UNIVERSIDAD DE OVIEDO
Escuela Politécnica de Mieres

Máster en Teledetección y Sistemas de Información Geográfica

EDICIÓN Y CONTROL DE CALIDAD DE CARTOGRAFÍA ORIENTADA A S. I. G. EN FORMATO DE BASE TOPOGRÁFICA ARMONIZADA CON ARCGIS Y OTRAS HERRAMIENTAS DE DESARROLLO

TRABAJO FIN DE MÁSTER

AUTORA: Iciar Martínez González
TUTORA: Ángela Alonso Fernández

Junio 2012

Agradecimientos:

A César Argul García (SERESCO).

Índice

Memoria

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Anejos

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Índice de Figuras

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Memoria

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Iciar Martínez González
Ingeniera Técnica en Topografía.
Graduada en Ingeniería Geomática y Topografía.

"Edición y control de calidad de cartografía orientada a S.I.G. en formato de Base Topográfica Armonizada con ArcGIS y otras herramientas de desarrollo".

Resumen.
El presente documento constituye la memoria del Trabajo Final del Máster Universitario en Teledetección y Sistemas de Información Geográfica, organizado por la Universidad de Oviedo.
El mismo surge como una propuesta para automatizar tareas en proyectos de producción cartográfica que hayan de cumplir las normas de la "Base Topográfica Armonizada".
Este proyecto consiste en la creación y organización de comandos de edición y análisis dentro del entorno de ArcMap, agrupando una serie de funcionalidades creadas o preexistentes, en una nueva barra de herramientas.
En cuanto al desarrollo de la aplicación, la programación de barra se realiza íntegramente con VBA (Visual Basic for Aplications) a través del editor integrando en ArcMap, haciendo uso de librerías ArcObjects. Por tanto, se trata de una personalización de la interface a nivel interno, ya que el código ha sido generado dentro del propio entorno de ArcMap y almacenado en el proyecto activo.
Por otra parte, los controles que integran la aplicación se han organizado según su funcionalidad: A cada uno, se le asocia un código VBA, de tal manera que al ser pulsado se accede a una funcionalidad dentro de la vista de trabajo.

Palabras Clave: ArcGIS, desarrollar, Visual Basic for Applications, programación, personalización.

Abstract.
This document is the memory of the final project of the Master's Degree in Remote Sensing and Geographic Information Systems, organized by the University of Oviedo.
It arises for automating tasks in cartographic production projects that are to meet the standards of the "Harmonized Topographic Base."
This project involves the creation and organization of editing commands and analysis within the ArcMap environment, grouping a number of features created or existing in a new toolbar.
In terms of application development, programming bar has bees done entirely with VBA (Visual Basic for Applications) through the editor integrated into ArcMap, using ArcObjects libraries. Therefore, it is a customization of the interface internally, since the code has been generated inside the ArcMap environment and stored in the current project.
Moreover, the controls that make up the application have been organized by functionality: A VBA code is associated to each, so that a functionality is accessed in the working view, when clicked.

Keywords: ArcGIS, to develop, Visual Basic for Applications programming, customization.

1. INTRODUCCIÓN

El Consejo Superior Geográfico (CSG) es el órgano de dirección del Sistema Cartográfico Nacional, depende del Ministerio de Fomento y ejerce la función consultiva y de planificación de la información geográfica y la cartografía oficial. Las principales funciones del CSG son la proposición del Plan Cartográfico Nacional y su coordinación con los planes y programas cartográficos de todas las Administraciones públicas y la normalización de criterios de producción cartografía. Está constituido por comisiones especializadas como órganos de estudio y propuesta de normas y decisiones a probar por el CSG. Actualmente las comisiones establecidas son:

- Comisión del Plan Cartográfico Nacional
- Comisión de Normas Cartográficas (CNC)
- Comisión de Nombres Geográficos
- Comisión de Infraestructuras de Datos Espaciales
- Comisión de Observación de Territorio

La misión de la CNC consiste en la elaboración de propuestas de normas cartográficas que contengan los criterios cartográficos a los que deberá ajustarse la cartografía incluida en el Sistema Cartográfico Nacional, a fin de armonizar los conjuntos de datos geográficos realizados por distintos productores y facilitar el uso de de la información geográfica.

La publicación en 1992 de las normas para la elaboración de cartografía a diversas escalas nunca fue actualizada teniendo en cuenta los requerimientos de la producción de información digital, es decir, mediante la incorporación de las tecnologías digitales en los flujos de producción, el uso de los Sistemas de Información Geográfica para la explotación del producto o su integración en una Infraestructura de Datos Espacial.

Ante la ausencia de unas normas comunes, cada administración aplicó su criterio y metodología para adaptar las normas existentes para productos analógicos a la generación de productos digitales consiguiendo a finales de los 90 conjuntos de datos no armonizados que ni eran ni podían ser compartidos ni utilizados de forma interoperable.

1.1 El porqué de la BTA

La Comisión de Normas Cartográficas del Consejo Superior Geográfico ha seguido con atención los avances tecnológicos y cambios metodológicos introducidos en el proceso de obtención de la cartografía y, lo que es más importante, la evolución conceptual habida desde la cartografía, ya sea como mapas en papel o como mapas digitales, hasta la información geográfica, en forma de conjuntos de datos geográficos para ser cargados en un SIG o integrados en una IDE. *(Ministerio de Fomento. CSG,2008).*

Los organismos oficiales y administraciones públicas antaño productores de cartografía oficial distribuyen, en la actualidad, no sólo información cartográfica sino también información geográfica.

En este contexto, las normas cartográficas elaboradas en el seno de la Comisión y publicadas en 1992 son insuficientes para asegurar la compatibilidad de los datos generados por distintos organismos siguiendo dicha normativa.

Manteniendo el espíritu inicial por conseguir la necesaria homogeneidad de la cartografía oficial española a grandes escalas, se han consensuado las presentes especificaciones técnicas, a fin de lograr la armonización de las bases topográficas mediante la definición de un producto virtual llamado Base Topográfica Armonizada (BTA), que permiten la generación de la cartografía topográfica a escalas 1:5 000 o 1:10 000 en las distintas Comunidades Autónomas (CC.AA.), Diputaciones Forales (DD.FF.) o en la Administración General del Estado (AGE) para hacer posible el intercambio de información geográfica digital, su integración e interoperabilidad.

La BTA será por tanto, un conjunto de datos vectoriales de carácter topográfico, formado mediante la armonización de las bases topográficas a escala 1:5 000 y 1:10 000 producidas por las CC.AA. y DD.FF., que cubre todo el territorio español. La BTA está

organizada en bloques, uno por cada productor de datos, y los bloques están formados por hojas según una división que garantiza el case exacto de los marcos de hojas.

El propósito de estas especificaciones es que a cada productor de datos, le sea posible convertir su información original al modelo de datos aquí definido, de modo automático y con un trabajo interactivo mínimo y residual. Para la redacción de este documento se han tenido en cuenta pliegos de condiciones técnicas, diccionarios, especificaciones y demás textos descriptivos de las bases cartográficas de diversas CC.AA., así como los referidos a la BCN25 del IGN. También se han utilizado como material de referencia normas, borradores o documentos de trabajo del Comité Técnico 211 de la Organización Internacional de Estandarización (ISO), en especial los relacionados con las normas ISO19131, ISO19109, ISO19110 e ISO19137; las traducciones realizadas por el comité técnico 148 de AENOR de las normas ISO19115, ISO19113 e ISO19114 y la recomendación de la Comisión de Geomática del Consejo Superior Geográfico sobre el Núcleo Español de Metadatos (NEM).

1.2 Especificaciones Técnicas

Estas especificaciones son aplicables a la BTA, entendiendo como tal una armonización de las bases topográficas a escala 1:5 000 y 1:10 000 producidas por las CC.AA. y DD.FF. de España. Sin embargo, se han definido de manera que sus principios y filosofía sean también aplicables y de utilidad a otras bases a diferentes escalas, como las bases topográficas del IGN o las bases municipales a escalas grandes, con las ventajas que ello supondría en cuanto a armonización multiescala, posibilidades de generalización de datos, facilidad para determinaciones de calidad, etc.

La diversidad de modelos existentes, en las CC.AA., DD.FF. y en la AGE, y el firme propósito de fijar un núcleo común en cuanto a modelo de aplicación y catálogo de fenómenos, dar cabida a las distintas visiones del territorio, algunas más próximas al mapa que otras, y marcar las directrices para hacer evolucionar el producto hacia una base topográfica con un modelo de aplicación complejo, ha propiciado una primera versión de las especificaciones de la base con un modelo sencillo, de forma que se asegure su uso en un entorno CAD con una pérdida mínima de información y un contenido mínimo recomendado.

1.3 Marco de referencia

1.3.1 Sistema de referencia

La BTA, en aplicación del Real Decreto 1071/2007, de 27 de julio, por el que se regula el sistema geodésico oficial en España, establece el sistema ETRS89 como sistema de referencia geodésico oficial.

En la Península, Baleares, Ceuta y Melilla el sistema de referencia es el ETRS89 (ITRF89 época 89,0) y en Canarias el REGCAN95 (ITRS93 época 1994,9), constituidos por:

Elipsoide GRS80 (Geodetic Referente System 1980)

a = 6 378 137 metros

f = 1:298,257222101

Origen geocéntrico, cuyos ejes son:

Eje X: Intersección del meridiano de Greenwich y el plano del Ecuador medio

Eje Z: Eje de rotación del elipsoide en la dirección del CIO

Eje Y: Perpendicular y formando un triedro directo con los ejes X y Z.

La diferencia entre ambos sistemas es centimétrica y se debe a que las coordenadas de las estaciones utilizadas para determinar el sistema de referencia se han obtenido en distintos ajustes.

Teniendo en cuenta que estamos en un período de transición y que, en general, el sistema de referencia en el que se encuentran los datos es el denominado ED50 se recomienda que el proceso de transformación de ED50 a ETRS89 se haga siguiendo las indicaciones del Grupo de Trabajo para la transición a ETRS89 (GT-ETRS89) del Consejo Superior Geográfico.

Sistema geodésico

El sistema de referencia se materializa sobre el territorio mediante los vértices REGENTE de la Red Geodésica Nacional y su densificación en las distintas CC.AA. o DD.FF.

Las altitudes están referidas al nivel medio del mar y quedarán materializadas en el territorio por los vértices de las líneas de Nivelación de Alta Precisión. Caso de trabajar con altitudes elipsoidales se recomienda el uso del geoide IBERGEO2006 o uno similar que asegure una precisión equivalente o superior.

Las altitudes están referidas al nivel medio del mar definido por el mareógrafo fundamental de Alicante para la Península y, por el mareógrafo o escalas de mareas ubicados en diferentes puertos para las islas y ciudades de Ceuta y Melilla.

Sistema cartográfico de representación

El sistema de representación plana es la proyección conforme Universal Transversa de Mercator (UTM), recomendándose siempre la utilización del huso que corresponda en cada caso para evitar la utilización de husos extendidos.

Si se intercambian, gestionan o representan datos en coordenadas UTM, cada hoja debería ir en su huso. En el caso de tener parte en un huso y parte en otro, en el huso en el que tenga mayor superficie y si las dos partes son iguales, en el huso 30.

1.3.2 Organización por hojas

Aunque el contenido de las distintas bases topográficas es continuo, para su planificación, captura, gestión, almacenamiento, distribución y actualización se siguen organizando los datos, en la mayoría de casos, en hojas basadas en la división en hojas del Mapa Topográfico Nacional 1:50 000 (MTN50).

Por lo tanto, conviene fijar criterios comunes para el cálculo de las esquinas de hoja y su denominación. Las hojas extendidas y hojas especiales (como las hojas MTN25 de Canarias), no entran dentro de esta división en hojas, que no es más que un mecanismo de indexación común para datos SIG. Cada productor de cartografía puede, según sus necesidades, definir las hojas compuestas, extendidas o especiales que juzgue conveniente

para su impresión en papel, siempre que defina claramente de qué hojas, de las que forman la división armonizada está compuesta cada una.

Corte geodésico

En lo que a división por hojas se refiere, el corte geodésico del MTN50 establecido en el R.D. 1071/2007, de 27 de julio, está compuesto por hojas de 10' en latitud y 20' en longitud siguiendo los meridianos y paralelos.

Para que el cambio, respecto al corte vigente hasta la fecha de publicación del R.D., origine un desplazamiento mínimo respecto del corte actual en ED50, se ha tomado como polo el punto definido por las coordenadas geodésicas $\lambda = -9^{o}\ 51'\ 15''$ y $\varphi = 44^{o}\ 00'\ 00''$.

Según las premisas anteriores, la línea de borde de una hoja 1:50 000 se define como una recta en coordenadas geográficas, constituida por una poligonal recta en ETRS89 cuyos vértices se obtienen de la subdivisión de dicha línea en 80 segmentos iguales para asegurar la coincidencia de esquinas hasta las hojas a escala 1:500. El resultado es una poligonal de 81 vértices y el marco de hoja MTN50 será un polígono formado por 320 vértices, el marco de una hoja 1:25 000 un polígono de 160 vértices, el marco de una hoja 1:10 000 un polígono de 80 vértices y el marco de una hoja 1:5 000 un polígono de 40 vértices.

Las coordenadas geográficas de los vértices de cualquier hoja, siempre en ETRS89, se pueden convertir a ED50 mediante la mejor transformación disponible propuesta y publicada por el Grupo de Trabajo para la transición a ETRS89 (GT-ETRS89) del Consejo Superior Geográfico, con la precisión requerida hasta escala 1:5.000.

Denominación

El identificador de cada hoja se define como una cadena de caracteres formada por el número de hoja del MTN50 expresado con 4 dígitos y rellenado con ceros por la izquierda, un carácter para el tipo de hoja (A normal, B bis, C

tris,…), guión, y la columna-fila, que ocupa cuatro posiciones en total, y cuyos valores máximos dependerán de la escala de la serie. Si se trata de una hoja del MTN50 columna y fila tomarán el valor cero. Por ejemplo: 0123A-0304 identifica una zona correspondiente a la columna 3 fila 4 de la hoja MTN50 0123A.

Para convertir el número de hoja, identificador único de una hoja dentro de una serie cartográfica, en un identificador único absoluto, independiente de la serie, añadir dos dígitos por la izquierda que indiquen la escala a cuya división nos referimos, separados por dos puntos: 50 para 1:50 000; 25 para 1:25 000; 10 para 1:10 000 y 05 para 1:5 000. Así, el ejemplo anterior permitiría diferenciar 10:0123A-0304 (serie 1:10 000) de 05:0123A-0304.

Se recomienda que cada productor de cartografía o bien adopte el método de nomenclatura descrito, o bien construya y ponga a disposición pública, la tabla de equivalencia entre la nomenclatura de la serie y la nomenclatura recomendada.

