = M2 Else cteConc = M2 + (2 *
ht)

If Tabla1.ImpresionMarca(4).Value = 1 Then
    ' Concepto
    insertionPoint(0) = M1: insertionPoint(1) = cteConc: insertionPoint(2) = m(2)
    Alignmentpoint(0) = M1: Alignmentpoint(1) = cteConc: Alignmentpoint(2) = m(2)

TextoCte = Tabla1.Conceptos
    'If Tabla1.MenuPrincipal.Tab = 3 Then TextoCte = Tabla1.DuctoName Else
TextoCte = Tabla1.Conceptos
Set El = MS.AddMText(insertionPoint, ht * 0.5, TextoCte)

    El.Height = ht * 0.7
    El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
El.BackgroundFill = True
    El.Width = ht * 2
    Etiqueta = Etiqueta + 1
    El.AttachmentPoint = acAttachmentPointMiddleCenter
    El.insertionPoint = insertionPoint
End If
```

```
FacX = 1.2
cteX = M2

If Tabla1.ImpresionMarca(0).Value = 1 Then
    ' Elemento
    insertionPoint(0) = M1: insertionPoint(1) = cteX: insertionPoint(2) = m(2)
    Alignmentpoint(0) = M1: Alignmentpoint(1) = cteX: Alignmentpoint(2) = m(2)

    TextoCte = textom
    'If Tabla1.MenuPrincipal.Tab = 3 Then TextoCte = xlApp.Range("a" & rx) Else
TextoCte = textom
    Set El = MS.AddMText(insertionPoint, ht * 0.5, TextoCte)

    El.Height = ht * 0.7
    El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
El.BackgroundFill = True
El.Width = ht * 2
    Etiqueta = Etiqueta + 1
El.AttachmentPoint = acAttachmentPointMiddleCenter
    El.insertionPoint = insertionPoint
End If

cteX = cteX - (0.4 * ht)
If Tabla1.ImpresionMarca(1).Value = 1 Then
    ' Longitud
    cteX = cteX - (FacX * ht)
    insertionPoint(0) = M1: insertionPoint(1) = cteX: insertionPoint(2) = m(2)
    Alignmentpoint(0) = M1: Alignmentpoint(1) = cteX: Alignmentpoint(2) = m(2)

TextoCte = "L:" & Tabla1.tabla.TextMatrix(r, 1)
'If Tabla1.MenuPrincipal.Tab = 3 Then TextoCte = "L:" & xlApp.Range("c" & rx)
Else TextoCte = "L:" & Tabla1.tabla.TextMatrix(r, 1)
    Set El = MS.AddMText(insertionPoint, ht * 0.5, TextoCte)

    El.Height = ht * 0.7
    El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
El.BackgroundFill = True
    El.Width = ht * 2
    Etiqueta = Etiqueta + 1
    El.AttachmentPoint = acAttachmentPointMiddleCenter
    El.insertionPoint = insertionPoint
End If

If Tabla1.ImpresionMarca(2).Value = 1 Then
' Área
    cteX = cteX - (FacX * ht)
insertionPoint(0) = M1: insertionPoint(1) = cteX: insertionPoint(2) = m(2)
    Alignmentpoint(0) = M1: Alignmentpoint(1) = cteX: Alignmentpoint(2) = m(2)

    If Tabla1.MenuPrincipal.Tab = 3 Or Tabla1.MenuPrincipal.Tab = 4 Then TextoCte
=   "W:"   &   Tabla1.tabla.TextMatrix(r,   3)   Else   TextoCte   =   "A:"   &
Tabla1.tabla.TextMatrix(r, 2)
    Set El = MS.AddMText(insertionPoint, ht * 0.5, TextoCte)
```

```
    El.Height = ht * 0.7
    El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
El.BackgroundFill = True
    El.Width = ht * 2
    Etiqueta = Etiqueta + 1
    El.AttachmentPoint = acAttachmentPointMiddleCenter
    El.insertionPoint = insertionPoint
End If

If Tabla1.ImpresionMarca(3).Value = 1 Then
    ' L x h
    cteX = cteX - (FacX * ht)
    insertionPoint(0) = M1: insertionPoint(1) = cteX: insertionPoint(2) = m(2)
    Alignmentpoint(0) = M1: Alignmentpoint(1) = cteX: Alignmentpoint(2) = m(2)

    If Tabla1.MenuPrincipal.Tab = 3 Or Tabla1.MenuPrincipal.Tab = 4 Then TextoCte
=   "Wt:"   &   Tabla1.tabla.TextMatrix(r,   4)   Else   TextoCte   =   "Lxh:"   &
Tabla1.tabla.TextMatrix(r, 4)
    Set El = MS.AddMText(insertionPoint, ht * 0.5, TextoCte)

    El.Height = ht * 0.7
    El.Color = colorV
El.Layer = empresa & "Cuant_NUM"
El.BackgroundFill = True
    El.Width = ht * 2
    Etiqueta = Etiqueta + 1
    El.AttachmentPoint = acAttachmentPointMiddleCenter
    El.insertionPoint = insertionPoint
End If

El.Update

End Function
```

Function Seleccion

```
Set acad = Nothing
On Error Resume Next
Set acad = GetObject(, "AutoCAD.Application")
If Err <> 0 Then
Set acad = CreateObject("AutoCAD.Application")
acad.Visible = True

End If

Set doc = acad.ActiveDocument
Set MSpace = doc.ModelSpace
Set l = acad.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
g01 = " GRICC "
Set AD = GetObject(, "autocad.application").ActiveDocument
X = Mid(l.Name, 1, Len(l.Name) - 4)
RunTimes = GetSetting(X, "Counts", "NumVeces", 0) + 1
```

```
SaveSetting X, "Counts", "NumVeces", RunTimes
Contador = RunTimes

Seleccion = X & Contador

End Function
```

Function SinEspacios

```
'--------- elimina los espacios en la lista de propiedades
Dim Z As Variant, Y As Variant
 Z = ""
For i = 1 To Len(X)
    Y = Mid(X, i, 1)
    If Asc(Y) <> 32 Then Z = Z + Y
Next

SinEspacios = Z

End Function
```

4.4. Base de datos, creación y administración.

Function CrearTabla

```
X = Dir(nombre)

Dim db As Database
Dim Fd As Field
Dim Tb As New TableDef
Dim Idx As New Index, Idy As New Index, Idz As New Index
Dim i As Integer

sbase = LCase(nombre)

If X = "" Then
    Set db = CreateDatabase(sbase, dbLangSpanish, dbVersion30)

    ' Tabla Compuesto
        Set Tb = db.CreateTableDef("Compuesto")
        Set Fd = Tb.CreateField("ID", dbLong)
        Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
        Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Clave", dbText): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Codigo", dbText): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Concepto", dbText): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Metodo", dbText): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Cantidad", dbDouble): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Factor1", dbDouble): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Factor2", dbDouble): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Partida", dbText): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Grupo", dbText): Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Unidad", dbText): Tb.Fields.Append Fd
```

```
        Idy.Name = "PrimaryKey"
        Idy.Unique = True
        Idy.Primary = True
        Idy.Fields = "ID"
        Tb.Indexes.Append Idy
        db.TableDefs.Append Tb

    ' Tabla Valores
        Set Tb = db.CreateTableDef("Valores")
        Set Fd = Tb.CreateField("ID", dbLong)
        Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
        Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("No", dbText):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Longitud", dbDouble):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Area", dbDouble):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Altura", dbDouble):   Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Lxh", dbDouble):    Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Valor", dbText):    Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Concepto", dbText):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Compuesto", dbText):    Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Color", dbDouble):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Marca", dbText):    Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Objeto", dbText):   Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("IdObj", dbText):   Tb.Fields.Append Fd
        Idx.Name = "PrimaryKey"
        Idx.Unique = True
        Idx.Primary = True
        Idx.Fields = "ID"
        Tb.Indexes.Append Idx
        db.TableDefs.Append Tb

    ' Tabla Conceptos
        Set Tb = db.CreateTableDef("Conceptos")
        Set Fd = Tb.CreateField("ID", dbLong)
        Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
        Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Codigo", dbText):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Concepto", dbText):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Compuesto", dbText):    Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Clave", dbText):    Tb.Fields.Append Fd
        Idz.Name = "PrimaryKey"
        Idz.Unique = True
        Idz.Primary = True
        Idz.Fields = "ID"
        Tb.Indexes.Append Idz
        db.TableDefs.Append Tb

    db.Close
End If

End Function
```

Function CrearTablaConceptos

```
nombre = "C:\Cuantificador2012\Tablas\conceptos.mdb"
X = Dir(nombre)
```

```
Dim db As Database
Dim Fd As Field
Dim Tb As New TableDef
Dim Idx As New Index, Idy As New Index, Idz As New Index
Dim i As Integer

sbase = LCase(nombre)

If X = "" Then
    Set db = CreateDatabase(sbase, dbLangSpanish, dbVersion30)

' Tabla Partidas
        Set Tb = db.CreateTableDef("Partidas")
Set Fd = Tb.CreateField("ID", dbLong)
        Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
        Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Partida", dbText): Tb.Fields.Append Fd
        Idy.Name = "PrimaryKey"
        Idy.Unique = True
        Idy.Primary = True
        Idy.Fields = "ID"
        Tb.Indexes.Append Idy
        db.TableDefs.Append Tb

' Tabla GRupos
        Set Tb = db.CreateTableDef("Grupos")
Set Fd = Tb.CreateField("ID", dbLong)
        Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
        Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Partida", dbText):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Grupo", dbDouble):  Tb.Fields.Append Fd
        Idx.Name = "PrimaryKey"
        Idx.Unique = True
        Idx.Primary = True
        Idx.Fields = "ID"
        Tb.Indexes.Append Idx
        db.TableDefs.Append Tb

    ' Tabla Conceptos
        Set Tb = db.CreateTableDef("Conceptos")
        Set Fd = Tb.CreateField("ID", dbLong)
        Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
        Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Partida", dbText):  Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("Grupo", dbText):  Tb.Fields.Append Fd
            Set  Fd  =  Tb.CreateField("Descripcion", dbText):     Tb.Fields.Append
Fd
            Set Fd = Tb.CreateField("unidad", dbText):    Tb.Fields.Append Fd
            Set Fd = Tb.CreateField("codigo", dbText):    Tb.Fields.Append Fd
        Idz.Name = "PrimaryKey"
        Idz.Unique = True
        Idz.Primary = True
        Idz.Fields = "ID"
        Tb.Indexes.Append Idz
        db.TableDefs.Append Tb
    db.Close
```