1.4 Modelo de datos

Los entes del mundo real están agrupados en la BTA en clases con propiedades comunes. Cada una de estas clases determina un tipo de fenómeno, siendo éste el nivel básico de clasificación del Catálogo de fenómenos. Se consideran tipos de fenómeno genéricos y otros más específicos, teniendo el Catálogo una estructura jerárquica con supertipos y subtipos de fenómeno, con un número ilimitado de niveles. Tanto unos como otros pueden tener atributos, heredando los subtipos los atributos de los supertipos.

A cada tipo de fenómeno se le asigna un nombre y código que lo identifican. El código está definido como un número asignado de forma correlativa, formado por una cadena de caracteres de 4 posiciones, rellenado con ceros a la izquierda. Éste código no implica ni sigue ninguna relación de orden. Los tipos de fenómenos se agrupan en temas, también llamados capas, subconjuntos de datos relativos a fenómenos de una misma temática o categoría, como Hidrografía, Elevación o Vías de Comunicación, que han sido definidos en concordancia con la directiva INSPIRE.

1.5 Diccionario o Catálogo de Fenómenos

El propósito de el Diccionario de Fenómenos es completar las especificaciones técnicas y describir de forma detallada todos y cada uno de los fenómenos que modelan los entes topográficos del mundo real en la BTA 1:5 000 v1.0.

El diccionario está estructurado en dos partes claramente diferenciadas. En la primera se especifican las características generales del catálogo de fenómenos (preámbulo) y en la segunda, que constituye en sí el diccionario, se explicitan de forma concreta y detallada la geometría, definición, método de captura, atributos, clasificación basada en los atributos, criterios de selección, gráficos aclaratorios y recomendaciones para la representación de cada uno de los fenómenos de la BTA 1:5000.

Contiene los fenómenos que representan el conjunto de entes del mundo real seleccionados para describirlo topográficamente.

Características:

Se agrupan en temas o capas con propiedades comunes.
Estructura jerárquica (fenómenos genéricos y otros más específicos)
Cada tipo de fenómeno tiene un código o nombre único que lo identifica.

Temas o capas del catálogo de fenómenos BTA

Puntos de referencia
Redes de transporte
Nombres geográficos
Hidrografía
Relieve
Cubierta terrestre
Edificaciones, poblaciones y construcciones
Servicios e instalaciones

1.6 Calidad de los productos BTA

La calidad de la información geográfica de la base puede variar en función de la calidad de la fuente de datos, del método de captura o de su origen.

Su explotación en un entorno SIG exige el cumplimiento de ciertos requisitos o, al menos, el conocimiento del grado de cumplimiento; por ello, dentro del apartado de calidad se detallan, de acuerdo con las normas ISO19113 e ISO19114, los parámetros que describirán la calidad de los datos así como una propuesta para evaluarlos y los valores esperados.

En el catálogo de fenómenos se mencionarán los controles específicos que se recomienda verificar para cada fenómeno.

La información sobre la calidad de los datos, es decir, los resultados de la evaluación de la calidad se encontrarán en los metadatos.

1.6.1 Exactitud posicional

La evaluación de la exactitud posicional consiste en verificar la proximidad de la posición de los fenómenos con respecto a su posición verdadera o asumida como verdadera. Se hará por muestreo, comparando la posición de un cierto número de vértices de la base con su posición obtenida por métodos independientes y con una exactitud 3 veces mejor que la esperada.

Absoluta horizontal

La exactitud planimétrica se describirá como EMC y sesgo sobre una muestra de al menos 30 puntos bien definidos y distribuidos por hoja MTN50, o superficie equivalente (18km x 36km), de una muestra del 10% del terreno a evaluar.

El resultado esperado es de un sesgo menor que 0,05 m y un EMC de 0,60 m por componente, lo que equivale a una exactitud de 1m en el 90% de los casos.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Absoluta vertical

La exactitud altimétrica se describirá como EMC y sesgo sobre una muestra de al menos 30 puntos bien definidos y distribuidos por hoja MTN50, o superficie equivalente (18kmx36km), de una muestra del 10% del terreno a evaluar.

El resultado esperado es de un sesgo menor que 0,05 m y un EMC de 0,75 m, lo que equivale a una exactitud de 1,25 m en el 90% de los casos.

Relativa vertical

La exactitud vertical relativa se describirá como una variable lógica (cumple/ no cumple); asegura la coherencia altimétrica de las curvas de nivel con determinados fenómenos topográficos como cursos fluviales, carreteras o vías férreas. En el catálogo de fenómenos hay información detallada.

De forma sistemática se comprueba que la distancia entre una curva de nivel y la instancia a evaluar es inferior a la exactitud absoluta vertical.

1.6.2 Compleción

Para describir en qué grado el conjunto de datos es fiel a la realidad se utilizará una muestra del conjunto de datos.

La selección de la muestra se hará por área geográfica, es decir, no se trata de seleccionar un determinado subconjunto de datos sino seleccionar una zona o zonas representativas del territorio y verificar la ausencia de determinados fenómenos que deberían estar en la base o la presencia de datos que no deberían aparecer.

Se recomienda que la superficie de la muestra no sea inferior al 10% del área de cobertura del conjunto de datos. Asimismo, para la selección se tendrá en cuenta el método de obtención de los datos, es decir, la muestra incluirá zonas en las que se ha realizado trabajo de campo y otras cuya información solo proceda de restitución o trabajo de gabinete.

Omisión

Se computará como omisión la ausencia de un fenómeno y no su interpretación errónea. Se expresará mediante el porcentaje de fenómenos omitidos frente al número total de fenómenos del mundo real. El resultado esperado es que no sea superior al 4%.

Comisión

Se computará como comisión la presencia en la base de un fenómeno sobrante, bien sea por no existir en el universo de discurso en la fecha de obtención de los datos (toma de imagen, trabajo de campo) bien sea por no estar contemplado en el modelo de aplicación. Se expresará mediante el tanto por ciento de fenómenos superfluos frente al número de fenómenos del conjunto de datos. El resultado esperado es que no sea superior al 2%.

1.6.3 Consistencia lógica

Los distintos aspectos a comprobar indican el grado de certidumbre con el que se cumplen las especificaciones en lo que respecta a la estructura interna de los datos y la topología.

Consistencia de dominio

La consistencia de dominio se describirá como una variable lógica (cumple/ no cumple) cuyo significado muestra que no existen instancias no previstas.

Control de códigos: Control para asegurar que no hay instancias con códigos que no estén en el catálogo de fenómenos.

Control de atributos: Control para garantizar que los atributos alfanuméricos que describen al objeto están incluidos, y además que sus valores pertenecen al dominio previsto.

Consistencia conceptual

Este aspecto da información sobre el grado de adherencia a las reglas del modelo conceptual Se recomienda verificar que se cumplen ciertas reglas topológicas o geométricas con carácter global:

Solape de instancias: Control para garantizar que no existen instancias puntuales, lineales o superficiales del mismo fenómeno con la misma codificación de atributos y geometría parcialmente coincidente (más de un vértice).

Duplicidad de vértices: Control para garantizar que no hay vértices repetidos en una instancia.

Bucles: Control para garantizar que no hay bucles no deseados.

Continuidad entre hojas: Control para garantizar la conexión geométrica entre las instancias de una hoja y las instancias correspondientes de las hojas limítrofes.

Vértices superfluos: Control para garantizar que no hay vértices dentro de una primitiva lineal que subtiendan una flecha menor que 0,10 mm a escala respecto del segmento que une los vértices anterior y siguiente (algoritmo de Douglas-Peucker).

Resolución de anclajes: Control para garantizar de manera semiautomática que no existen extremos libres no deseados por subtrazo (underchoot) o sobretrazo (overshoot).

Además se aconseja realizar otros controles de carácter específico detallados en el catálogo de fenómenos:

Conectividad: Control para verificar que las conexiones entre fenómenos son correctas. Por ejemplo, curva de nivel y edificio no deben tener vértices coincidentes.

Conectividad 3D: Control para garantizar la conexión 3D entre los objetos que presenten este tipo de relación, coincidencia de coordenadas (x, y, H).

Conectividad 2D: Control para garantizar la conexión 2D entre los objetos que presenten este tipo de relación, coincidencia de coordenadas (x, y).

Cierre de recintos: Control para garantizar el cierre de las líneas que componen los objetos poligonales. Si está cortado por el marco de hoja, deberá quedar cerrado por una línea coincidente con él.

Líneas en recintos: Control para verificar que las proyecciones planas de las líneas clasificadas como eje o esquema son interiores a la proyección plana de las líneas que componen el recinto del fenómeno correspondiente.

Ejes en recintos: Control para verificar el cumplimiento de la relación "es_eje_de"

Esquemas en recintos: Control para garantizar la inclusión de las líneas esquema en la proyección horizontal de las líneas que componen el recinto del fenómeno correspondiente.

Orientación de líneas: Control para garantizar que aquellos elementos lineales que han de capturarse con una determinada orientación por razones altimétricas, como ríos y canales, o por razones topológicas, como bosques, taludes o curvas de nivel de depresión, tengan el sentido correcto.

Mínimos: Control para garantizar que no existen instancias de fenómenos cuya superficie o longitud sean inferiores a las descritas en el catálogo.

Fidelidad geométrica: Control para verificar que el aspecto visual del modelo representa la realidad, como la alineación de fachadas, la ortogonalidad de edificios, falta de algún vértice, altura constante de las masas de agua.

Resolución de intersecciones: Control para garantizar que siempre que se cortan dos primitivas geométricas, el punto de intersección se ha calculado y se ha incluido como vértice en cada una de ellas.

Clases del recinto: Control para garantizar que el polígono se forma con el tipo de líneas que permite el modelo de aplicación.

Continuidad semántica entre hojas: Control para garantizar la coincidencia de código entre las instancias de una hoja y las instancias correspondientes de las hojas limítrofes salvo que la fecha de actualización de las hojas sea diferente.

1.6.4 Exactitud temática

Este elemento de calidad describe el grado de fidelidad de los fenómenos o atributos en relación al valor correcto o al considerado como tal.

Corrección de la clasificación

La corrección semántica indica el porcentaje de códigos asignados correctamente. Para describir el grado de conformidad del conjunto de datos, se utilizará una muestra del conjunto de datos siguiendo el mismo criterio que para evaluar la compleción (véase 7.2). El resultado esperado es mayor o igual al 96%.

Corrección del nombre geográfico

La corrección de nombres geográficos indica hasta qué punto los nombres asociados a los fenómenos están libres de errores o no. Se describirá como el tanto por ciento de nombres correctos encontrados al comparar con otra fuente de datos más fiable, con fuentes oficiales, Nomenclátores oficiales, el Registro de Entidades Locales, o con trabajo de campo, sobre un

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

muestra de al menos el 10% de los nombres de la zona estudiada. El resultado se deberá mantener mayor o igual al 95%.

1.7 Metadatos

La norma establece la obligatoriedad de entregar los metadatos de la cartografía.

Los datos sobre los datos geográficos permiten a los usuarios su búsqueda, localización, comparación y utilización. El Consejo Superior Geográfico, a través de la Comisión de Geomática, ha establecido un conjunto mínimo de metadatos, definido como un perfil de ISO19115, que constituyen el Núcleo Español de Metadatos (NEM).

Como el producto que se define mediante estas especificaciones puede presentar diferencias sustanciales según sea el organismo productor, los metadatos serán la principal fuente de información para valorar estas diferencias.

Siguiendo las recomendaciones del documento NEM v1.0, cualquier conjunto de datos conforme a éstas especificaciones deberá acompañarse de un documento con información acerca de los metadatos y de los datos.

La norma ISO19115 contempla la posibilidad de asignar metadatos a varios niveles de detalle, desde el producto completo a las instancias de atributo. En esta primera versión del producto se aportan metadatos a nivel de conjunto de datos o a nivel de hoja por organismo productor.

1.8 Formato de transferencia

Para el intercambio de datos se recomienda la utilización del formato ESRI Shapefile, teniendo presente la intención de utilizar en un futuro el formato GML (Geographic Markup Language) tal y como lo define Open Geospatial Consortium, en la versión que apruebe la norma ISO19136.

Alternativamente, puede utilizarse cualquier otro formato propietario o abierto, acordado entre el proveedor y el usuario de los datos.

2 OBJETIVOS: DESARROLLO CON ArcObjects y VBA en ArcGIS Desktop

2.1. Marco Conceptual

2.1.1 Introducción

El presente documento constituye la memoria del Trabajo Final del Máster Universitario en Teledetección y Sistemas de Información Geográfica, organizado por la Universidad de Oviedo.

El mismo surge como una propuesta para automatizar tareas en proyectos de producción cartográfica que hayan de cumplir las normas de la "Base Topográfica Armonizada".

Este proyecto consiste en la creación y organización de comandos de edición y análisis dentro del entorno de ArcMap, agrupando una serie de funcionalidades creadas o preexistentes, en una nueva barra de herramientas.

En cuanto al desarrollo de la aplicación, la programación de barra se realiza íntegramente con VBA (Visual Basic for Aplications) a través del editor integrando en ArcMap, haciendo uso de librerías ArcObjects. Por tanto, se trata de una personalización de la interface a nivel interno, ya que el código ha sido generado dentro del propio entorno de ArcMap y almacenado en el proyecto activo.

Por otra parte, los controles que integran la aplicación se han organizado según su funcionalidad: A cada uno, se le asocia un código VBA, de tal manera que al ser pulsado se accede a una funcionalidad dentro de la vista de trabajo.

A continuación, se detalla el proceso llevado a cabo para la creación e implementación de la aplicación.

2.1.2 Objetivos

El principal objetivo del proyecto consiste en desarrollar una aplicación que facilite y optimice los procesos de edición de elementos gráficos dentro de ArcMap, de acuerdo a la

Base Topográfica Armonizada. Concretamente se trata detectar "puntos fugados", es decir, localizar todos los puntos que debido a algún error, su cota es superior a 3.000 m, igual a 0 o inferior a 0 m (cota negativa), exportarlos a una nueva capa para visualizarlos y posteriormente tratarlos. Hemos considerado cota errónea la superior a 3.000 m ya que en España solo se supera esa altitud en zonas muy concretas a identificables. Sin embargo, no es extraño encontrar lugares con cotas entre los 2.000 m – 3.000 m. En cualquier caso si por alguna necesidad se quiere cambiar esta configuración de búsqueda solo sería necesario editar tres líneas del código.

Asimismo hay que elaborar el diseño operativo y funciones específicas a implementar en la barra de herramientas.

Se integra la aplicación en ArcGis versión 9.3 mediante ArcObjects y Visual Basic for Applications (VBA).

El alcance de estos objetivos, implica la realización de una serie de tareas:

- Lectura y análisis de información previa.
- Estudio de otras aplicaciones existentes.
- Inventario de las herramientas de edición ya existentes en ArcMap.
- Definición de las posibles herramientas a desarrollar.
- Diseño de la estructura funcional de la aplicación.
- Diseño de la parte gráfica.
- Programación de las funcionalidades a implementar mediante Visual Basic for Applications, utilizando las librerías de ArcObjects.
- Integración de las funcionalidades de edición ya existentes (previamente seleccionadas) en ArcMap dentro de la nueva barra de herramientas.
- Almacenamiento de la aplicación en un proyecto de ArcMap (.mxd).

2.1.3 Aspectos conceptuales

Es necesario considerar algunos aspectos teóricos que resultaron fundamentales para el desarrollo de la aplicación:

Tecnología COM

Esta tecnología no trata de un lenguaje de programación orientado a objetos sino una forma de protocolos de comunicación entre diferentes componentes de aplicaciones, independientemente del lenguaje de programación que se utilice.

De esta forma es posible desarrollar componentes reutilizables e intercambiables. Esta tecnología sigue un modelo de programación orientado a interfaces.

ArcObjects

Es la plataforma de desarrollo para ArcGis Desktop. Concretamente es una colección de objetos que se manejan a través de código VBA que se asocia a los botones, herramientas y menús que los programadores crean para automatizar tareas en las diferentes aplicaciones dependientes de ArcGis (ArcMap, ArcCatalogo, etc).

ArcObjects se apoya en la tecnología COM de Microsoft.

Programación Orientada a Objetos (POO)

Es un tipo de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo y encapsulamiento.

Entre los lenguajes de programación que soportan la orientación a objetos, están: C#, Java, Visual Basic.Net, Visual Basic, Python.

3 PROCEDIMIENTO

3.1. Marco Metodológico

3.1.1 Fases del trabajo

La realización de este proyecto implicó varias etapas de desarrollo, las cuales se describen a continuación:

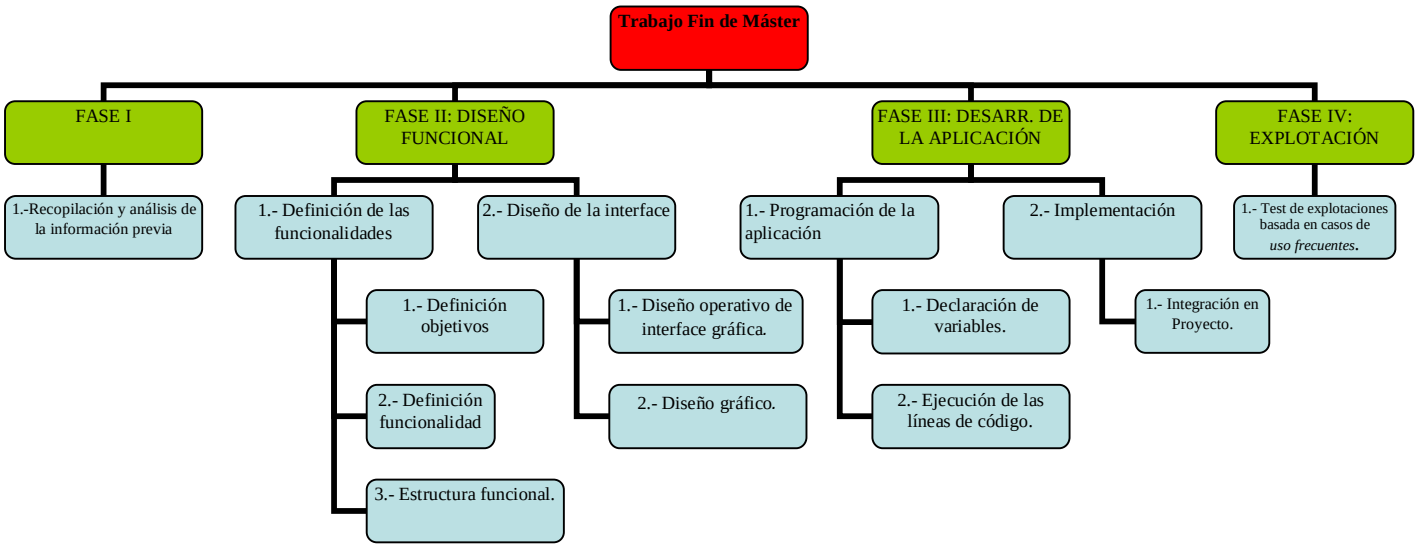

Figura nº 1. Etapas de desarrollo del trabajo.

3.1.1.1 Fase I : Recopilación y análisis de la información disponible

La primera fase consistió en analizar los datos de partida disponibles y las posibles utilidades del proyecto.

A partir de este diagnóstico se realizó la conceptualización del mismo y se plantearon los objetivos y las tareas a realizar para dar cumplimiento a los mismos.

3.1.1.2 Fase II: Diseño Funcional

En esta fase se seleccionaron las herramientas de desarrollo que hicieron posible la realización del proyecto, se definieron las funcionalidades de la aplicación y la serie de tareas a programar.

Perfil del usuario:

El usuario final de la aplicación será el técnico del departamento de Cartografía en una empresa que emplea software SIG, para su producción.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Lenguaje de programación

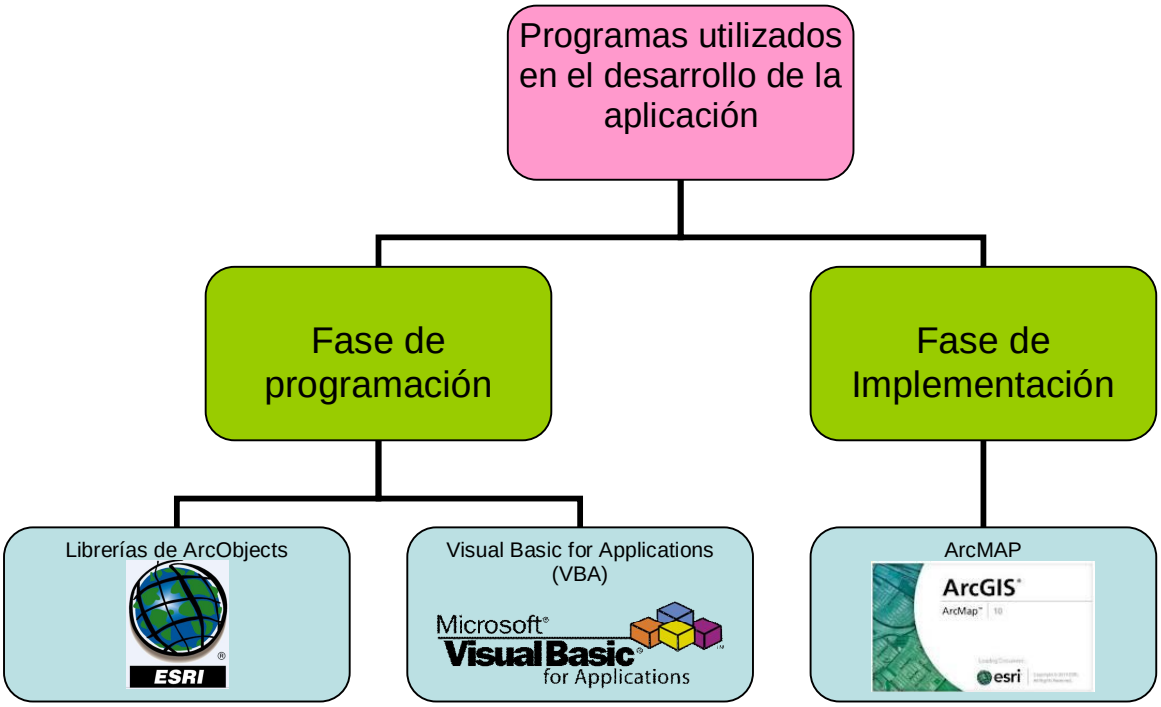

Figura nº 2.Programas utilizados en el desarrollo de la aplicación.

La programación de barra de herramientas se realizó íntegramente con VBA (Visual Basic for Aplications), utilizando el conjunto de componentes y objetos de ArcObjects.

VBA constituyó el lenguaje de programación y el entorno de desarrollo al mismo tiempo.

VBA, está formado por un subconjunto de Visual Basic y es muy utilizado en la programación de "Macros" en aplicaciones de Windows para la automatización de tareas cotidianas y el incremento de las capacidades de estas aplicaciones.

Este lenguaje permite la programación de eventos y disfruta de algunas de las funcionalidades de un lenguaje orientado a objetos. Todas estas características han hecho que ESRI haya seleccionado este lenguaje para el desarrollo de aplicaciones dentro de ArcGIS.

El desarrollo de una aplicación en VBA a nivel interno se lleva a cabo desde el entorno de desarrollo que ArcGIS proporciona, el editor de VBA, un entorno de desarrollo muy similar al de VB y desde el cual se crearon los formularios, se añadieron controles y se escribió el código asociado a estos elementos.

Este editor es accesible para el caso de ArcMap desde la opción:
Tools >Macros >Visual Basic Editor.

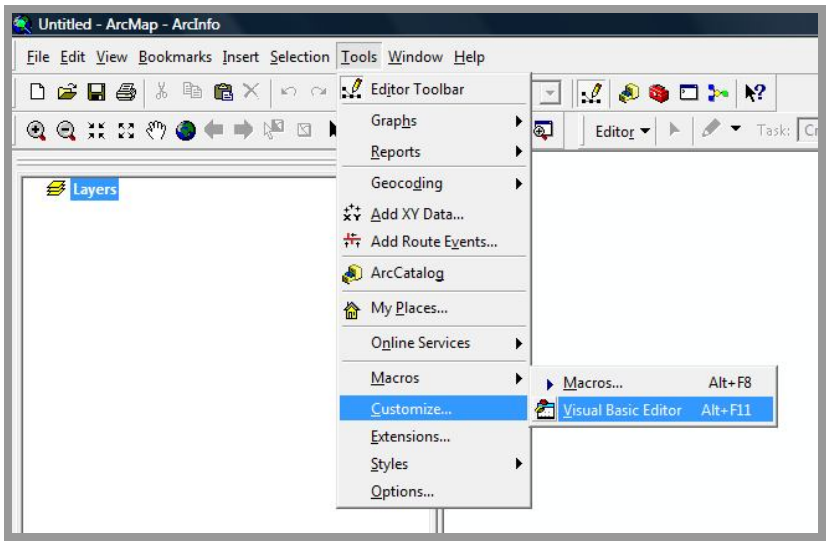

Figura nº 3. Acceso a editos de VBA desde ArcMAP.

En cuanto al uso de ArcObjects, este constituye el conjunto de herramientas y funcionalidades que permitió desarrollar una aplicación SIG personalizada y extensión de la funcionalidad de ArcGIS, al proporcionar la infraestructura para su desarrollo.

La combinación ArcObjects/VBA es una buena opción cuando se quiere desarrollar aplicaciones que se ejecutan en el entorno de ArcGIS Desktop. Para la realización de este proyecto, se ha utilizado esta opción.

Requisitos Funcionales

Una vez precisadas las herramientas de desarrollo, se definieron las funcionalidades de la aplicación, el diseño operativo y la serie de funciones a programar:

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Formatos soportados:

Shapefile, ya que este es el formato que habitualmente utiliza la oficina SIG para la elaboración de su cartografía digital y el almacenamiento de información espacial.

Funcionalidades:

En cuanto a las funcionalidades la aplicación, la misma cuenta con funciones de análisis y visualización de los elementos gráficos.

- Análisis:
  - Detección.
  - Contabilización de vértices fugados.
  - Salida a fichero ".txt" de errores.

- Visualización:
  - Pasar vértices a Capa.
  - Seleccionar vértices fugados.
  - Seleccionar capa de errores.
  - Descargar capa de errores.

3.1.1.3 Fase III: Desarrollo de la aplicación

Esta fase consistió en la realización de una serie de rutinas de programación para desarrollar la aplicación.

La programación como se ha mencionado anteriormente se baso en la tecnología de ArcObjects, componentes de programación con funcionalidades SIG e interfaces programables mediante las cuales han sido desarrollado ArcGIS y sus aplicaciones. En este caso específicamente se trabajó sobre ArcMap.

La realización de la personalización de ArcGIS con ArcObjects, se realizó a través de Visual Basic para aplicaciones (VBA) aunque es posible utilizar otros lenguajes que cumpla con las especificaciones COM (Component Object Model).

No obstante, fue elegido VBA por ser la forma más común que los desarrolladores utilizan para personalizar ArcGIS, básicamente por tratarse de un lenguaje menos complejo.

Estrategia de personalización

El primer paso a realizar durante esta fase fue definir la estrategia de personalización, considerando las diversas posibilidades de programación para la versión 9.3 de ArcGIS.

Teniendo las siguientes opciones:

Programación dentro de ArcGIS:

Como se ha mencionado anteriormente dentro de ArcMap y ArcCatalogo se encuentra un entorno de desarrollo en VBA donde podemos manejar los objetos de ArcObjects; el modelo de objetos que gestiona el comportamiento de cada una de los componentes que forman estas aplicaciones.

Los macros escritos en este entorno no funcionan fuera de él, ya que son dependientes de la aplicación y necesitan que la misma éste abierta.

Programación fuera de ArcGIS:

Al ser ArcObjects un conjunto de objetos COM, se pueden utilizar objetos programables o sus librerías dentro del entorno de desarrollo de otros lenguajes de programación, como VB, C ++, .Net o Phyton.

Para estos casos se desarrollan aplicaciones independientes, por lo que no hace falta que ArcMap esté abierto pero sí que éste instalado ArcGIS en la máquina donde se ejecuta el programa.

Además de estas opciones, existen unos niveles de personalización atendiendo a distintas cotas de complejidad, los cuales son:

Nivel Básico:

El cual se restringe a organizar la interfaz gráfica de usuario, creando nueva barras de herramientas, botones o menús y asociándole a estos comandos ya existentes en ArcMap. Dentro de este nivel no es necesario programar.

Nivel avanzado interno:

Además de construir nuevas barras de herramientas, botones o menús, podemos asociar a estos códigos escritos en VBA que se ejecutará sobre el comando correspondiente. El código generado se programa dentro del entorno de ArcMap y se guarda en el proyecto activo .mxd.

Nivel avanzado externo:

El código de las rutinas creadas puede ser escrito en cualquier lenguaje que soporte COM, y crear aplicaciones independientes de ArcMap.

Analizando las ventajas e inconvenientes, así como la complejidad de las distintas opciones, se seleccionó como estrategia: Una programación dentro de ArcGIS, con un nivel de personalización básico, se organizó en una nueva barra de herramientas algunos de los comandos ya existentes en ArcMap, para de esta manera aprovechar su macro.

Sin embargo, también se avanzó hasta el segundo nivel (avanzado interno) ya que se incorporaron nuevos comandos a los cuales desde la ventana del editor se les asoció el código respectivo.