```
End If

End Function


Function CrearTablaOriginal
    Dim db As Database
    Dim Fd As Field
    Dim Tb As New TableDef
    Dim Idx As New Index, Idy As New Index
Dim i As Integer

    'Crear base de datos, idioma español y para la versión 2.0 del Jet de Access

'=============================================================================
    'Si vas a adaptar este programa para VB3, usa dbVersion11 en lugar de
dbVersion20
'=============================================================================
    sbase = LCase(nombre)
    Set db = CreateDatabase(sbase, dbLangSpanish, dbVersion30)
    ' Set Db = CreateDatabase(sbase, dbLangSpanish & ";pwd=win", dbVersion30)

    Set Tb = db.CreateTableDef("Valores")
'Vamos a crear cada uno de los campos
Set Fd = Tb.CreateField("ID", dbLong)
'Ahora vamos a asignar las propiedades de contador, etc.
Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
Tb.Fields.Append Fd
    'El resto de los campos
Set Fd = Tb.CreateField("No", dbText):            Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Longitud", dbDouble):   Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Area", dbDouble):          Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Altura", dbDouble):        Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Lxh", dbDouble):           Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Valor", dbText):            Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Concepto", dbText):    Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Compuesto", dbText):         Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Color", dbDouble):    Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Marca", dbText):          Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("Objeto", dbText):      Tb.Fields.Append Fd
    Set Fd = Tb.CreateField("IdObj", dbText):       Tb.Fields.Append Fd

'Creamos un índice con el ID
Idx.Name = "PrimaryKey"
    Idx.Unique = True
    Idx.Primary = True
    Idx.Fields = "ID"
    Tb.Indexes.Append Idx
    db.TableDefs.Append Tb

    Set Tb = db.CreateTableDef("Conceptos")
    Set Fd = Tb.CreateField("ID", dbLong)
    Fd.Attributes = dbAutoIncrField Or dbUpdatableField Or dbFixedField
    Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Clave", dbText): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Codigo", dbText): Tb.Fields.Append Fd
```

```
        Set Fd = Tb.CreateField("Concepto", dbText): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Metodo", dbText): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Cantidad", dbDouble): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Factor1", dbDouble): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Factor2", dbDouble): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Partida", dbText): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Grupo", dbText): Tb.Fields.Append Fd
        Set Fd = Tb.CreateField("Unidad", dbText): Tb.Fields.Append Fd
    Idy.Name = "PrimaryKey"
    Idy.Unique = True
    Idy.Primary = True
    Idy.Fields = "ID"
    Tb.Indexes.Append Idy
    db.TableDefs.Append Tb

'Cerramos la base
    db.Close

    'MsgBox "Nueva base de datos " & sbase & " creada.", vbInformation

End Function
```

Function FiltrarTabla

```
If nombre = "" Then GoTo 100

    Dim db As Database
    Dim i As Integer

sbase = LCase(nombre)
Set db = OpenDatabase(sbase)

    ' Abre Tabla Valores

If Tabla1.Conceptos.ListCount > 0 And Tabla1.Visualizar.Value = True Then
        Set Rs = db.OpenRecordset("SELECT * FROM Valores WHERE Concepto = '" &
Tabla1.Conceptos & "'")
    Else
        Set Rs = db.OpenRecordset("Valores", dbOpenDynaset)
    End If

                        'If Tabla1.Visualizar.Value = True Then
                        '     Set Rs2 = db.OpenRecordset("SELECT Max(valores.No)
AS ElMax FROM Valores WHERE Concepto = '" & Tabla1.Conceptos & "' ORDER BY No ASC
")
                        '    If Rs2("ElMax") = Null Then Tabla1.Etiqueta = 0 Else
Tabla1.Etiqueta = Rs2("ElMax")
                        'Else
                        '     Set Rs2 = db.OpenRecordset("SELECT Max(valores.No)
AS ElMax FROM Valores ORDER BY No ASC")
                        '    If Rs2("ElMax") = Null Then Tabla1.Etiqueta = 0 Else
Tabla1.Etiqueta = Rs2("ElMax")
                        'End If
```

```
Tabla1.tabla.Rows = 1

        r = 0
        Do Until Rs.EOF
r = r + 1
            Tabla1.tabla.Rows = r + 1
            Tabla1.tabla.TextMatrix(r, 0) = Rs.Fields("No")
Tabla1.tabla.TextMatrix(r, 1) = Rs.Fields("Longitud")
            Tabla1.tabla.TextMatrix(r, 2) = Rs.Fields("Area")
Tabla1.tabla.TextMatrix(r, 3) = Rs.Fields("Altura")
            Tabla1.tabla.TextMatrix(r, 4) = Rs.Fields("Lxh")
            Tabla1.tabla.TextMatrix(r, 5) = Rs.Fields("Valor")
            Tabla1.tabla.TextMatrix(r, 6) = Rs.Fields("concepto")
            Tabla1.tabla.TextMatrix(r, 7) = Rs.Fields("Compuesto")
            Tabla1.tabla.TextMatrix(r, 8) = Rs.Fields("Color")
            Tabla1.tabla.TextMatrix(r, 9) = Rs.Fields("Marca")
            Tabla1.tabla.TextMatrix(r, 10) = Rs.Fields("Objeto")
Tabla1.tabla.TextMatrix(r, 11) = Rs.Fields("IdObj")

            Rs.MoveNext
        Loop
    db.Close
100

If Tabla1.tabla.Rows > 1 Then Tabla1.tabla.FixedRows = 1

End Function
```

Function GuardarTabla

```
    If Dir(nombre, vbArchive) <> "" Then Kill (nombre)
CrearTabla (nombre)

    Dim db As Database
Dim Fd As Field
    Dim Tb As New TableDef
    Dim i As Integer

sbase = LCase(nombre)
Set db = OpenDatabase(sbase)

        ' Guarda Tabla Valores
Set Rs = db.OpenRecordset("Valores", dbOpenDynaset)

        For i = 1 To Tabla1.tabla2.Rows - 1
        If Rs.EOF And Rs.BOF Then Rs.AddNew Else Rs.MoveLast: Rs.AddNew
r = i

            Rs.Fields("No") = Tabla1.tabla2.TextMatrix(r, 0)
            Rs.Fields("Longitud") = Tabla1.tabla2.TextMatrix(r, 1)
            Rs.Fields("Area") = Tabla1.tabla2.TextMatrix(r, 2)
            Rs.Fields("Altura") = Tabla1.tabla2.TextMatrix(r, 3)
            Rs.Fields("Lxh") = Tabla1.tabla2.TextMatrix(r, 4)
            Rs.Fields("Valor") = Tabla1.tabla2.TextMatrix(r, 5)
            Rs.Fields("Concepto") = Tabla1.tabla2.TextMatrix(r, 6)
```

```
            Rs.Fields("Compuesto") = Tabla1.tabla2.TextMatrix(r, 7)
            Rs.Fields("Color") = Tabla1.tabla2.TextMatrix(r, 8)
            Rs.Fields("Marca") = Tabla1.tabla2.TextMatrix(r, 9)
            Rs.Fields("Objeto") = Tabla1.tabla2.TextMatrix(r, 10)
            Rs.Fields("IdObj") = Tabla1.tabla2.TextMatrix(r, 11)
            Rs.Update

        Next

100
    'Guarda la Tabla Conceptos
n = Tabla1.Conceptos.ListCount
    If n = 0 Then GoTo 200
        Set Rs = db.OpenRecordset("Conceptos", dbOpenDynaset)
                concepto1 = Tabla1.Conceptos.Text
        For i = 1 To n
        If Rs.EOF And Rs.BOF Then Rs.AddNew Else Rs.MoveLast: Rs.AddNew
Tabla1.Conceptos.ListIndex = i - 1
            Rs.Fields("coDIGO") = 0
            Rs.Fields("concepto") = Tabla1.Conceptos
Rs.Fields("coMPUESto") = 0
Rs.Fields("cLAVE") = 0
            Rs.Update
Next
Tabla1.Conceptos.Text = concepto1

200

    ' Guarda la tabla Compuesto
Set Rs = db.OpenRecordset("Compuesto", dbOpenDynaset)
n = Tabla1.tablaC.Rows
        For i = 1 To n - 1
            If Rs.EOF And Rs.BOF Then Rs.AddNew Else Rs.MoveLast: Rs.AddNew
r = i
                Rs.Fields("CLAVE") = Tabla1.tablaC.TextMatrix(r, 0)
                Rs.Fields("CODIGO") = Tabla1.tablaC.TextMatrix(r, 1)
                Rs.Fields("CONCEPTO") = Tabla1.tablaC.TextMatrix(r, 2)
                Rs.Fields("METODO") = Tabla1.tablaC.TextMatrix(r, 3)
                Rs.Fields("CANTIDAD") = Tabla1.tablaC.TextMatrix(r, 4)
                Rs.Fields("FACTOR1") = Tabla1.tablaC.TextMatrix(r, 5)
                Rs.Fields("FACTOR2") = Tabla1.tablaC.TextMatrix(r, 6)
                Rs.Fields("PARTIDA") = Tabla1.tablaC.TextMatrix(r, 7)
                Rs.Fields("GRUPO") = Tabla1.tablaC.TextMatrix(r, 8)
Rs.Update
        Next
300
    db.Close

End Function
```

Function GuardarOpciones

```
    Dim db As Database
    Dim Fd As Field
    Dim Tb As New TableDef
```

```
    Dim i As Integer

sbase = LCase("C:\Cuantificador2012\System\Opciones.mdb")
Set db = OpenDatabase(sbase)

        Set Rs = db.OpenRecordset("Opciones", dbOpenDynaset)

            Rs.MoveFirst
            Rs.Edit
                Rs.Fields("marca") = Tabla1.Ma
Rs.Fields("color") = Tabla1.co
                Rs.Fields("tamaño") = Tabla1.TamanoMarca
                Rs.Fields("Altura") = Tabla1.Altura
                Rs.Fields("precision") = Tabla1.Presicion.ListIndex
                Rs.Fields("etiqueta") = Tabla1.Etiqueta
            Rs.Update

Set Rs = db.OpenRecordset("Impresion", dbOpenDynaset)