Programación- Estructura del código

Una vez definidos el tipo de personalización se inició el proceso de programación en el editor de Visual Basic:

- Definición de objetos y propiedades.
- Declaración de variables.
- Almacenaje y ejecución del código en un Map Document.

La estructura y organización del código se describe a continuación:

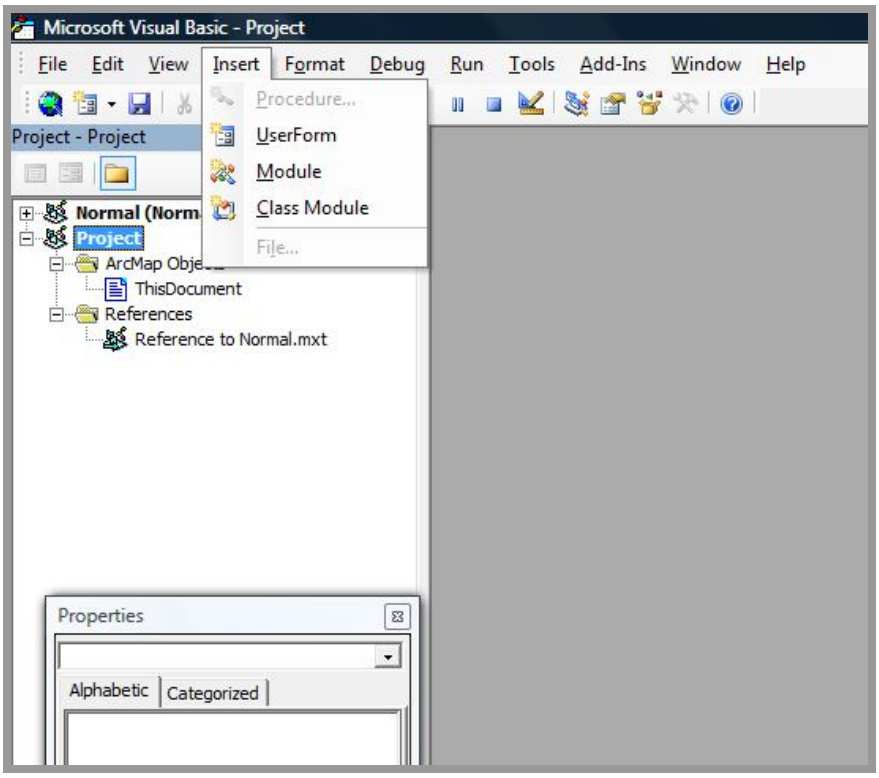

Figura nº 4. Explorador de proyectos del editor VBA.

Programación de procedimientos y formularios:

La aplicación está compuesta de 5 formularios, para los cuales se realizó la programación de los acontecimientos, a fin de que el usuario interactué con estos.

FrmDeteccptosfugados

FrmZmayor3000

FrmZ0

FrmZfugados

FrmZnegativa

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

3.1.1.4 Fase IV : Implementación/Test de explotaciones

En cuanto al almacenamiento de la aplicación, cuando se trabaja a nivel básico o avanzado interno como en este caso, existen varias opciones de distribución de la aplicación.

La primera opción es proporcionar el archivo .mxd con el código incluido:

Esto es posible al realizar el almacenamiento de la información en el documento actual. El documento (extensión. Mxd) se refiere al proyecto que este activo en un momento determinado en ArcMap. Cualquier modificación de los ajustes o personalización puede ser guardada en el documento si lo indicamos de forma explícita, bien sea desde la ventana de personalización,

Tools > Customize >Commands > Save in,

o bien desde el editor,

Tools> Macros >Visual Basic Editor.

Esta forma de almacenamiento permite que las modificaciones solo sean visibles cuando se habrá el documento en concreto.

La segunda opción es el almacenamiento de la personalización en plantillas: Al abrir una sección de ArcMap o de ArcCatalog por defecto todas las modificaciones serán guardadas en una plantilla llamada normal.mxt, por lo que si se desea que la aplicación VBA esté disponible para cualquier sección de ArcMap, solo es necesario escribir el código en el área de la ventana de proyecto reservada para la plantilla.

Para el desarrollo de la aplicación se utilizó la primera opción de almacenamiento. La personalización se almacenó en un archivo .mxd al que llamamos "Tfm" (Trabajo Fin de Máster), siendo posible transportar la misma a otros ordenadores.

Una vez realizado el proceso de programación y definido el tipo de almacenamiento se procedió a la realización de pruebas o test de explotaciones a fin de examinar las funcionalidades implantadas y su operatividad.

3.1.1.5 Fase V: Resultados y conclusiones

Una vez realizados la implementación de la aplicación y realizado los respectivos ensayos en la fase experimental a través de los test de explotación, se describieron los resultados y se formularon las conclusiones finales más relevantes en cuanto a operatividad, líneas futuras de trabajo y a nivel personal.

3.1.2 Plan de trabajo

El marco temporal en el que se desarrolló este proyecto fue un período de 12 semanas: Del 12 de marzo al 3 de junio del año 2012.

Con el propósito de organizar las tareas a desarrollar durante ese período y realizar un adecuado seguimiento y control del proyecto, se estableció un plan de trabajo en el que se delimitaron temporalmente cada una de las actividades a realizar en las distintas fases del mismo.

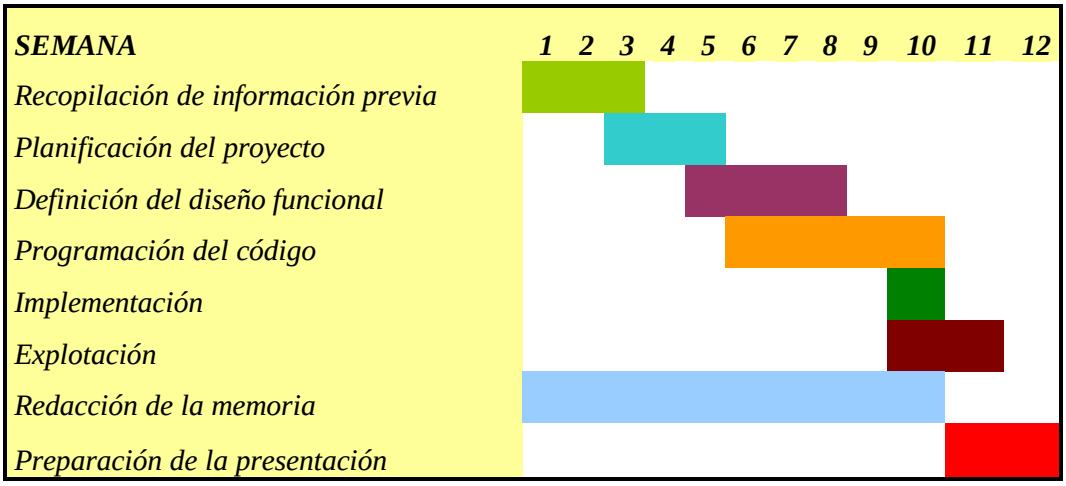

Figura nº 5. Plan de trabajo.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

3.2. Diseño funcional

3.2.1 Creación de herramientas y comandos a programar.

Se ha creado una barra de herramientas llamada "DetPuntos Fugados".

Las diferentes funcionalidades que se desarrollaron para ser integradas en la aplicación, se organizan en grupo según su funcionalidad.

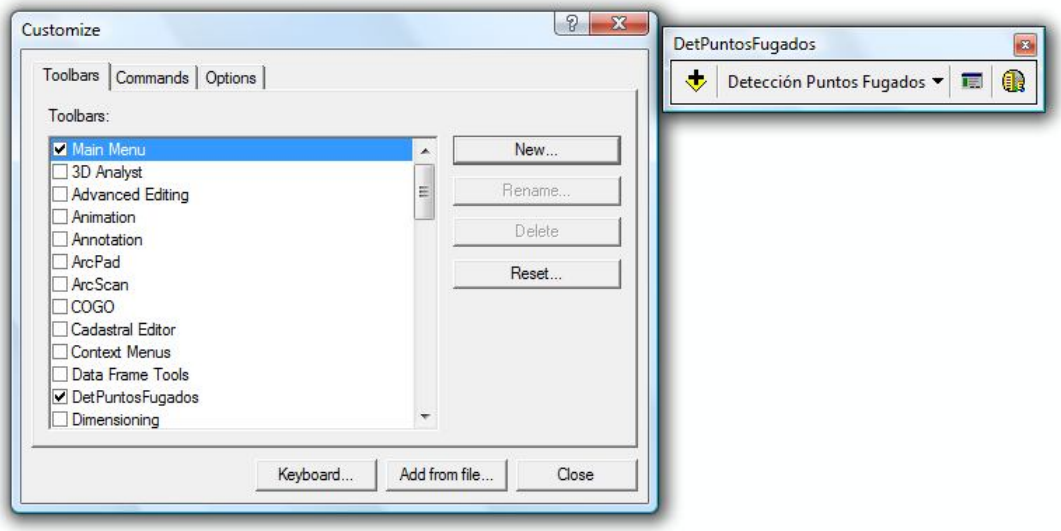

Figura nº 6. Barra de herramientas "DetPuntos Fugados".

3.2.2 Diseño operativo y gráfico de la aplicación

El diseño de la barra fue realizado considerando las funciones que llevan a cabo las herramientas, las cuales podemos clasificarlas en 3 grupos:

- Comando de carga de "layers".
- Menú "Detección puntos fugados" y Comando "Detección puntos fugados".
- Comando de acceso a "ArcScene".

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

En cuanto a la operatividad la barra, la misma está integrada dentro de ArcMap. Por otra parte, al pulsar sobre sus controles se producirán las respectivas funcionalidades que fueron previamente programadas a través de código.

A continuación se muestra de manera gráfica el diseño grafico y operativo de la aplicación:

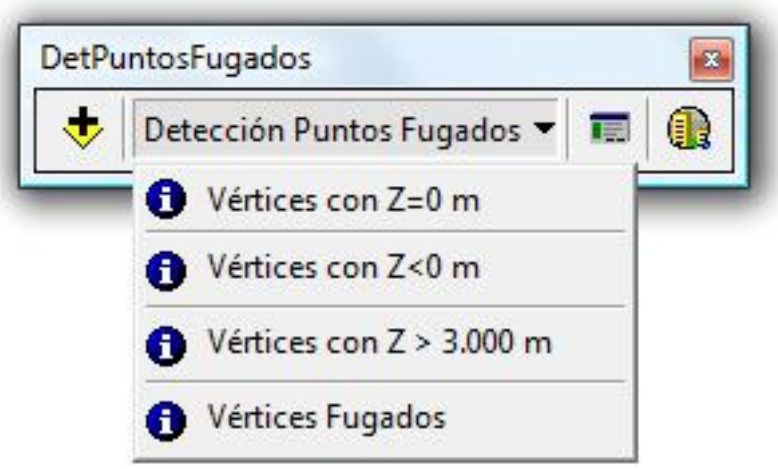

Figura nº 7. Diseño gráfico de la barra de herramientas.

Tal como se aprecia en la figura anterior la barra de herramientas la conforman una serie de controles que se encuentran integrados en menús, que a su vez se dividen en submenús.

A continuación, se desarrolla cada uno de los menús- submenús, cuyas funcionalidades han sido descritas previamente:

-  Comando de carga de "layers".

Permite acceder a las carpetas desde las que queremos cargar la información a analizar.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

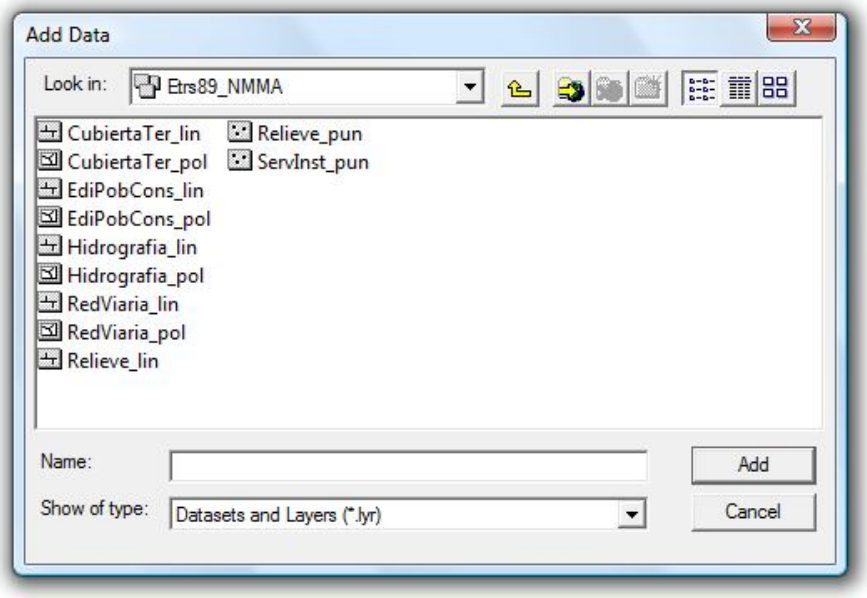

Figura nº 8. Comando de carga de "layers".

- Menú "Detección puntos fugados" y Comando "Detección puntos fugados".

Permite acceder a los 4 formularios de análisis en función de lo que se necesite comprobar:

- Vértices con Z=0 m
- Vértices con Z<0 m
- Vértices con Z>3.000 m

O bien ejecutar estas 3 condiciones a la vez, con:

- Vértices Fugados.

Hay que destacar en este punto, que estas condiciones de análisis definidas para este trabajo, pueden ser adaptadas a otras circunstancias, con una modificación muy simple del código.

-  Comando "Detección puntos fugados".

Igualmente permite acceder a los 4 formularios de análisis en función de lo que se necesite comprobar:

- Vértices con Z=0 m
- Vértices con Z<0 m
- Vértices con Z>3.000 m

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

O bien ejecutar estas 3 condiciones a la vez, con:

- Vértices Fugados.

-  Comando "ArcScene".

Permite acceder a ArcScene, el visor donde se podrán observar en 3D los puntos fugados.

3.3. Desarrollo del la aplicación.

3.3.1 Proceso de programación

Este trabajo emplea los principios de la Programación Orientada a Objetos (POO), un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas.

El elemento fundamental de la POO es el objeto, el cual puede ser definido como un conjunto complejo de datos, que en su interior contiene cierto número de componentes bien estructurados. Éste objeto forma parte de una organización jerárquica o de otro tipo y pertenece a una clase.

El objeto puede dividirse en tres partes:

- Propiedades: Son las que distinguen a un objeto del resto que forma parte de la organización, pudiendo ser estas propias o heredadas.

- Métodos: Son las operaciones que pueden realizarse sobre el objeto. Un método, también llamado comportamiento, realiza una acción específica.

- Relaciones: Permiten que el objeto se inserte en la organización y están formadas principalmente por punteros a otros objetos.