            Rs.MoveFirst
            Rs.Edit
                Rs.Fields("no") = Tabla1.ImpresionMarca(0).Value
                Rs.Fields("longitud") = Tabla1.ImpresionMarca(1).Value
                Rs.Fields("area") = Tabla1.ImpresionMarca(2).Value
Rs.Fields("lxh") = Tabla1.ImpresionMarca(3).Value
                Rs.Fields("marca") = Tabla1.ImpresionMarca(4).Value
Rs.Update

        Set Rs = db.OpenRecordset("Varios", dbOpenDynaset)

            Rs.MoveFirst
            Rs.Edit
                Rs.Fields("anterior") = Tabla1.ArchAnt
            Rs.Update

    db.Close

End Function
```

Function CargarTabla

```
If nombre = "" Then GoTo 100

    Dim db As Database
    Dim Fd As Field
    Dim Tb As New TableDef
    Dim i As Integer

sbase = LCase(nombre)
    Set db = OpenDatabase(sbase)

    'Abre la Tabla Conceptos

Set Rs = db.OpenRecordset("SELECT DISTINCT valores.concepto FROM valores")
    Tabla1.Conceptos.Clear
```

```
        If Rs.EOF And Rs.BOF Then GoTo 200
        Rs.MoveFirst

        Do Until Rs.EOF
            Tabla1.Conceptos.AddItem Rs.Fields("concepto")
            Rs.MoveNext
        Loop
200

Set Rs = db.OpenRecordset("SELECT DISTINCT compuesto.clave FROM compuesto")
    Do Until Rs.EOF
        cteClave = UCase(SinEspacios(Rs.Fields("clave")))
        If Trim(cteClave) = "" Then GoTo 32

            For i = 0 To Tabla1.Conceptos.ListCount - 1
Tabla1.Conceptos.ListIndex = i
                If Tabla1.Conceptos.Text = cteClave Then
                    Tabla1.Conceptos = cteClave
GoTo 32
                End If
            Next

        Tabla1.Conceptos.AddItem cteClave
32
        Rs.MoveNext
    Loop


If Tabla1.Conceptos.ListCount > 0 Then Tabla1.Conceptos.ListIndex = 0


' Abre Tabla Valores
    'MsgBox tabla1.Conceptos.ListCount & vbCr & tabla1.Conceptos & vbCr &
tabla1.Visualizar.Value

If Tabla1.Conceptos.ListCount > 0 And Tabla1.Visualizar.Value = True Then
        Set Rs = db.OpenRecordset("SELECT * FROM Valores WHERE Concepto = '" &
Tabla1.Conceptos & "'")
    Else
        Set Rs = db.OpenRecordset("Valores", dbOpenDynaset)
    End If

        If Rs.EOF And Rs.BOF Then GoTo 100
        Rs.MoveFirst

        r = 0
        Do Until Rs.EOF
r = r + 1
            Tabla1.tabla.Rows = r + 1
            Tabla1.tabla.TextMatrix(r, 0) = Rs.Fields("No")
Tabla1.tabla.TextMatrix(r, 1) = Rs.Fields("Longitud")
            Tabla1.tabla.TextMatrix(r, 2) = Rs.Fields("Area")
Tabla1.tabla.TextMatrix(r, 3) = Rs.Fields("Altura")
            Tabla1.tabla.TextMatrix(r, 4) = Rs.Fields("Lxh")
            Tabla1.tabla.TextMatrix(r, 5) = Rs.Fields("Valor")
            Tabla1.tabla.TextMatrix(r, 6) = Rs.Fields("concepto")
            Tabla1.tabla.TextMatrix(r, 7) = Rs.Fields("Compuesto")
```

```
            Tabla1.tabla.TextMatrix(r, 8) = Rs.Fields("Color")
            Tabla1.tabla.TextMatrix(r, 9) = Rs.Fields("Marca")
            Tabla1.tabla.TextMatrix(r, 10) = Rs.Fields("Objeto")
            Tabla1.tabla.TextMatrix(r, 11) = Rs.Fields("IdObj")

            Tabla1.Etiqueta = Tabla1.tabla.TextMatrix(r, 0)

Rs.MoveNext
        Loop
100


 Set Rs = db.OpenRecordset("Compuesto", dbOpenDynaset)
        If Rs.EOF And Rs.BOF Then GoTo 300
        Rs.MoveFirst

        r = 0
        Do Until Rs.EOF
r = r + 1
            Tabla1.tablaC.Rows = r + 1
            Tabla1.tablaC.TextMatrix(r, 0) = Rs.Fields("CLAVE")
            Tabla1.tablaC.TextMatrix(r, 1) = Rs.Fields("CODIGO")
            Tabla1.tablaC.TextMatrix(r, 2) = Rs.Fields("CONCEPTO")
            Tabla1.tablaC.TextMatrix(r, 3) = Rs.Fields("METODO")
            Tabla1.tablaC.TextMatrix(r, 4) = Rs.Fields("CANTIDAD")
            Tabla1.tablaC.TextMatrix(r, 5) = Rs.Fields("FACTOR1")
            Tabla1.tablaC.TextMatrix(r, 6) = Rs.Fields("FACTOR2")
            Tabla1.tablaC.TextMatrix(r, 7) = Rs.Fields("PARTIDA")
            Tabla1.tablaC.TextMatrix(r, 8) = Rs.Fields("GRUPO")
Rs.MoveNext
        Loop
300

    db.Close

End Function
```

4.5. Conexión a Autocad

Para realizar la conexión a Autocad desde Visual Basic, en la compilación necesitamos hacer referencia a la librería que se encuentre instalada en nuestra PC, basta con ejecutar "Referencias" del menú "Proyecto"; nos muestra una ventana con un listado de DLL instaladas en nuestro equipo, para mi caso, selecciono "**Autocad 2011 Type Library**", así ya podemos utilizar los comandos de esta librería y controlar Autocad desde nuestra aplicación.

A continuación se muestra la rutina para conectarse con Autocad en tiempo de ejecución y cómo realizar la redacción para ejecutar los comandos.

Para cualquier tipo de error en la secuencia de comandos, nos finaliza la rutina, lo más común es que no contemos con Autocad en nuestro equipo o que no esté iniciado dicho programa.

```
On Error Resume Next
```

Llamámos al programa.

```
Set AApp = GetObject(, "AutoCAD.Application")
```

Se evalua el resultado de la petición que se hace.

```
If Err Then
```

Si se muestra un error, ejecuta Autocad y en modo visible en pantalla.

```
Err.Clear
Set AApp = CreateObject("AutoCAD.Application")
AApp.Visible = True
```

Se muestra el tipo de error que ocurrió en la ejecución.

```
If Err Then
                MsgBox Err.Description, vbCritical, "GRICC"
Exit Sub
End If
```

Se asignan los procedimientos a cada variable a utilizar.

```
Set AApp = GetObject(, "Autocad.Application")
Set l = AApp.ActiveDocument
Set MS = l.ModelSpace
Set U = AApp.thisdrawing.Utility
Set AD = GetObject(, "autocad.application").ActiveDocument
```

Como resultado, tenemos**AM**para iniciar y hacer la referencia a los comandos de Autocad y trabajar en la pestaña de "Model".

```
Set AM = AD.ModelSpace
```

4.6. Conexión a Excel

Para realizar la conexión a Excel desde Visual Basic, en la compilación necesitamos hacer referencia a la librería que se encuentre instalada en nuestra PC, basta con ejecutar "Referencias" del menú "Proyecto"; nos muestra una ventana con un listado de DLL instaladas en nuestro equipo, para mi caso, selecciono "**MicrosoftExcel 12.0 Type Library**",

así ya podemos utilizar los comandos de esta librería y controlar Excel desde nuestra aplicación.

A continuación se muestra la rutina para conectarse con Excel en tiempo de ejecución y cómo realizar la redacción para ejecutar los comandos.

Se declaran las variables a utilizar como prefijo en nuestros comandos, tal y como los utilizamos en los Macros de Excel

Donde **XlApp** es el prefijo a utilizar comúnmente en el uso de los comandos.

```
Dim xlApp As Excel.Application
Dim xlWB As Excel.Workbook
Dim xlWS As Excel.Worksheet
```

Si se presenta algún error termina la rutina sin finalizar el programa, el error típico es que no esté abierta la aplicación.

```
On Error Resume Next
```

Se llama a la aplicación, abierta por el usuario.

```
Set xlApp = GetObject(, "Excel.Application")
```

Si se presenta el error porque no está instalado Excel en el sistema, muestra un mensaje con la descripción de dicho error termina la rutina sin conectarse.

```
If Err Then
    MsgBox Err.Description, vbCritical, "GRICC || Excel"
Exit Sub
End If
```

4.7. Ejemplos de aplicación

Para mostrar la aplicación del programa realizado, se muestran 3 ejemplos en diferentes áreas dentro de la construcción.Partiendo de los planos de proyecto en formato de Autocad, definiendo detalles propios del concepto, unidad de medición y alcances generales. Para los ejemplos se utilizará parte del proyecto de una clínica.

Ejemplo de aplicación 1.

Del plano estructural, se necesita cuantificar las trabes de acero por metro lineal, y se multiplica por el peso nominal del perfil (kg/ml) para determinar el peso del acero.

Le damos un formato rápido al plano, ocultando las capas no necesarias y asignando a todos los elementos color gris, para que quede como indica la figura .

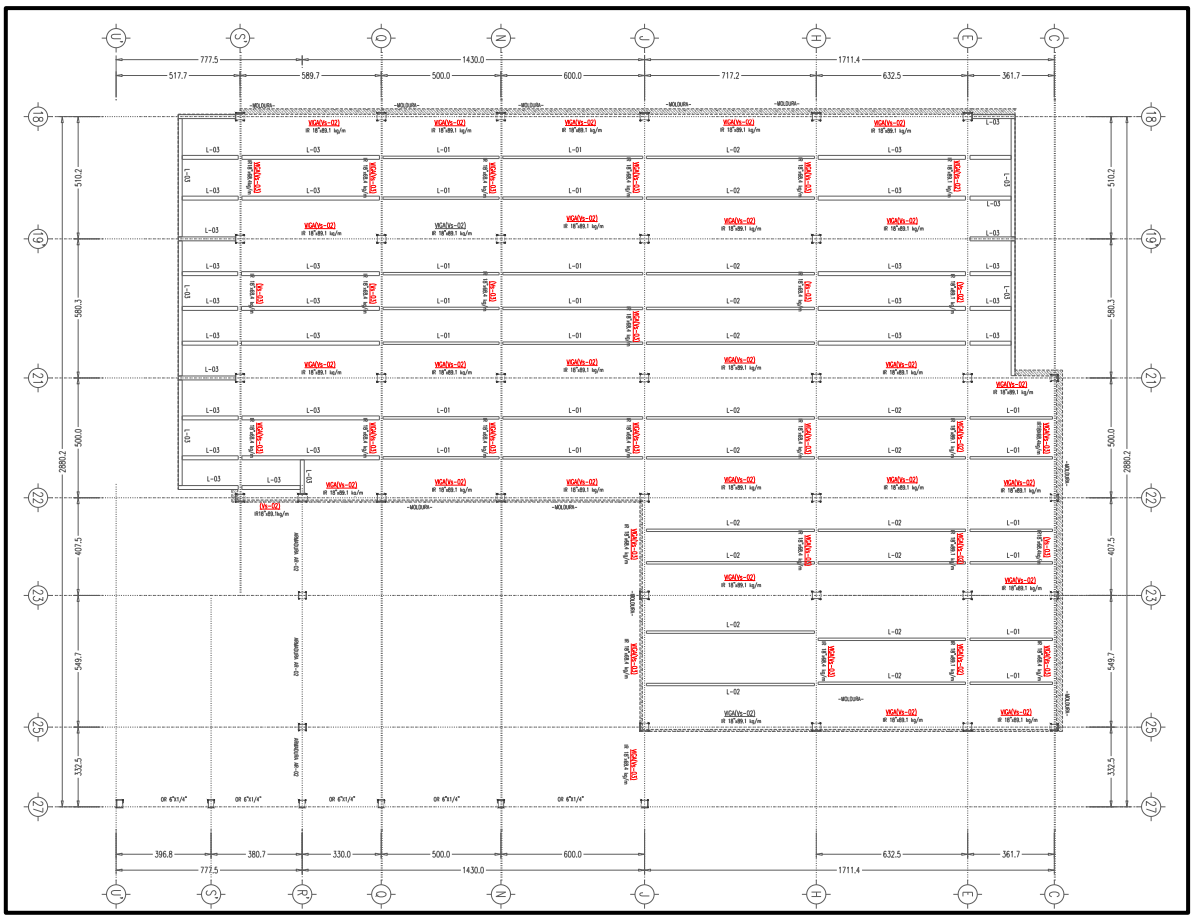


Figura 4.17, Plano de estructura metálica

Para nuestro caso, tenemos las siguientes secciones de trabes

Viga (Vs-02), IR 18"x89.10 kg/m

Viga (Vs-03), IR 18"x68.40 kg/m

Larguero (L-01), IR 12"x23.9 kg/m

Larguero (L-02), IR 12"x32.8 kg/m

Larguero (L-03), IR 12"x38.7 kg/m

Creamos una carpeta en una ruta cualquiera, para almacenar nuestros ejemplos; abrimos el programa de cuantificación y creamos un archivo de trabajo en la carpeta elegida con el botón Nuevo. Como idica la figura 4.18.

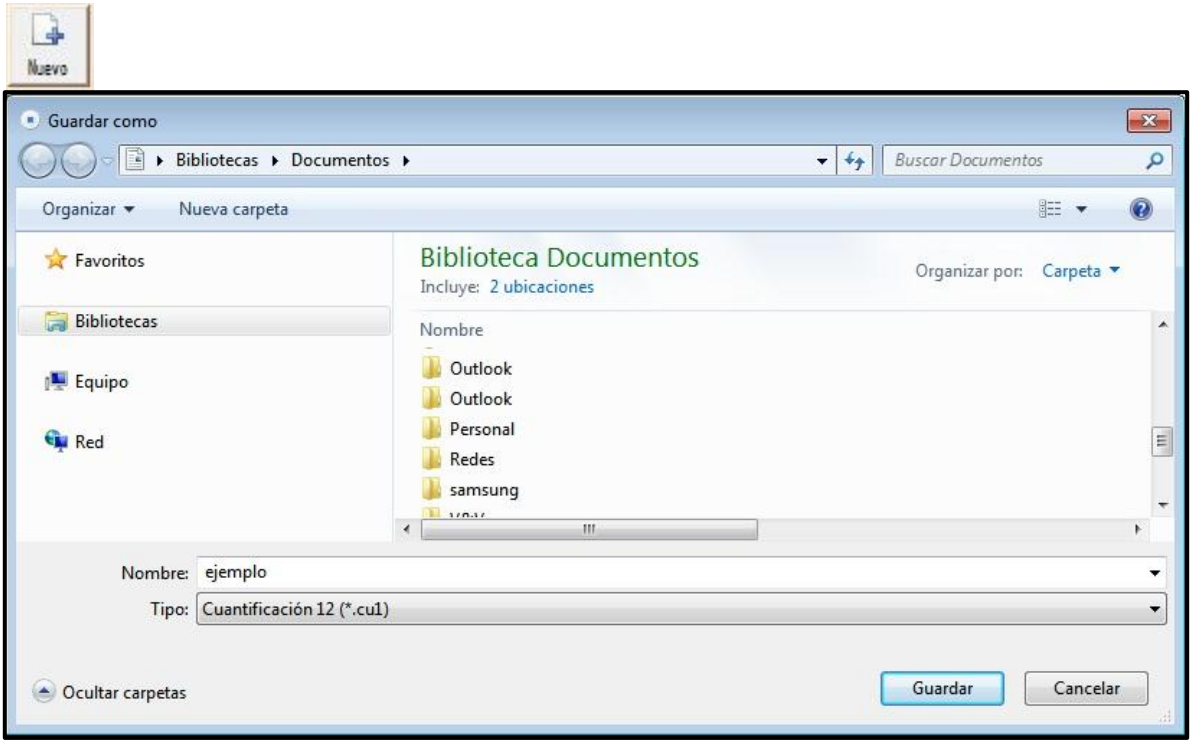


Figura 4.18. Ventana de 'Guardar como'

Definimos los conceptos a cuantificar, esto es en el menú Conceptos

Con el botón agregar, añadimos un concepto para cuantificar, para facilitar el manejo, se sugiere un nombre corto.

VS02, para la viga (Vs-02), IR 18"x89.10 kg/m

VS03, para la viga (Vs-03), IR 18"x68.40 kg/m

L01, para el larguero (L-01), IR 12"x23.9 kg/m

L02, para el larguero (L-02), IR 12"x32.8 kg/m

L03, para el larguero (L-03), IR 12"x38.7 kg/m

Ahora para cuantificar debemos seleccionar un concepto, para iniciar elegimos **VS02**, y como en el plano los objetos que vamos a seleccionar son líneas, utilizamos el botón General.

Con este botón, al darle click nos dirige a Autocad y activa el comando de selección de objetos, con ello seleccionamos todas las líneas correspondientes a **VS02**, en cuanto distinguimos todos los objetos, para terminar el comando damos click al botón derecho del mouse; con esto comienza el trabajo interno del programa marcando y etiquetando los elementos seleccionados, al termino de este proceso nos dirigimos al programa de cuantificación donde se nos muestra en la tabla principal el listado de lo cuantificado, como lo se ejemplifica en la figura 4.19.

No.	Long. (l)	Área (A)	Altura (h)	l * h	Signo	Concepto	Compuesto	Color	Marca	Objeto	Id. Obj.
29	5.526	0	4.26	23.539	+	VS02	0	6	9	Line	BA2C1
30	5.51	0	4.26	23.472	+	VS02	0	6	9	Line	BA2C1
31	5.509	0	4.26	23.47	+	VS02	0	6	9	Line	BA2C1
32	2.928	0	4.26	12.475	+	VS02	0	6	9	Line	BA2C1
33	2.226	0	4.26	9.481	+	VS02	0	6	9	Line	BA2C1
34	4.613	0	4.26	19.65	+	VS02	0	6	9	Line	BA2C1
35	5.613	0	4.26	23.91	+	VS02	0	6	9	Line	BA2C1
36	4.613	0	4.26	19.65	+	VS02	0	6	9	Line	BA2C1
37	5.613	0	4.26	23.91	+	VS02	0	6	9	Line	BA2C1
38	4.613	0	4.26	19.65	+	VS02	0	6	9	Line	BA2C1
39	5.613	0	4.26	23.91	+	VS02	0	6	9	Line	BA2C1