3.3.1.1 Programación en ArcObjects

ArcObjects es un marco que le permite crear dominios específicos de los componentes de otros componentes. Proporciona una infraestructura para la aplicación de personalizaciones, con la finalidad de que por medio de estas pueda ajustarse a las necesidades específicas de sus clientes.

Todas las aplicaciones dependientes de ArcGIS se desarrollan en base a los objetos de ArcObjects, ya que cada componente de éste tiene su correspondencia con una clase de ArcObjects.

Como se ha mencionado anteriormente existen distintos escenarios de programación en ArcObjects, los cuales van desde realizar aplicaciones dependientes de ArcGIS en colaboración con otras aplicaciones COM o totalmente independientes mediante ArcGIS Engine.

En este caso, se trabajo dentro de ArcGIS creando la aplicación desktop sobre ArcMap, utilizando VBA.

3.3.1.2 El modelo de objetos en ArcObjects

ArcObjects se compone de un modelo de datos geográficos orientado a objetos basado en las especificaciones COM (Component Object Model).

El proceso de programación se apoya en los diagramas de modelos de objetos también conocidos como DMO.

Estos representan una colección organizada de objetos y clases de objetos que marcan las relaciones entre las diferentes clases de objetos (mediante símbolos). Además describen las propiedades y métodos que pueden utilizarse con cada una de estas clases e indican como navegar a través de todo el conjunto de objetos.

Los DMO se basan en especificaciones UML o lenguaje unificado de modelado.

ArcObjects se compone de objetos y clases:

Un objeto representa una característica de tipo espacial como podría una carretera. Es una instancia de una clase, la cual es una serie de código que define a todos los elementos relacionados con ella.

Una clase por su parte, es un conjunto de objetos con similares atributos.

Para programar en ArcObjects es necesario conocer los siguientes fundamentos:

Las clases están almacenadas en archivos de código formando librerías.

Los objetos se crean a partir de las clases y se almacenan en la memoria. La clases tienen interfaces de programación formadas por grupos de propiedades y métodos.

Tipos de clases:

- Coclase:

Una coclase se puede utilizar para crear nuevos objetos. Un FeatureClass es una coclase que permite que nuevas características de la clase puedan ser como instancias de la coclase.

- Clase abstracta:

Una clase abstracta no se puede usar para crear nuevos objetos, sino que existe para que otras clases (es decir, las subclases) puedan usar o compartir la propiedades y métodos que la clase soporte. Por ejemplo, GeoDataset es una clase abstracta.

- Clase:

Una clase no se puede utilizar directamente para crear nuevos objetos; en su lugar, los objetos de una clase sólo pueden ser creados a partir de otra clase.

Las clases (definiciones de las propiedades y comportamiento de un tipo de objeto concreto) y las interfaces son a menudo denominadas el "qué" y el "cómo" de la COM. La interfaz define lo que un objeto puede hacer, mientras que la clase define el cómo se hace.

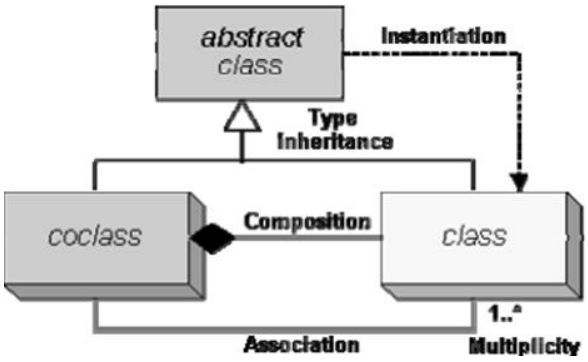

Figura nº 9. Tipos de clase en el modelo de objetos.

3.3.1.3 El uso de las interfaces

La programación bajo las especificaciones COM implican trabajar con interfaces. Las interfaces son una agrupación lógica de métodos y propiedades dentro de una clase. De esta forma, para comunicarse con un objeto de una clase determinada (crearlo o instanciarlo) es necesario indicar hacia qué interfaz se apunta, es decir, qué métodos o propiedades del objeto se quieren utilizar.

Por lo antes expuesto, cuando programamos con objetos en ArcObjects, no se trabaja directamente con el objeto, sino que se accede a él a través de una de sus interfaces.

Las interfaces en sí son de naturaleza abstracta, ya que no implementan código alguno, sino que sólo almacenan la estructura de las propiedades y métodos disponibles. La implementación del código se realiza a nivel de la clase que implementa dichas interfaces.

En este sentido varias clases pueden implementar (heredar) la misma interfaz, pero desarrollar un código diferente para la misma propiedad o método. A esto último es lo que se conoce como polimorfismo y es una de las características importantes de la programación orientada a objeto. En otras palabras, la interfaz decide qué puede hacer un objeto mientras que la clase decide cómo lo hace.

Una interfaz no contiene código, sino un listado con la definición de los métodos y propiedades.

Un objeto puede apoyar a dos o más interfaces y, además, el mismo objeto puede heredar interfaces de su superclase. Habida cuenta de múltiples interfaces, es posible para acceder a una interface a través de otro interfaz, o para saltar de una interfaz a otra.

Las principales interfaces utilizadas: IMap, Ilayer, IFeatureLayer, IFeature, Ielement,Ilayer,ItopologicalOperator,IDispacth.

3.3.2 Personalización de la interfaz de ArcMap

Todas las tareas básicas de creación o modificación de las barras de herramientas, botones o menús pueden controlarse desde la ventana de personalización (Customize).

En esta ventana aparecen tres pestañas:

- ToolBars (Barra de herramientas),desde la que se puede abrir o cerrar las barras de herramientas existentes o crear nuevas.

- Commands (Commandos), donde pueden seleccionarse herramientas ya existentes para añadirlas a las barras de herramientas.

- Options (Opciones) donde puede accederse a ciertas opciones de configuración.

3.3.2.1 Creación de la barra de herramientas

La creación de la barra de herramientas se realizó a través de la caja de dialogo de personalización de ArcMap, a la cual se accede desde,

Tools > Customize.

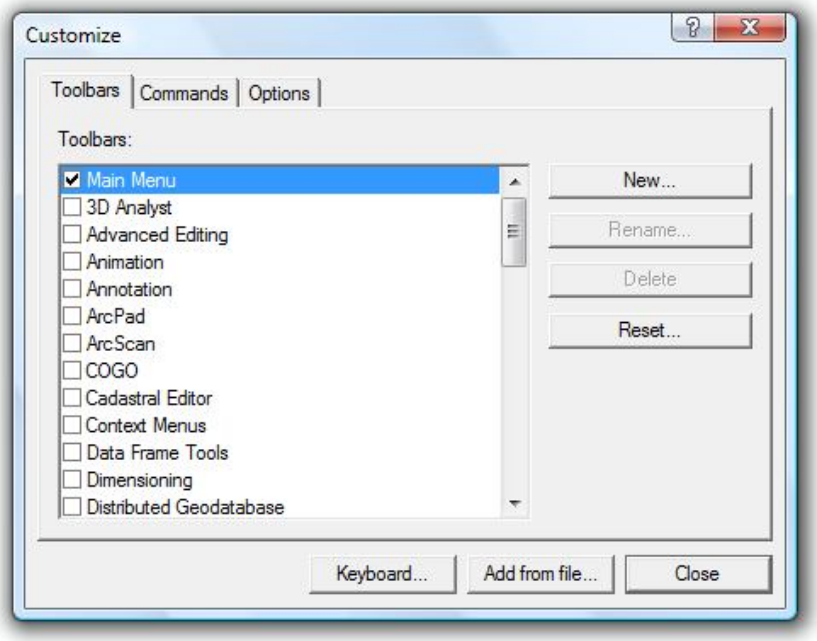

Figura nº 10. Ventana de personalización "Customize".

Esta ventana permite crear una nueva barra, insertar menús y añadir o borrar controles ya existentes en la aplicación o bien modificar sus propiedades sin ser necesario escribir código.

La pestaña ToolsBar (Barras de herramientas) muestra todas las barras de herramientas disponibles en ArcMap.

Por medio de esta ventana se creó la barra de herramientas en la que se organizaron de manera conjunta los comandos.

El procedimiento utilizado se describe a continuación:

1.- Se selecciona la opción personalizar (Customize) en el menú Herramientas (Tools) en ArcMap.

2.- Una vez desplegado el cuadro de dialogo en la pestaña Toolbars se selecciona la opción New para insertar una nueva barra. Luego introducimos el nombre de la barra y salvamos los cambios, en este caso sobre el fichero Tfm.mxd.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Una vez realizado este procedimiento una nueva barra de herramientas aparece ahora en ArcMap.

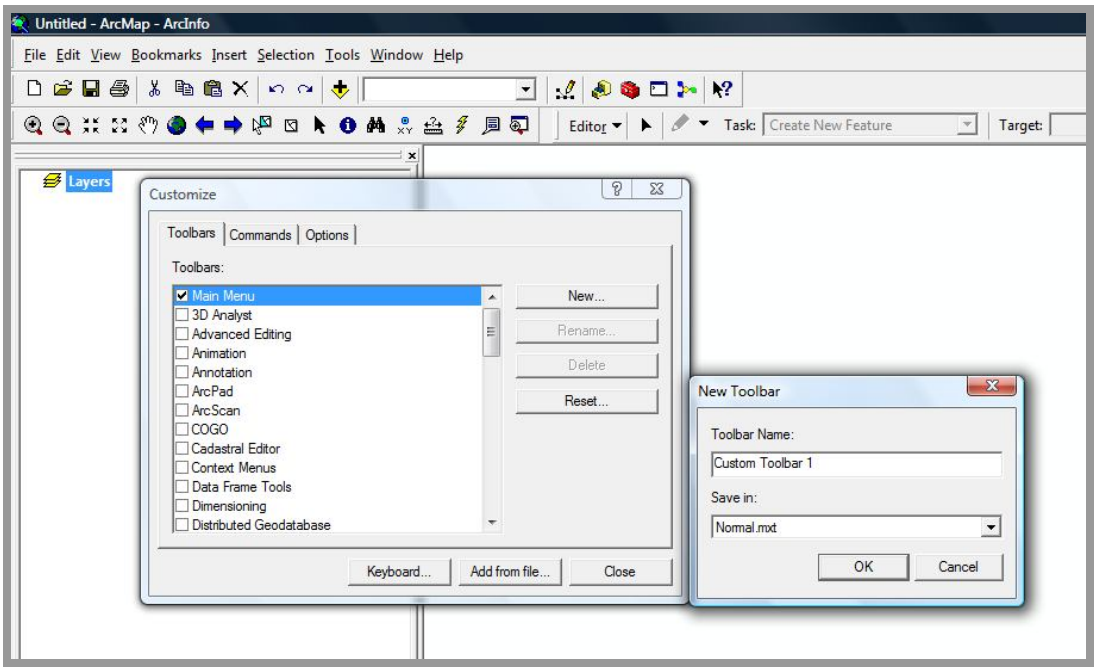

Figura nº 11. Pestaña "Toolbar", creación de una nueva barra de herramientas.

3.3.2.2 Inserción de controles de usuario

Una vez creada la barra de herramientas según el procedimiento descrito anteriormente, el siguiente paso fue insertar los controles de usuarios.

Los controles de usuario son los objetos que se añadieron a la nueva barra de herramientas para interactuar con la misma.

Existen cuatro tipos de controles: Botones, herramientas, listas desplegables y cajas de textos, a los que se accede desde la pestaña Commands de la ventana de personalización: Customize.

Para aplicación se han utilizado los controles de tipo [Menus] y [UIControls]-UIButtonControls. Los botones tienen asociado código que se ejecutará nada más pulsar sobre el botón.

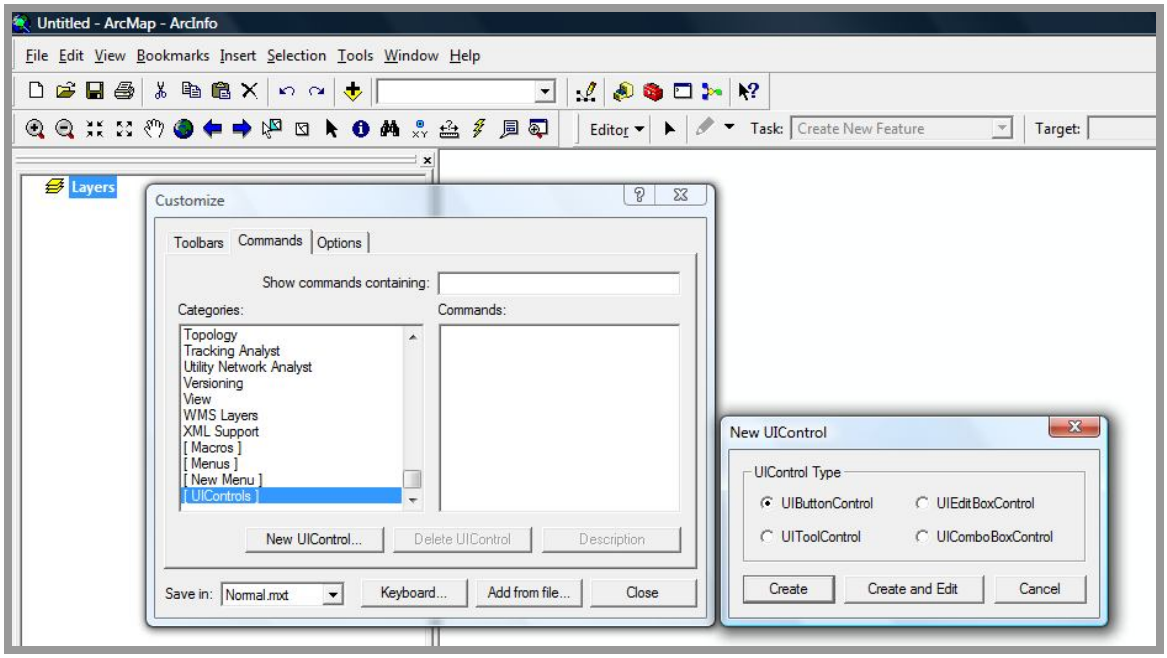

Figura nº 12. Pestaña "Commands", tipos de controles de usuarios en ArcObjects.

Para crear nuevos controles de usuario se seleccionó la opción [UIControls] en la caja de categorías. Al seleccionar esta opción, el botón New UIcontrol nos permitió la creación de un nuevo control.

El proceso general que se ha empleado para la integración de los controles a la barra se describe a continuación:

- Selección de la pestaña de comandos y desplazándose hasta la categoría [UIControls].
    - Pulsando sobre New UIControl se accede a los controles de usuario.
    - Se selecciona un tipo control de control de acuerdo a la funcionalidad de este.
    - Se pulsa sobre Create y el nuevo control es añadido a la lista Commands. Luego se selecciona y arrastra sobre la nueva barra de herramientas anteriormente creada.
    - Con el botón derecho del ratón, se pulsa sobre el nuevo control para acceder a sus propiedades.
    - Luego se accede a la opción Change Button Image para cambiar el icono asociado del control respectivo.