40	4.613	0	4.26	19.65	+	VS02	0	6	9	Line	BA2C1
41	5.613	0	4.26	23.91	+	VS02	0	6	9	Line	BA2C1
42	6.785	0	4.26	28.903	+	VS02	0	6	9	Line	BA2C1
43	5.938	0	4.26	25.294	+	VS02	0	6	9	Line	BA2C1
44	6.785	0	4.26	28.903	+	VS02	0	6	9	Line	BA2C1
45	6.035	0	4.26	25.71	+	VS02	0	6	9	Line	BA2C1
46	6.785	0	4.26	28.903	+	VS02	0	6	9	Line	BA2C1
47	5.938	0	4.26	25.294	+	VS02	0	6	9	Line	BA2C1
48	6.785	0	4.26	28.903	+	VS02	0	6	9	Line	BA2C1
49	6.035	0	4.26	25.71	+	VS02	0	6	9	Line	BA2C1
50	6.785	0	4.26	28.903	+	VS02	0	6	9	Line	BA2C1

Figura 4.19. Tabla principal, listado de conceptos cuantificados

Y el plano queda con las marcas en los elementos indicados.

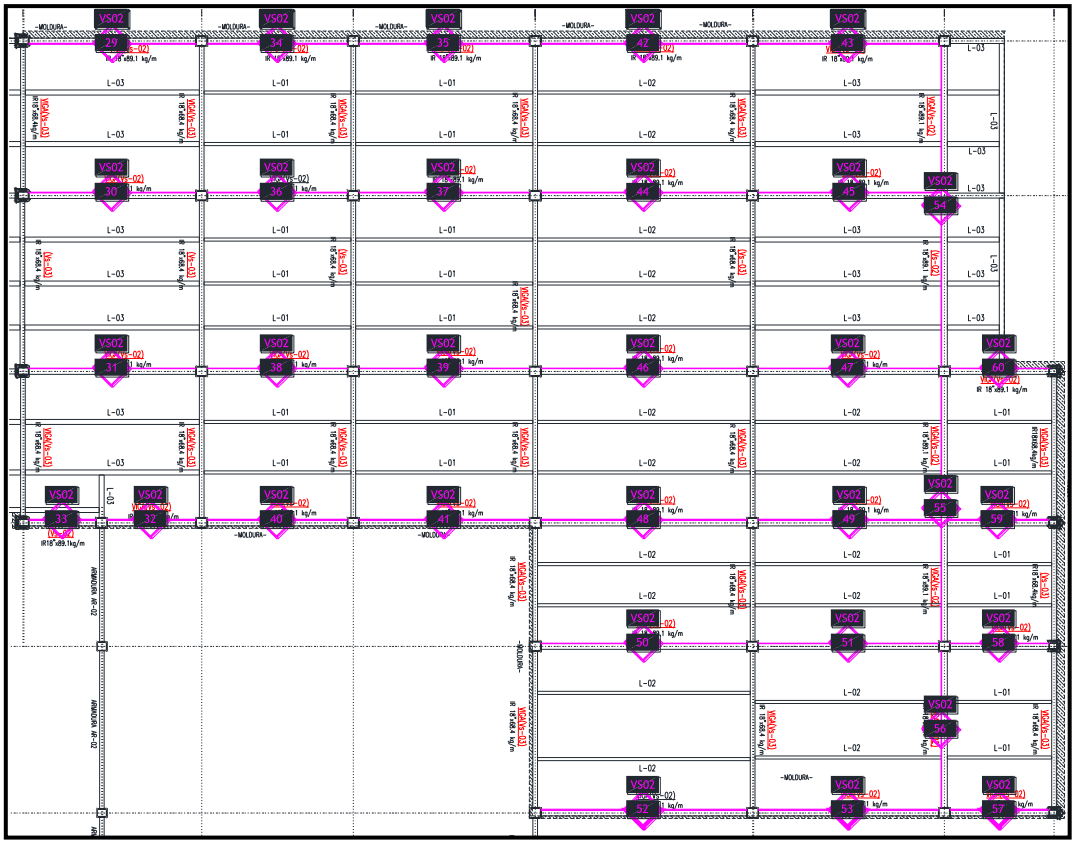


Figura 4.20. Representación del plano cuantificado

Ahora repetimos los pasos para los conceptos faltantes.

Seleccionamos concepto, color y figura; con el botón General seleccionamos las líneas del plano que simbolizan el concepto elegido.

Después de cuantificar todos los conceptos nuestro plano queda así.

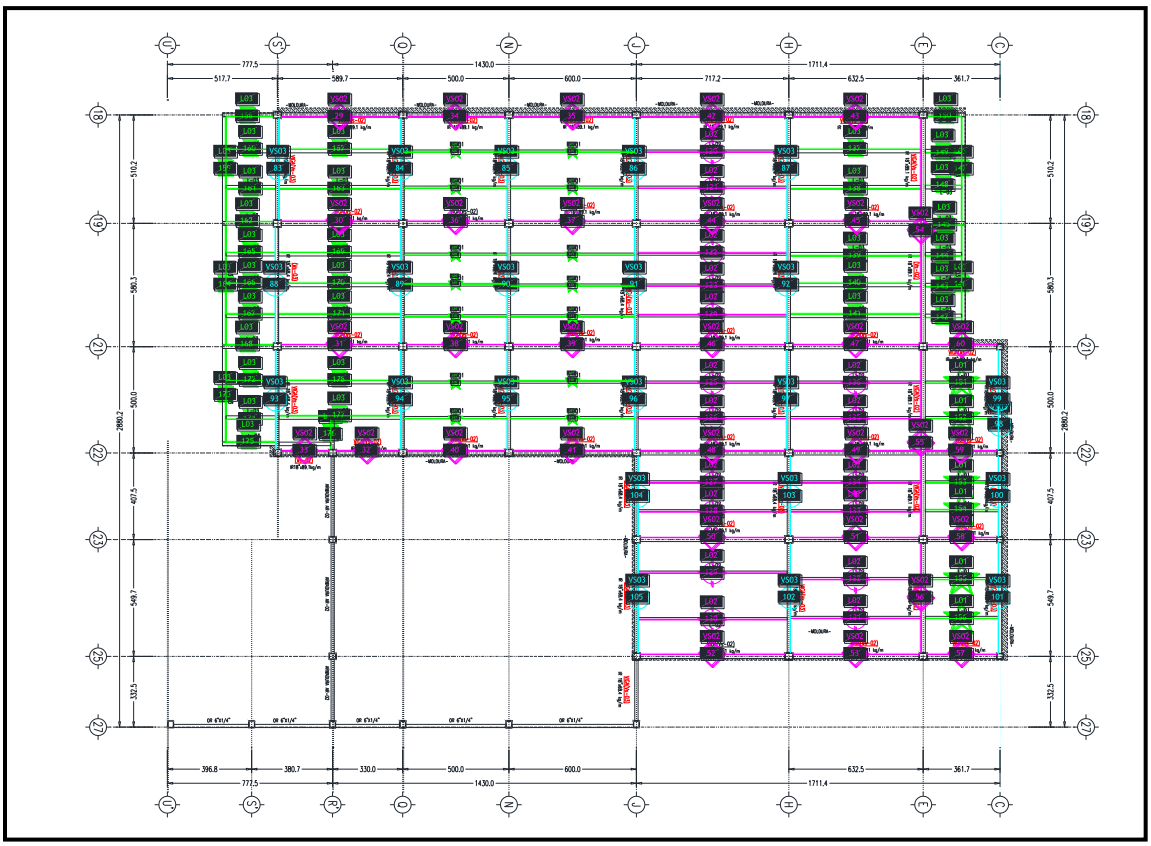


Figura 4.21. Plano estructural, cuantificación de vigas principales y secundarias

En detalle de una sección

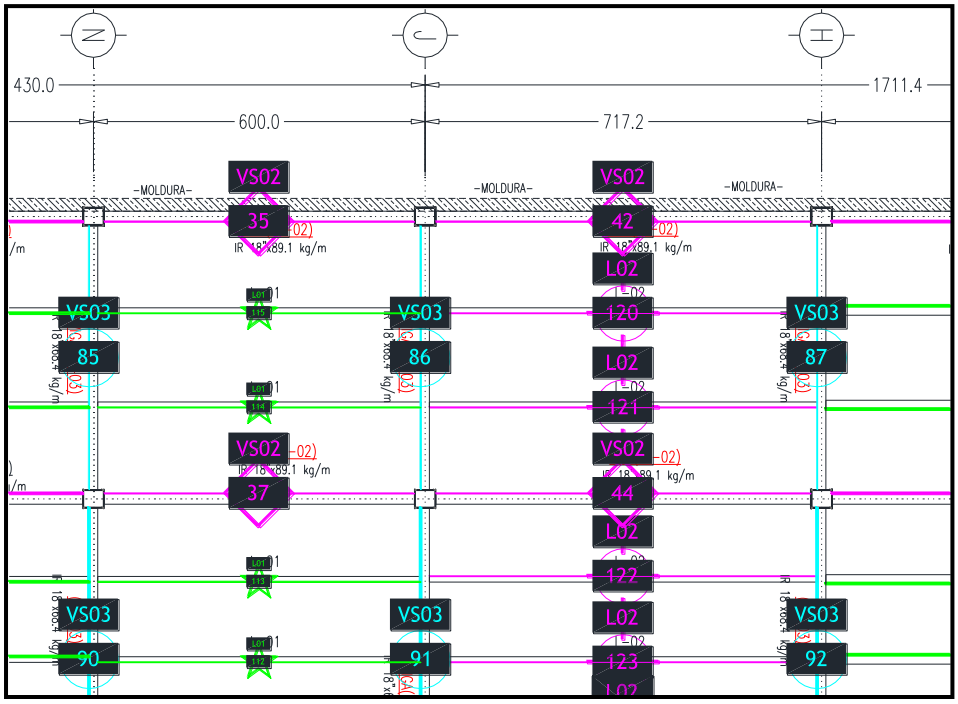


Figura 4.22. Detalle del plano cuantificado

Al terminar de cuantificar los conceptos de la lista, procedemos a realizar un reporte de lo cuantificado, para Autocad y para Excel.

Con el botón Reportes se despliga la ventana del mismo nombre donde nos muestra un árbol para seleccionar los conceptos que deseamos obtener su reporte

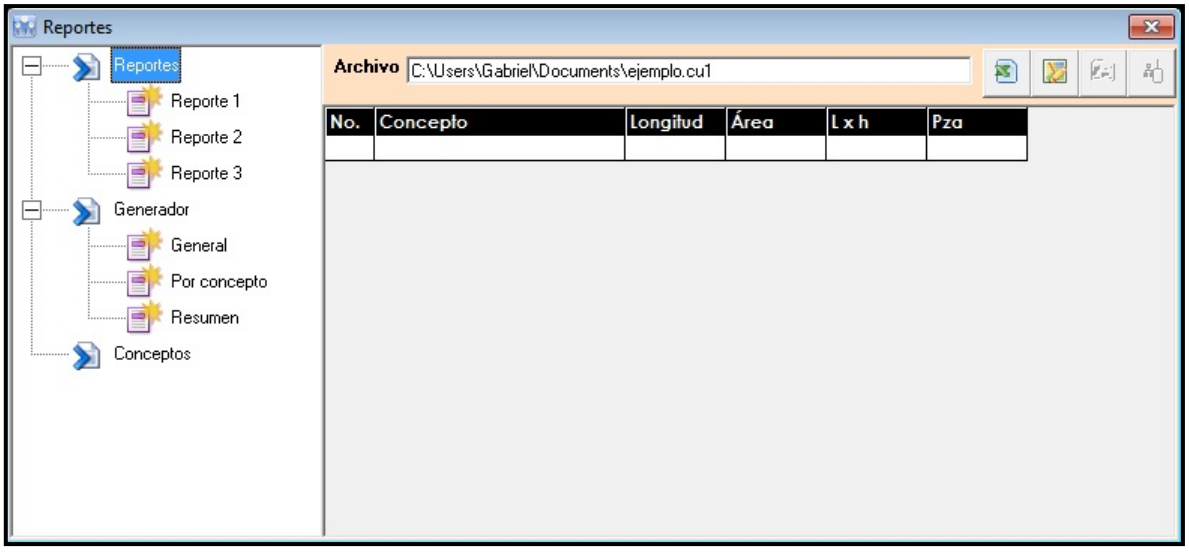


Figura 4.23. Ventana de Reportes

Para nuestro caso necesitamos la parte de Generador, de donde; General, obtenemos todos los conceptos cuantificados; Por concepto, se selecciona un concepto y es el único que nos muestra en la tabla; Resumen, aquí nom muestra los conceptos cuantificados con el total de los valores obtenidos.

ID	No.	Concepto	Long. (l)	Área (A)	Altura (h)	l * h	Matriz	Signo
1	106	L01	4.846	0	4.26	20.644	0	+
2	107	L01	4.846	0	4.26	20.644	0	+
3	108	L01	4.846	0	4.26	20.644	0	+
4	109	L01	4.846	0	4.26	20.644	0	+
5	110	L01	4.846	0	4.26	20.644	0	+
6	111	L01	5.846	0	4.26	24.904	0	+
7	112	L01	5.846	0	4.26	24.904	0	+
8	113	L01	5.846	0	4.26	24.904	0	+
9	114	L01	5.846	0	4.26	24.904	0	+
10	115	L01	5.846	0	4.26	24.904	0	+
11	116	L01	4.846	0	4.26	20.645	0	+

Figura 4.24. Tabla de generador general

Tabla que muestra el Generador por concepto, se eligio en concepto VS02

ID	No.	Concepto	Long. (l)	Área (A)	Altura (h)	l * h	Matriz	Signo
1	29	VS02	5.526	0	4.26	23.539	0	+
2	30	VS02	5.51	0	4.26	23.472	0	+
3	31	VS02	5.509	0	4.26	23.47	0	+
4	32	VS02	2.928	0	4.26	12.475	0	+
5	33	VS02	2.226	0	4.26	9.481	0	+
6	34	VS02	4.613	0	4.26	19.65	0	+
7	35	VS02	5.613	0	4.26	23.91	0	+
8	36	VS02	4.613	0	4.26	19.65	0	+
9	37	VS02	5.613	0	4.26	23.91	0	+
10	38	VS02	4.613	0	4.26	19.65	0	+
11	39	VS02	5.613	0	4.26	23.91	0	+

Figura 4.25. Tabla de generador por concepto

Tabla que muestra el Generador Resumen

No.	Concepto	Longitud	Área	L x h	Pza
1	L01	95.51	0	406.87	20
2	L02	114.11	0	486.13	17
3	L03	136.22	0	580.29	36
4	VS02	176.74	0	752.87	32
5	VS03	140.54	0	598.7	23

Figura 4.26. Tabla de generador resume

Luego de elegir la tabla que necesitamos, indicamos donde queremos enviar la información a Autocad o Excel.

 Con este botón la tabla, se crea en Excel, en un formato predeterminado.

 Con este botón la tabla creada, se escribe en Autocad, en el formato tipo.

Ejemplo de la tabla Resumen, importada a Excel, figura 4.27.

F15

	A	B	C	D	E	F	G
1	No.	Concepto	Longitud	Área	L x h	Pza	
2	1	L01	95.51	0	406.87	20	
3	2	L02	114.11	0	486.13	17	
4	3	L03	136.22	0	580.29	36	
5	4	VS02	176.74	0	752.87	32	
6	5	VS03	140.54	0	598.7	23	
7							
8							

Figura 4.27. Reporte en Excel

Ejemplo de la tabla Resumen, importada a Autocad, figura 4.28.

No.	Concepto	Longitud	Área	L x h	Pza
1	L01	95.51	0	406.87	20
2	L02	114.11	0	486.13	17
3	L03	136.22	0	580.29	36
4	VS02	176.74	0	752.87	32
5	VS03	140.54	0	598.7	23

Figura 4.28. Reporte en Autocad

Al tener nuestros reportes en Excel o Autocad, fácilmente los podemos imprimir con las herramientas propias de cada programa y obtener sencillamente los valores de las cuantificaciones.

Ejemplo de aplicación 2.

Vamos a cuantificar las trayectorias de cableado de una pequeña instalación eléctrica. Donde obtendremos de manera precisa, tubería, cable con forro y desnudo. Dentro del plano eléctrico debemos localizar la tabla que contenga información acerca de las cédulas de cableado, como se muestra en la figura.

Figura 4.29. Cédulas de caleado

Ahora, debemos definir dentro del programa, conceptos con el desglose de la cédula correspondiente, de la manera siguiente.

En la ventana "Conceptos", primero agregamos con el botón "partida", una partida con el nombre "Electrico", agregamos la cédula necesaria, en este caso, "CED02", como se ejemplifica en la figura 4.30.

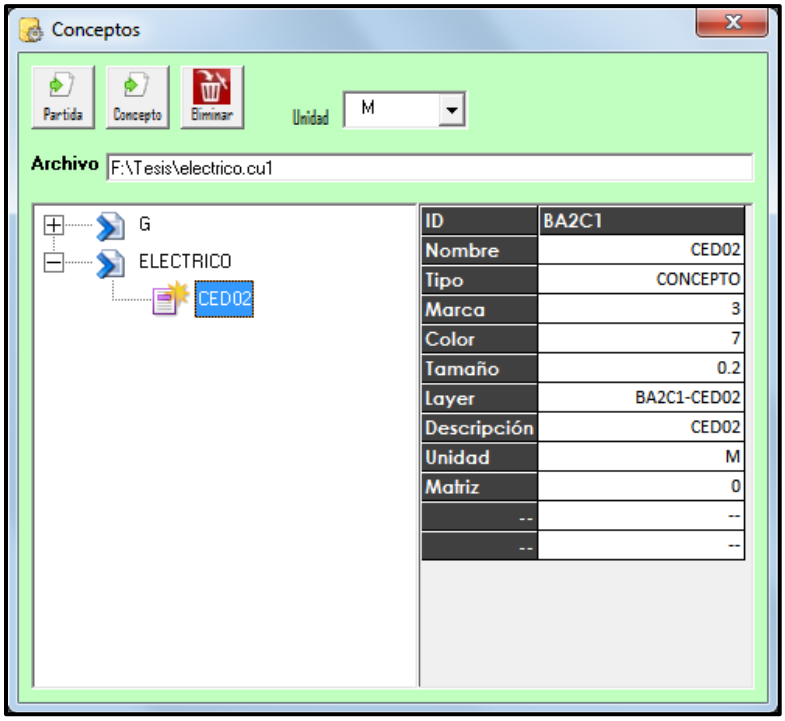


Figura 4.30. Agregar conceptos

Para agregar una matriz se tiene que dar doble clic sobre el número que aparece el concepto matriz[9]donde abrirá una ventana para definir una matriz.

En dicha ventana vamos a ensamblar la cédula de cableado número 2, la cual nos dice que necesitamos dos cables THW del número 10 y un cable desnudo del número 12 en una tubería de 19 mm. Esto se hace seleccionando primeramente la partida se se necesita, "eléctrico", y el grupo de materiales que vamos a utilizar, "cable", después de seleccionar la partida y el grupo nos muestra una lista de material de donde debemos seleccionar lo que nos indica la cédula de cableado. Para el caso del cable del número 10 que necesitamos dos tramos, en la celda "factor 1" colocamos la cantidad 2. Al terminar de agregar los conceptos necesarios, le damos click en "Agregar", es con esto se almacena en la base de datos el concepto con su desglose, ver figura 4.31.

De esta forma cargamos todas las cédulas indicadas en la simbología; considerando los pasos anteriores, regresamos a la ventana principal y procedemos a cuantificar.

[9]**Valores de la matriz**; Cero significa que el concepto no tiene matriz. Uno, indica que el concepto contiene matriz, la cual la podemos visualizar al dar click con el botón derecho sobre el concepto.

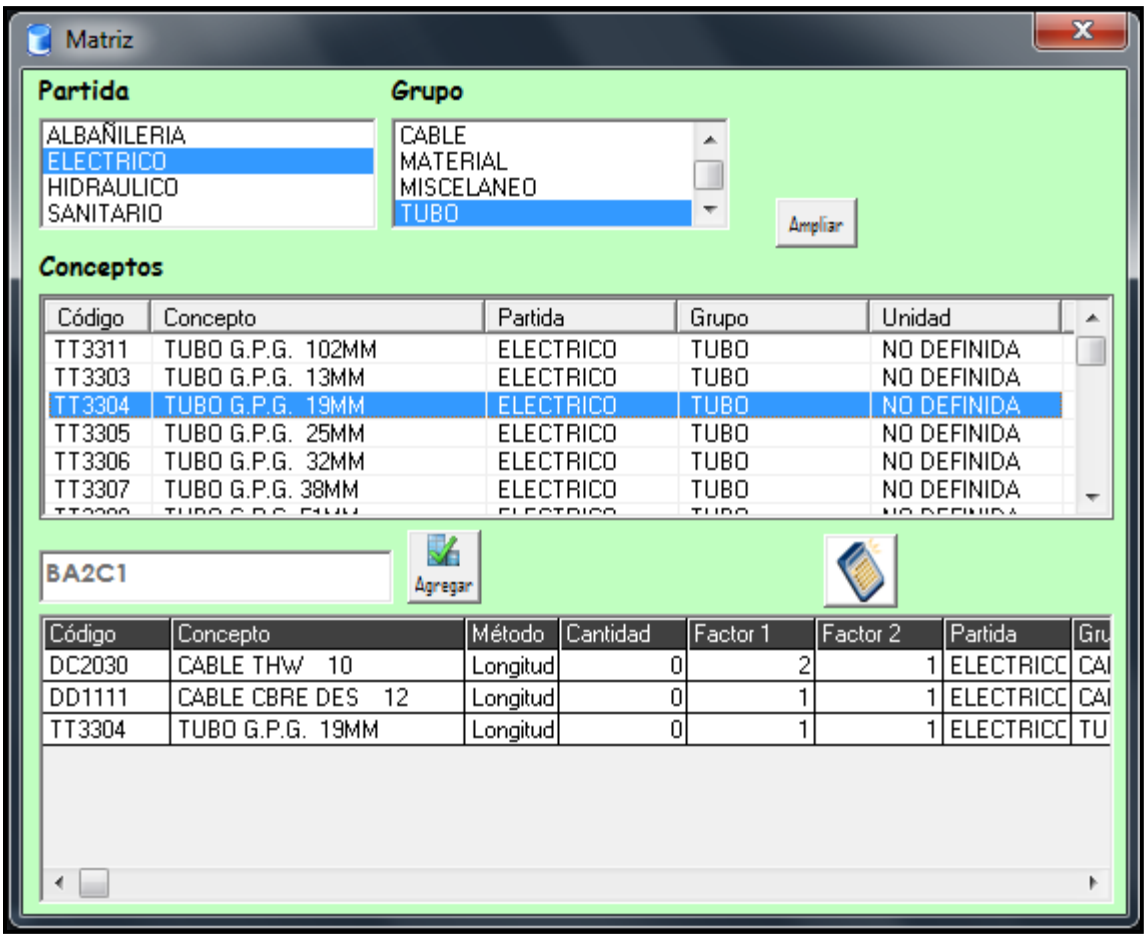


Figura 4.31. Creación de matrices