De la misma manera con el botón derecho del ratón, pulsando sobre el nuevo control se accede a la opción View Source, que despliega el editor VBA y la ventana de edición. En ella se escribió el procedimiento de evento y una función respectivamente.

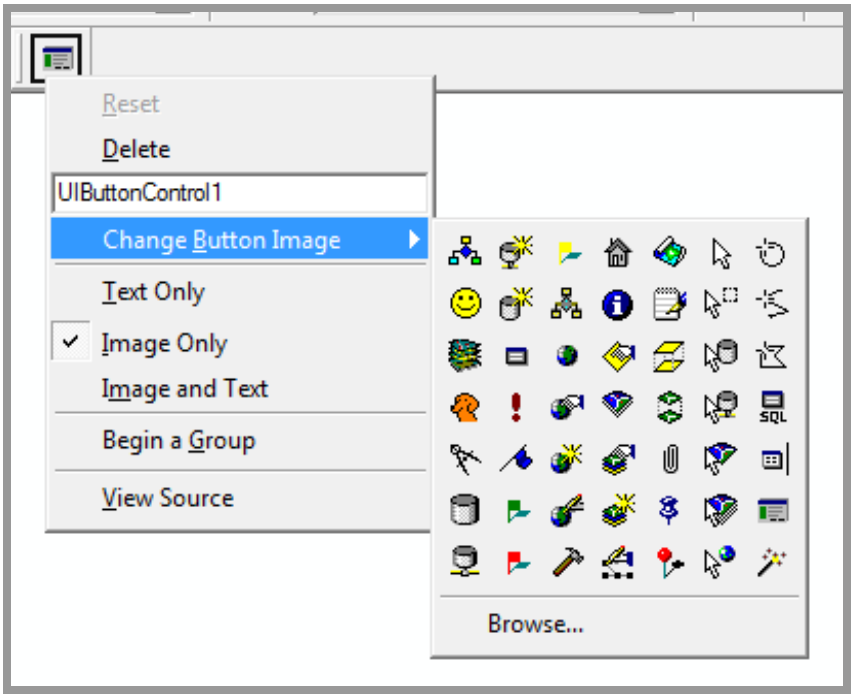

Figura nº 13. Procedimiento para acceder a las propiedades de un botón.

3.3.3 Estructura y organización del código

La aplicación está conformada por un proyecto que a su vez contiene módulos, procedimientos y formularios.

A continuación se expone las tareas generales realizadas durante la fase de programación y la organización del código que compone la aplicación:

1.- Se elige dónde escribir el código, que para este caso fue en el documento activo Tfm.mxd.

2.- Se insertan los formularios.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

3.- Se añaden los controles a los distintos formularios.

4.- Se añade el código asociado a los eventos de cada control en el módulo de formulario correspondiente.

5.- Se insertan módulos estándar si es necesario.

6.- Se ejecuta el formulario.

Procedimientos

Los procedimientos son todos aquellos bloques de sentencia de código (Limitadas por una cabecera y un pie) destinas a realizar tareas especificas dentro de la aplicación. Estos procedimientos (que en la aplicación se denominan "Macros"), se agruparon dentro de módulos, por lo que cada modulo viene a ser una colección de procedimientos.

Módulos

Este proyecto consta 5 módulos de formulario.

En "This Document" se encuentra almacenado todo el código que hace referencia al proyecto y que interactúa directamente con la barra de herramientas.

En los módulos estándar se incluye el código almacenado en procedimientos o funciones genéricas.

Módulos de formulario

Cada formulario generado tiene asociado su propio modulo de código.

Desde la ventana de dicho modulo se accede a todos los controles que están insertados en el formulario y todos los eventos de cada control.

4 RESULTADOS

4.1 Implementación

4.1.1 Activación de la aplicación

La nueva barra de herramientas es visible al abrir el documento Tfm.mxd.

En caso que el usuario no la encontrase o desease cerrarla puede hacerlo desde menú, View> Toolbars>.

4.1.2 Test de Explotaciones

Se han realizado pruebas para verificar el correcto funcionamiento de los distintos controles que se han integrado en la barra de herramienta, a fin de verificar su operatividad y realizar un control de calidad.

A continuación se reseña e ilustra a su vez el proceso seguido para comprobar las funcionalidades incorporadas. Cabe destacar que estas pruebas fueron realizadas únicamente para las nuevas herramientas programadas:

4.1.2.1 Carga de la información a analizar.

Antes de comenzar a usar la herramienta, se debe configurar la ruta de acceso donde se desea guardar los ficheros de texto ".txt" que serán generados por los formularios "FrmZ0", "FrmZnegativa", "FrmZmayor3000" y "FrmZfugados".

Al empezar a usar la herramienta, en primer lugar hay que agregar aquellas capas de elementos sobre las que se desea trabajar. En este caso se utilizan como muestra, 3 hojas de la Cartografía de Cantabria producida según la BTA, concretamente:

Cantabria_0056_87, Cantabria_0058_31 y Cantabria_0058_63.

Para ello accedemos a dichas geodatabases desde el primer comando de la aplicación:

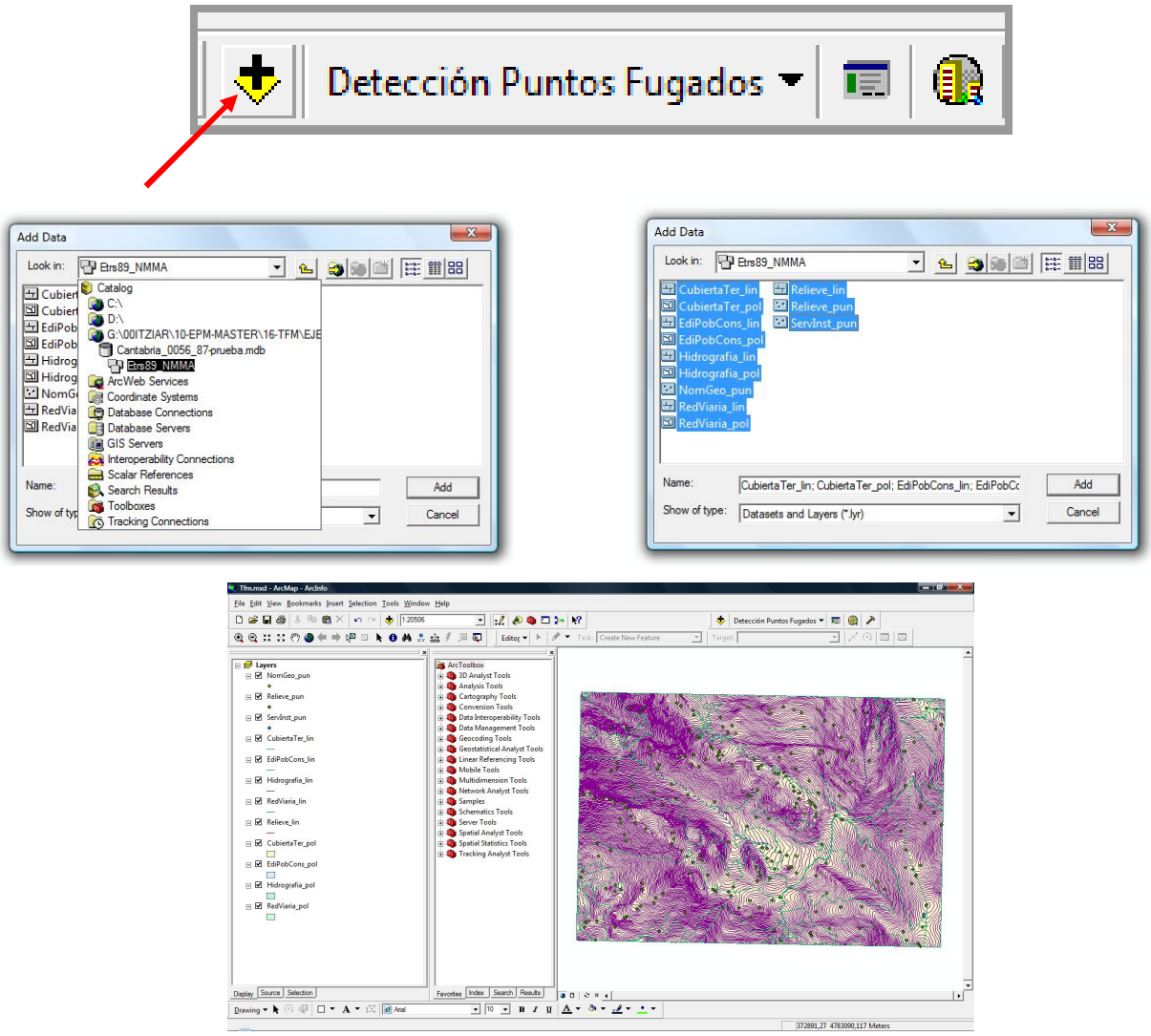

Figura nº 14. Paso Inicial para realizar los test de explotaciones-Agregación de capas.

4.1.2.2 Menú "Detección Puntos Fugados".

Al hacer clic sobre el Menú "Detección Puntos Fugados", se despliegan 4 opciones que permiten elegir los parámetros según se van a analizar los "vértices fugados":

Vértices con cota igual a cero (Z=0 m), vértices con cota negativa (Z<0 m), vértices con cota superior a 3.000 m (Z>3.000 m), o todas las opciones anteriores a la vez en "vértices fugados".

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

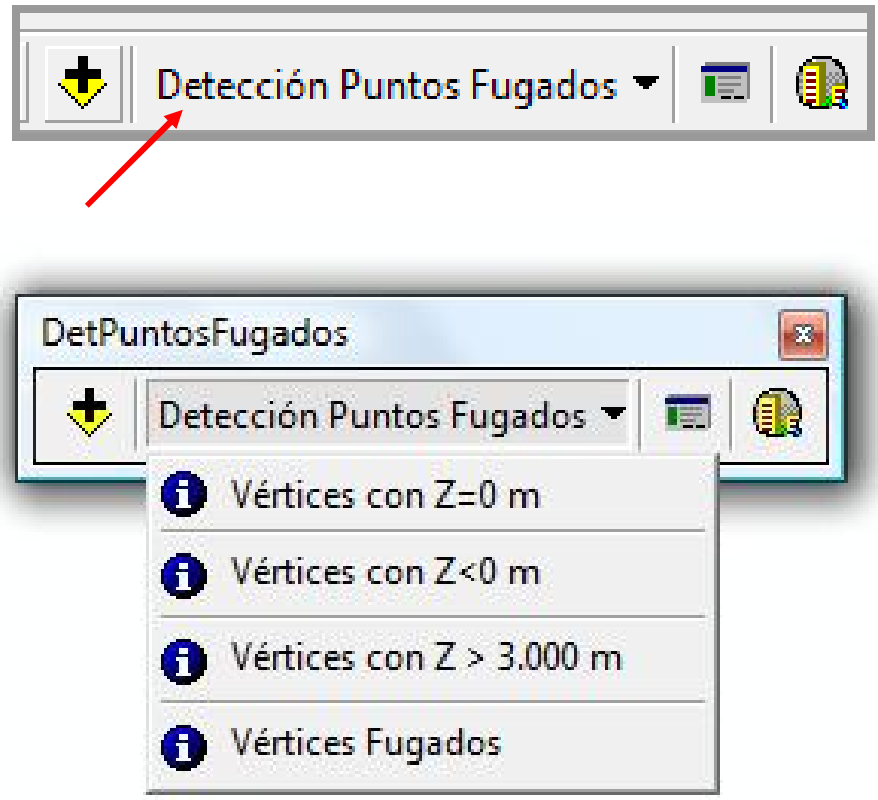

Figura nº 15. Menú "Detección Puntos Fugados".

Según se elija una opción, se accederá al formulario correspondiente. Cabe reiterar aquí de nuevo, que estas condiciones de análisis definidas para este trabajo, pueden ser adaptadas a otras circunstancias, con una modificación muy simple del código.

Otra forma de acceso a estos mismos formularios es a través del comando "Detección Puntos Fugados", el cual da acceso a un formulario, donde de nuevo podemos elegir entre las 4 opciones de parámetros, para analizar los "puntos fugados":

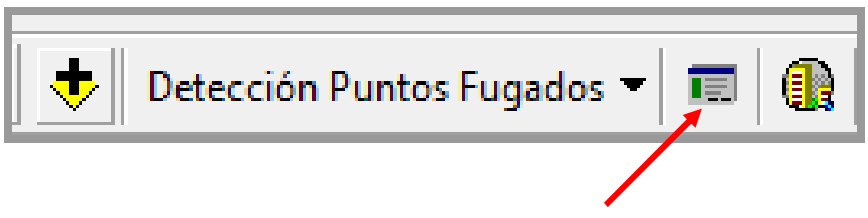

Figura nº 16. Comando "Detección Puntos Fugados".

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

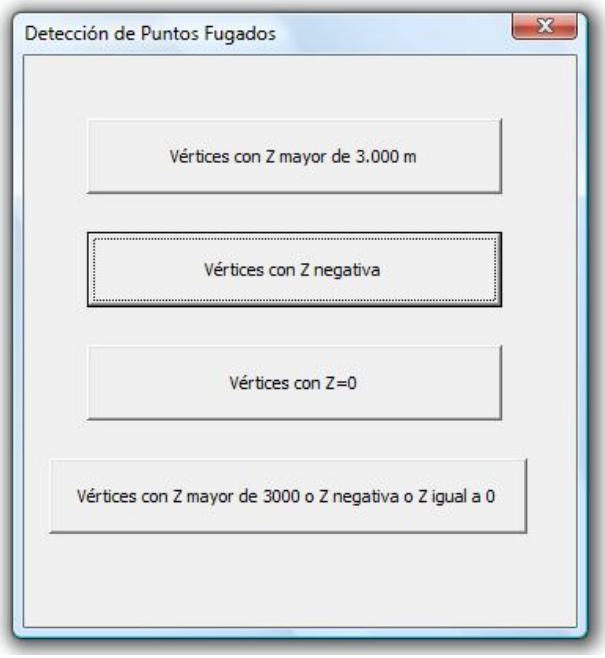

Figura nº 17. Formulario de acceso a opciones de análisis de "vértices Fugados".

4.1.2.3 Menú "Detección Vértices Fugados".

Independientemente de cómo se acceda a los "Formularios de análisis de "Vértices Fugados", dentro de cada uno es posible llevar a cabo 7 pasos ordenadamente (3 de análisis y 4 de visualización), aunque los mínimos necesarios para llegar al objetivo, son los 3 primeros. Cada paso se ejecuta con su comando correspondiente.

Análisis:

Paso 1.- "Detectar".

Localiza en todas las capas cargadas, los vértices que cumplan la condición impuesta de "vértice fugado". Se identifican y en los correspondientes "listbox" se muestran sus propiedades:

- Layer.
- OID
- X
- Y
- Z

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

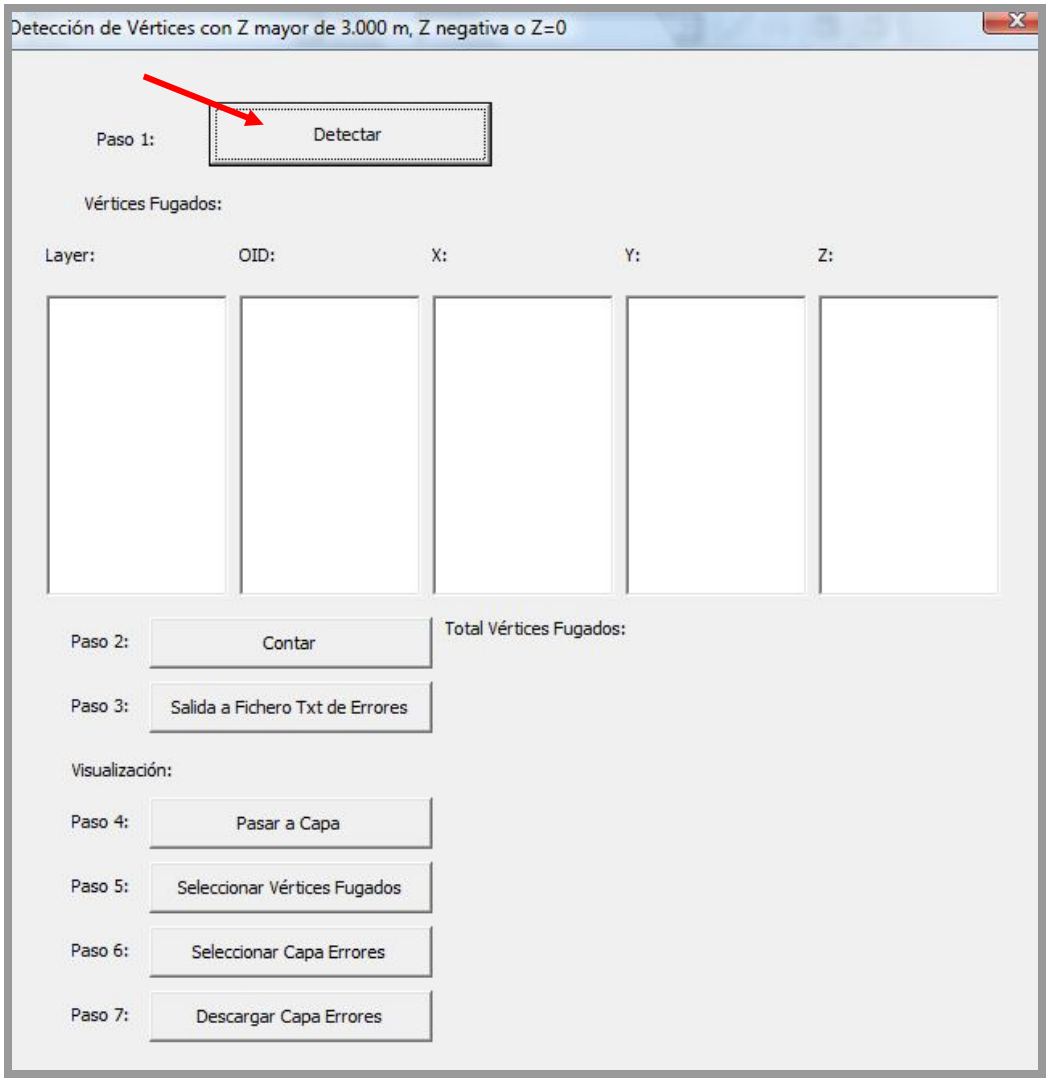

Figura nº 18. Formulario de análisis y visualización de "Vértices Fugados".

A medida que se termina de analizar una capa se muestra un "Cuadro de Mensaje" (Msgbox) que advierte de que se ha ejecutado correctamente.

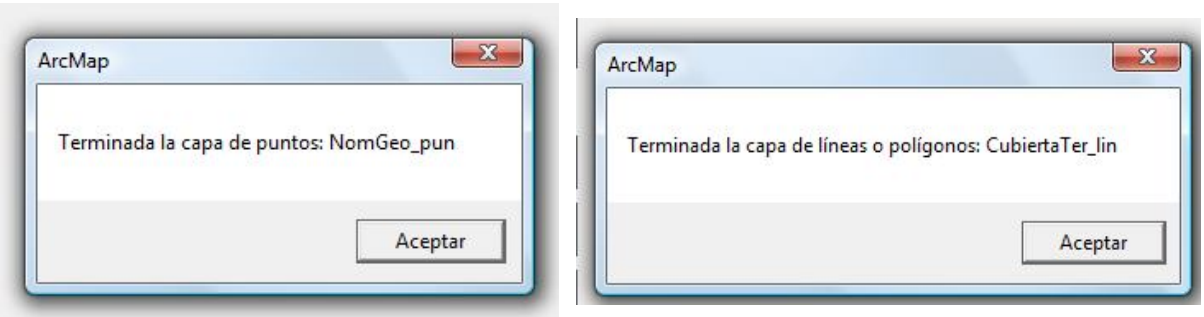

Figura nº 19. "Msgbox" de finalización de detección de vértices fugados en capas".

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

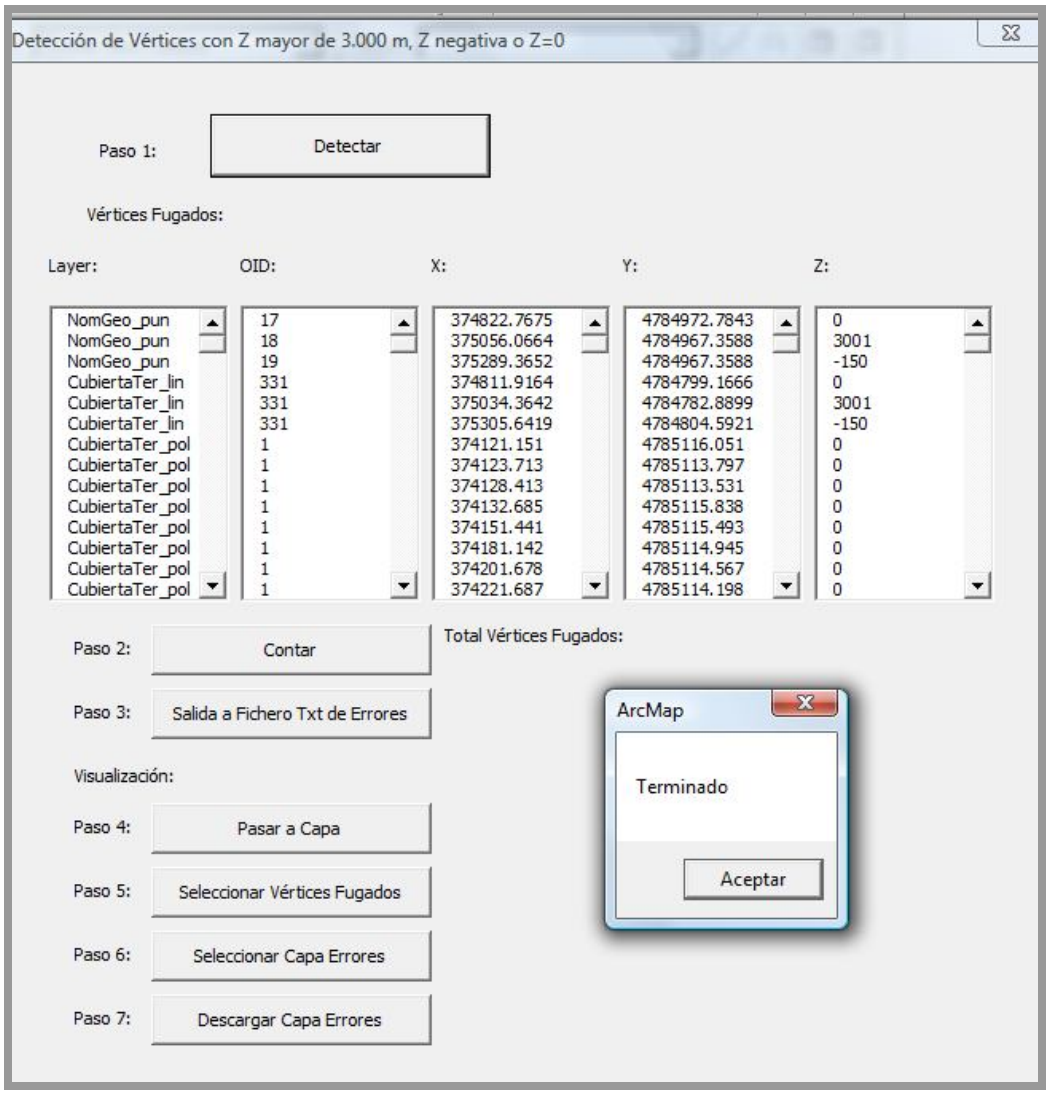

Figura nº 20. Paso 1, Resultados presentados en "listbox" y "Msgbox" de finalización de detección de vértices fugados en capas".

Paso 2.- "Contar".

Se inicia un contador para saber cuántos elementos se identificaron como "vértices fugados" en el total de las capas.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

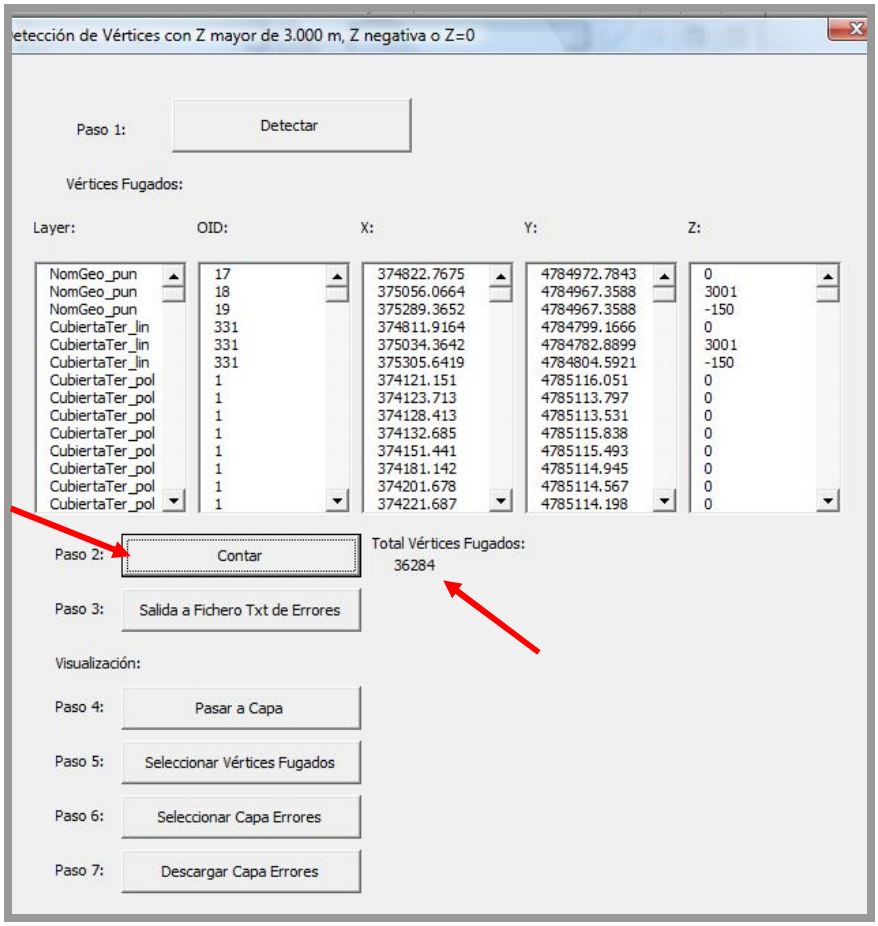

Figura nº 21. Paso 2, Inicialización del "Contador" y resultado.

Paso 3.- "Salida a Fichero Txt de Errores".

Ya se tienen identificados los vértices que el técnico deberá corregir. Una manera cómoda para poder editar dichos vértices, es disponerlos en un listado imprimible, de tal manera que el técnico vea cada elemento, en qué capa está y que OID tiene, para acceder rápidamente a las "Attribute tables". Para ello el comando del paso 3 "Salida a Fichero Txt de Errores", permite exportar los mismos datos presentados en los "listbox" a un fichero de texto ".txt".

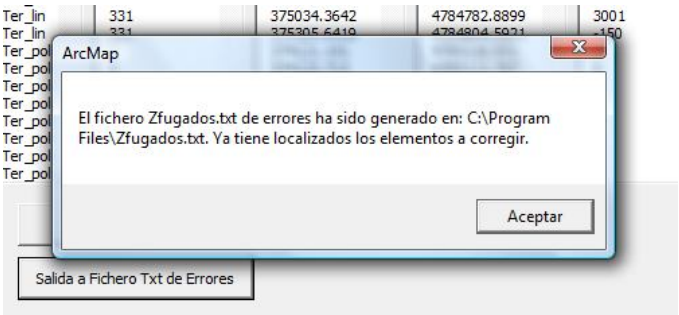

Figura nº 22. Paso 3,"Salida a Fichero Txt de Errores".

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Visualización:

Los próximos pasos 4, 5, 6 y 7, son opcionales. Permiten visualizar los "vértices fugados" en la pantalla de ArcMap.