Como nuestro concepto lo único que se necesita obtener es longitud del tramo; en la ventana principal en la pestaña "Opciones" habilitamos únicamente Concepto, No. Elemento y Longitud; ahora en la pestaña de "Cuantificar" seleccionamos el concepto para cuantificarlo;iniciamos con "CED02", y dentro del plano seleccionamos las líneas que están indicadas con la cédula 2, cuando terminamos de seleccionar los objetos, finalizamos con el botón derecho del ratón.

Después de haber cuantificado los conceptos con las diferentes cédulas, de manera general, como la figura 4.31 se ve nuestro plano cuantificado.

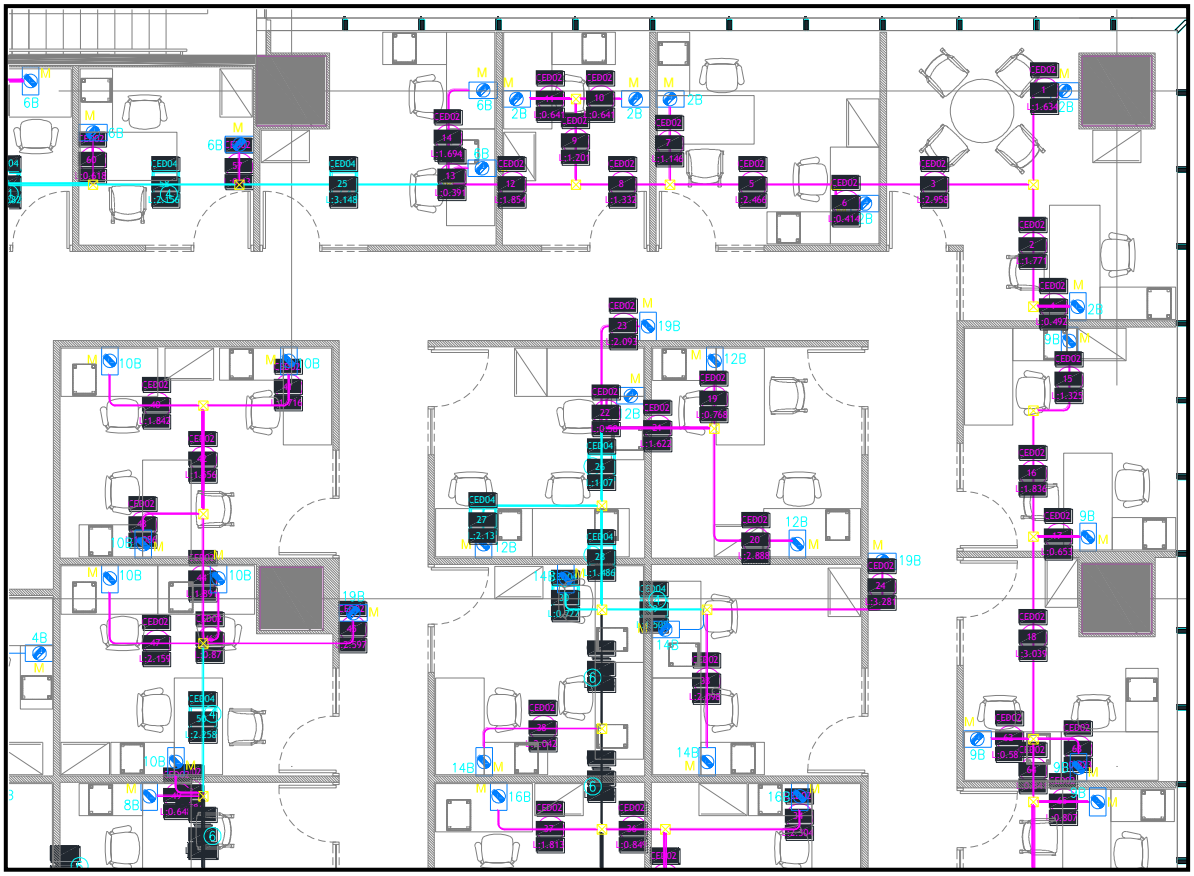


Figura 4.32. Plano con cuantificación de trayectorias eléctricas

Y de manera particular, así se indica un tramo cuantificado, como la figura 4.32.

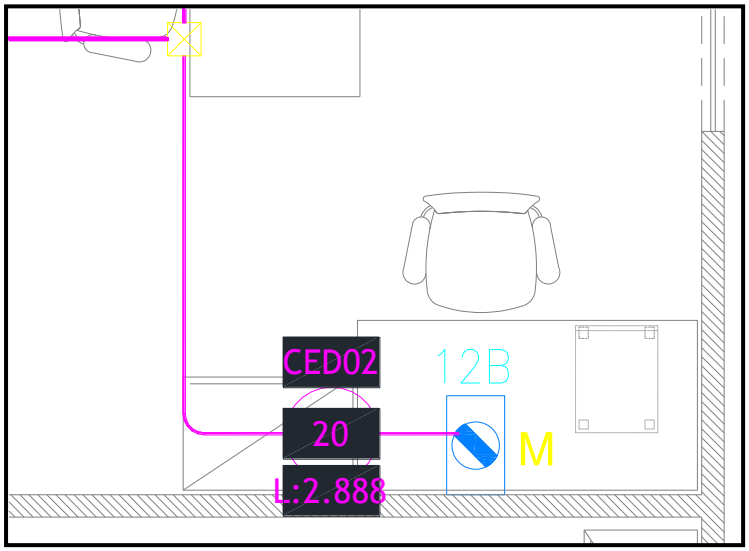
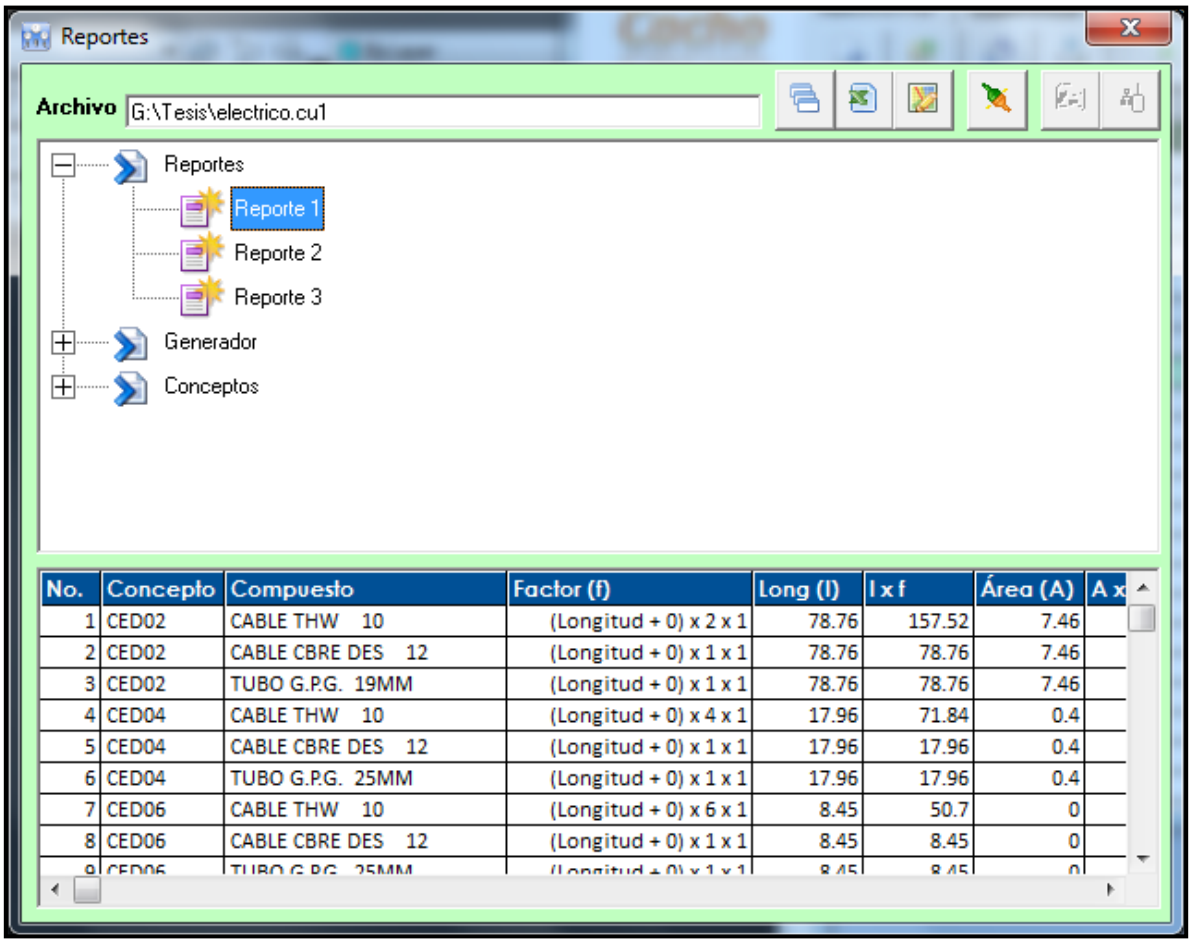


Figura 4.32. Detalle de cuantificación eléctrica

Ya que tenemos nuestros tramos medidos, procedemos a realizar nuestro Reporte, esto lo realizamos el la ventana de "Reportes", del árbol seleccionamos "Reportes" donde se despliegan 3 diferentes opciones, ver figura 4.33.



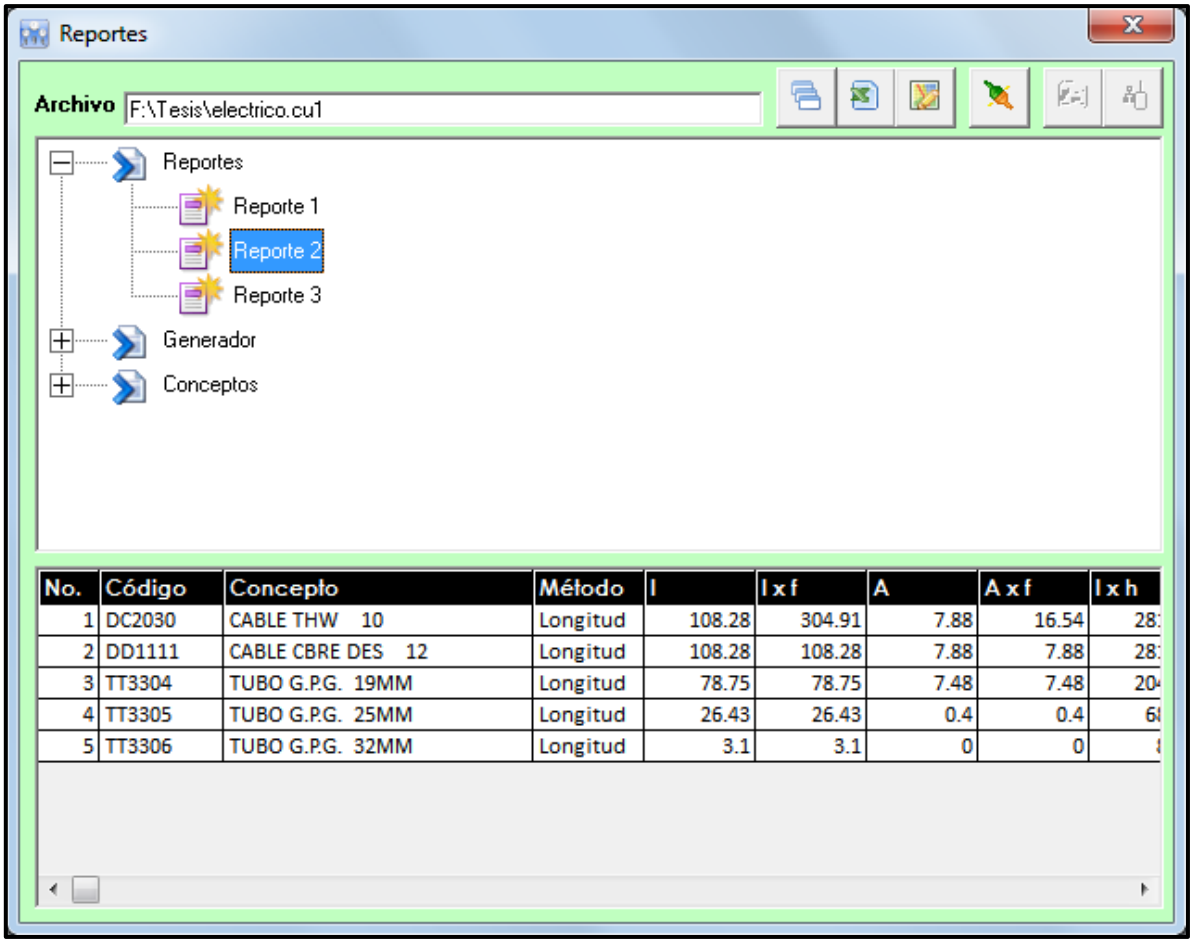
No.	Concepto	Compuesto	Factor (f)	Long (l)	l x f	Área (A)	A x
1	CED02	CABLE THW 10	(Longitud + 0) x 2 x 1	78.76	157.52	7.46	
2	CED02	CABLE CBRE DES 12	(Longitud + 0) x 1 x 1	78.76	78.76	7.46	
3	CED02	TUBO G.P.G. 19MM	(Longitud + 0) x 1 x 1	78.76	78.76	7.46	
4	CED04	CABLE THW 10	(Longitud + 0) x 4 x 1	17.96	71.84	0.4	
5	CED04	CABLE CBRE DES 12	(Longitud + 0) x 1 x 1	17.96	17.96	0.4	
6	CED04	TUBO G.P.G. 25MM	(Longitud + 0) x 1 x 1	17.96	17.96	0.4	
7	CED06	CABLE THW 10	(Longitud + 0) x 6 x 1	8.45	50.7	0	
8	CED06	CABLE CBRE DES 12	(Longitud + 0) x 1 x 1	8.45	8.45	0	
9	CED06	TUBO G.P.G. 25MM	(Longitud + 0) x 1 x 1	8.45	8.45	0	

Figura 4.33. Ventana con Reporte 1

Donde se muestra la cantidad resumida de cada cédula. Explicando los resultados de la tabla, por cada renglón, ver figura 4.33.

El renglón 1, 2 y 3, son los componentes de la cédula 2; el total de las trayectorias se muestra en la columna "Long (l)"; y la columna "l x f" es el total por el factor definido al momento de crear la matriz, en la columna de "factor (f)" se muestra la fórmula aplicada en este concepto, en consecuencia sólo tenemos que interpretar los valores de dicha tabla, en este caso como lo que necesitamos el la longitud, es la columna a la que damos lectura.

En la opción "Reporte 2"



No.	Código	Concepto	Método	l	l x f	A	A x f	l x h
1	DC2030	CABLE THW 10	Longitud	108.28	304.91	7.88	16.54	28
2	DD1111	CABLE CBRE DES 12	Longitud	108.28	108.28	7.88	7.88	28
3	TT3304	TUBO G.P.G. 19MM	Longitud	78.75	78.75	7.48	7.48	20
4	TT3305	TUBO G.P.G. 25MM	Longitud	26.43	26.43	0.4	0.4	6
5	TT3306	TUBO G.P.G. 32MM	Longitud	3.1	3.1	0	0	

Figura 4.34. Ventana con reporte 2

Donde se muestra el resumen por material; es decir, para el renglón 1, la suma de todas las trayectorías de todas las cédulas que contenga el concepto CABLE THW 10, por ello en comparación con el "Reporte 1" donde tenemos en los renglones 1, 4, 7 y 10, las cantidades de 78.76, 17.96, 8.45 y 3.11 respectivamente para cada cédula, en el "Reporte 2" nos muestra la suma de estos cuatro valores, que es 108.28, así como el total de la longitud afectada por su factor correspondiente, lo que nos resulta que por todos los tramos medidos el "CABLE THW 10" es 304.91, este valor nos sirve ya sea para incluir en un presupuesto o directamente hacer una compra. Como interpretación de los valores útiles de la tabla, ver figura 4.35., además considerando la escala de nuestro plano donde hicimos las mediciones, tenemos la figura 4.35.

Código	Concepto	L x f
DC2030	CABLE THW 10	304.91m
DD1111	CABLE CBRE DES 12	108.28m
TT3304	TUBO G.P.G. 19MM	78.75m
TT3305	TUBO G.P.G. 25MM	26.43m
TT3306	TUBO G.P.G. 32MM	3.1m

Figura 4.35. Tabla con resumen de materiales

Conestos valores podemos gestionar para ajustar los valores a las cantidades y presentaciones del material en la venta, es decir, nuestro cable desnudo tiene 108.28m y los presentaciones se venden cajas de 100m; entonces debemos comprar 4 cajas de este material.

CONCLUSIONES

El presente trabajo sienta las bases para programar en plataforma Visual Basic, e interactuar entre Autocad, Excel y base de datos; dichas bases, permiten desarrollar nuevas aplicaciones en servicio y mejora de los trabajos de ingeniería, agilizar la representación gráfica en Autocad a partir de los cálculos de Excel y además del caso contrario quepermite el envio de información a Excel con origen en planos de Autocad. Y no sólo en el área de costos y/o administración de obra, sino se puede extender al diferentes ámbitos.

Este trabajo esta enfocado a la parte de generación, administración y control de obra, apoyado escencialmente en la gestión de información en formato digital. Con lo que nos facilita la obtención de resultados.

Con una de las herramientas de Visual Basic, se compila el programa y se crea un archivo tipo aplicación el cual estará disponible el ejecutar el archivo; y se puede utilizar de manera independiente; donde se inicia con una portada y se administra desde la ventana principal.

Este proceso ahorra bastante tiempo y mejora la calidad del trabajo final; de acuerdo a los lineamientos de la mayoría de las empresas en México. Ya que desempeña todo el proceso de generación de obra como si se hiciera manualmente, sólo que conun ahorro de tiempo considerable; sólo determinando y dando una interpretación a la información obtenida. A partir de la medición del plano, indicación del elemento con una marca y un color en específico, gestión de datos y escritura en los formatos de generador con la alternativa de vincular a un presupuesto de Excel.

BIBLIOGRAFÍA:

SUAREZ SALAZAR CARLOS, 2007, COSTO Y TIEMPO EN EDIFICACION, MEXICO, LIMUSA

PLAZOLA CISNEROS ALFREDO, 1991, NORMAS Y COSTOS DE CONSTRUCCION, MÉXICO, LIMUSA

BALENA, FRANCESCO, 2002, VISUAL BASIC VERSION 6.0, PROGRAMACION AVANZADA CON MICROSOFT, ESPAÑA, MCGRAW-HILL

PRESTAMO RODRIGUEZ JONATHAN, 2000, CURSO PRACTICO DE PERSONALIZACION Y PROGRAMACION BAJO AUTOCAD, ESPAÑA, TEXTO LIBRE, PUBLICACION DE INTERNET
www.lawebdelprogramador.com/cursos/autocad/autocad_inicio.php

CEBALLOS SIERRA FRANCISCO JAVIER, 1999, MICROSOFT VISUAL BASIC 6: CURSO DE PROGRAMACION, MÉXICO, ALFA OMEGA

INSTITUTO MEXICANO DE LA CONSTRUCCIÓN EN ACERO A.C., 1995, MANUAL DE CONSTRUCCIÓN EN ACERO – DEP VOL. 1, MÉXICO, LIMUSA NORIEGA EDITORES

C.N.I.A.M., 1989, GUIA DE AUTOCONSTRUCCION, MÉXICO, C.N.I.A.M.