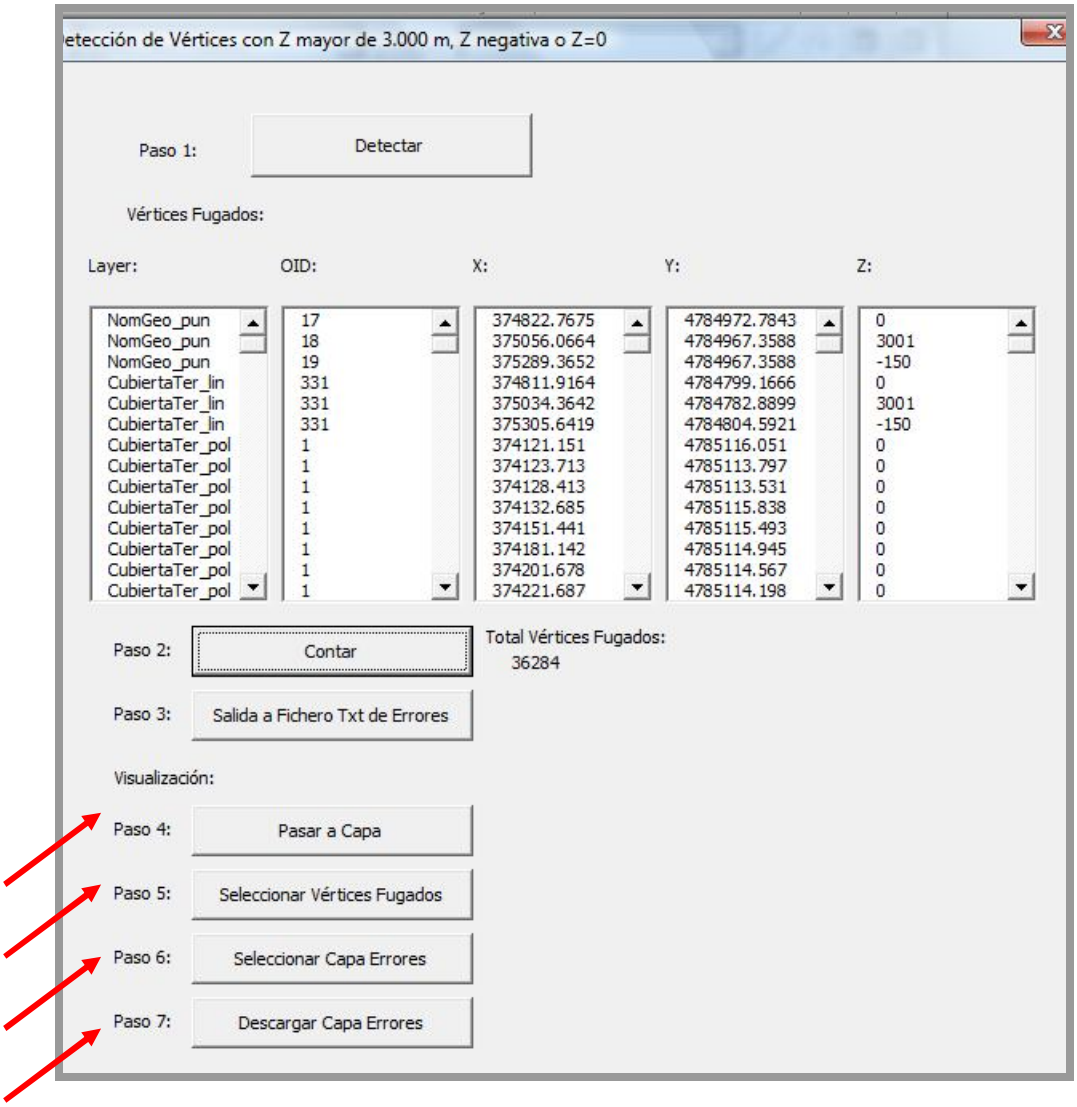

Figura nº 23. Visualización: pasos 4, 5, 6 y 7".

Paso 4.- "Pasar a Capa".

Los vértices que se identificaron como "vértices fugados" se copian en una nueva capa, que se carga la primera de la lista.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

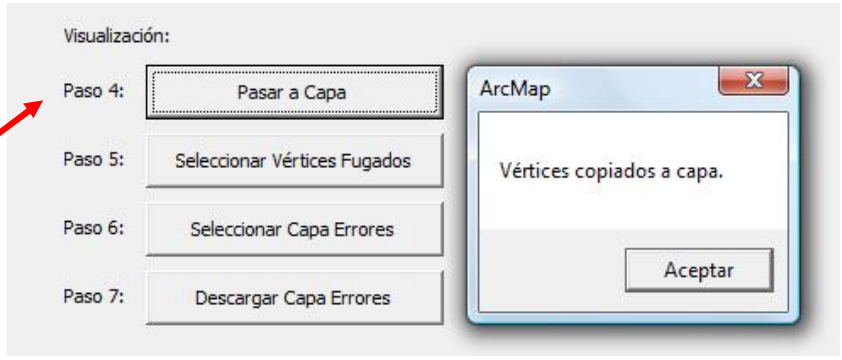

Figura nº 24. Paso 4,"Pasar vértices a capa".

Paso 5.- "Seleccionar Vértices Fugados".

Los vértices que se identificaron como "vértices fugados" se seleccionan y resaltan en la ventana de ArcMap.

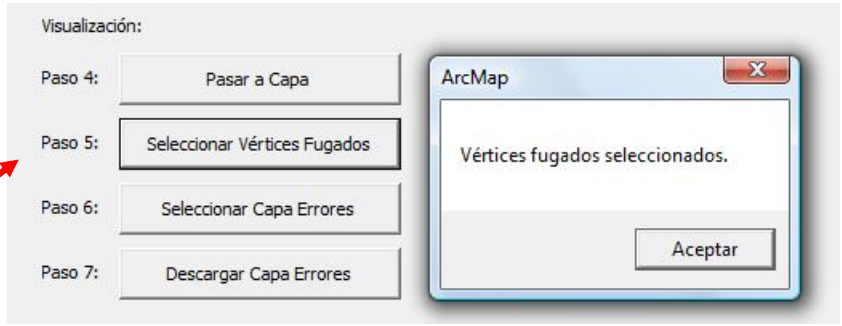

Figura nº 25. Paso 5,"Seleccionar vértices Fugados".

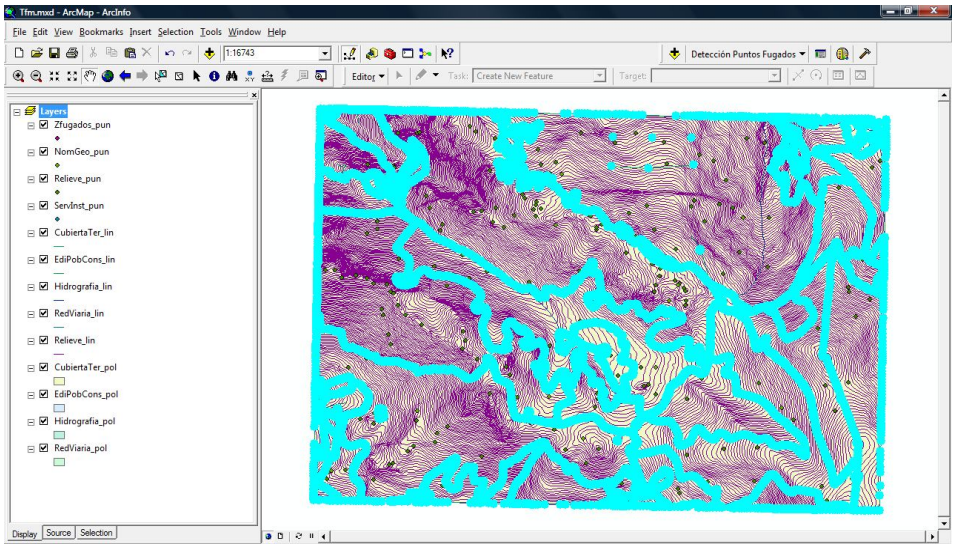

Figura nº 26.Vértices seleccionados en ventana de ArcMap.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Paso 6.- "Seleccionar Capa Errores".

La primera capa de la lista, que es la que contiene todos los "vértices fugados", se activa.

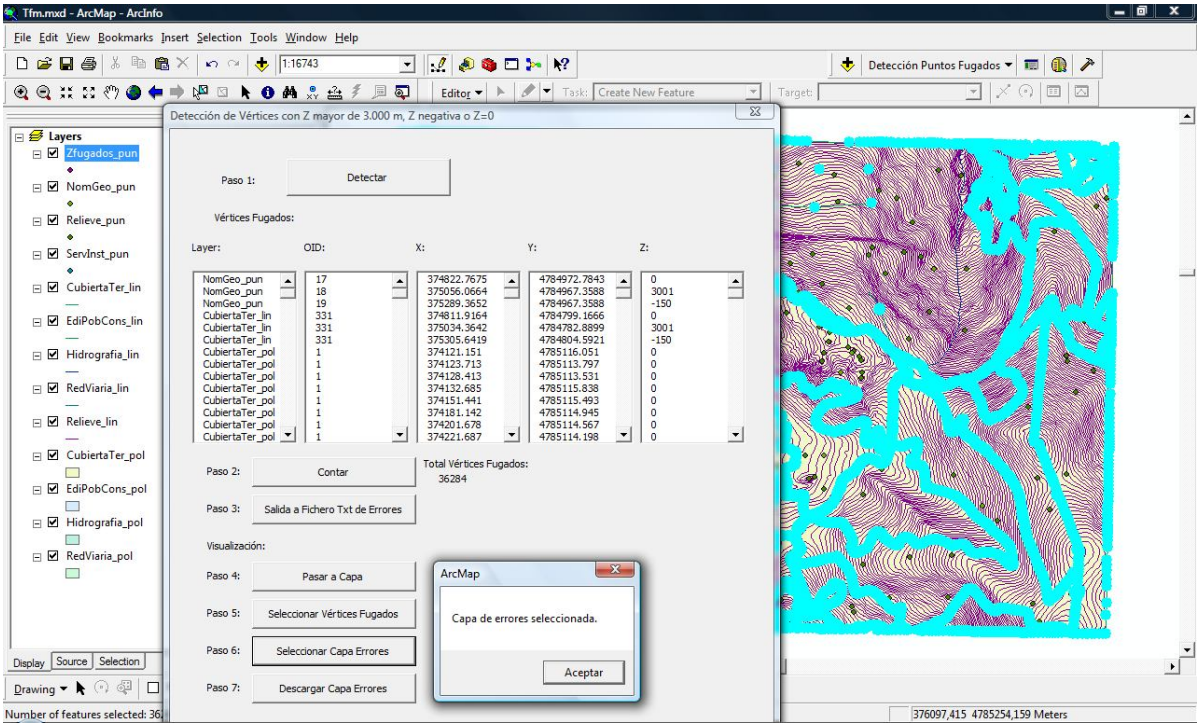

Figura nº 27, Paso 6,"Seleccionar Capa Errores".

Paso 7.- "Descargar Capa Errores".

La primera capa de la lista, que es la que contiene todos los "vértices fugados", se descarga y deja de visualizarse.

4.1.2.4 Acceso a ArcScene.

Permite acceder a ArcScene, el visor donde se podrán observar en 3D los puntos fugados.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

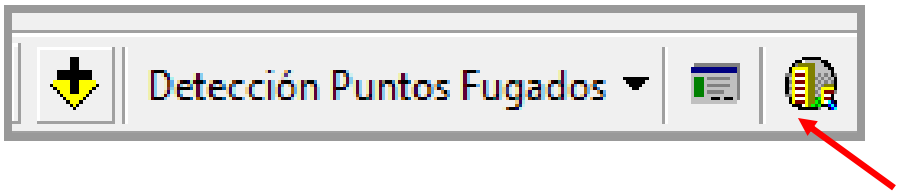

Figura nº 28. Comando de acceso a "ArcScene".

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

5 CONCLUSIONES

Se ha conseguido desarrollar una aplicación que cumple con los objetivos propuestos inicialmente, adaptándose a las características que se definieron al inicio del proyecto.

Una de las principales limitaciones para el desarrollo de la aplicación fue el poco conocimiento y falta agilidad en el área de programación, lo que supuso una gran inversión de tiempo para sobreponer estas restricciones. Por ello, es recomendable para desarrollar una aplicación similar, profundizar en el conocimiento y manejo de programación especialmente orientada a objetos, tener claro el manejo de las funcionalidades de ArcMap y la organización de los objetos dentro del mismo.

En cuanto a la operatividad de la aplicación, se trata de una herramienta sencilla y específica, de fácil manejo para el usuario, permitiéndole realizar las tareas de análisis y visualización en el entorno de ArcMap, de una forma ágil.

Dado que los comandos usualmente utilizados, han sido organizados en un solo entorno de trabajo, la barra de herramientas puede ser utilizada por un operador poco familiarizado con ArcMap, disminuyendo la inversión de tiempo.

En lo referente a futuras líneas de trabajo, es necesario continuar trabajando sobre la aplicación. Posiblemente este trabajo sirva de apoyo para crear otros aplicativos con nuevas y mejores funcionalidades. Esto será posible ya que la barra de herramientas es independiente del resto de objetos de ArcMap, lo cual permite que la misma pueda modificarse futuro. Esta aplicación es un buen punto de partida para próximas versiones.

La nueva barra de herramientas ha sido incorporada en un proyecto de ArcMap (Tfm.mxd), lo que permite que la misma sea fácilmente transportable a otros ordenadores.

6 BIBLIOGRAFÍA

- Jesús Palomar Vázquez. *"Programación en Sistemas de Información Geográfica: ArcObjects y VBA en ArcGis DesKtop"*. Valencia: Editorial UPV, 2008. ISBN 9778-84-8363-260-4.

- Burke, R. *"Getting Started with ArcObjects in ArcGis. Training Course. Esri"*. 1ª edición. Redlans, California. ESRI, 2003. ISBN 1-58948-018X.

- Ministerio de Fomento. Consejo Superior Geográfico. Comisión de Normas Cartográficas. "*Base Topográfica Armonizada 1:5 000 (BTA").* Enero 2008. [Consulta: marzo 2012]. Disponible en: < http://www.csg-cnc.es/web/cnccontent/bta.html>.

- Ruth Torres. *"Desarrollo de una aplicación para el manejo de elementos gráficos en el entorno de ArcMAP 9.2"*. Barcelona, 2009. [Consulta: marzo 2012]. Disponible en: <http://www.recercat.net/bitstream/handle/2072/41825/Treball_de_recerca.pdf?sequence=1>.

- Guillermo Som. "*Curso Básico de Programación en Visual Basic"*. Málaga, 2006. [Consulta: marzo 2012]. Disponible en: <http://www.elguille.info/vb/cursos_vb/BASICO/indice.htm>.

- "*ESRI Support Center*". Redlans, California. 1999 [Consulta: marzo 2012]. Disponible en: <http://www.support.esri.com>.

- *"ESRI Developer Network"*. Redlans, California. [Consulta: marzo 2012]. Disponible en:< http://edndoc.esri.com/arcobjects/9.2/CPP_VB6_VBA_VCPP_Doc/COM/VBA/welcome.htm>.

- *"GIS Pathway"*. East Carolina, 2011. [Consulta: marzo 2012]. Disponible en: <http://gispathway.com/>.

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Anejos

1. Código de programación de la aplicación

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

This Document:

```
Private Sub DeteccPuntosFugados_Click()
Dim FR As New FrmDeteccptosfugados
FR.Show
End Sub
Private Function DeteccPuntosFugados_ToolTip() As String
DeteccPuntosFugados_ToolTip = "Detección de Puntos Fugados"
End Function
Private Function AddData_ToolTip() As String
AddData_ToolTip = "Añadir Datos (capas)"
End Function
Private Function TINcomparar_ToolTip() As String
TINcomparar_ToolTip = "Comparación con TIN"
End Function
Private Sub Z0_Click()
Dim FR As New FrmZ0
FR.Show
End Sub
Private Sub Zmenor0_Click()
Dim FR As New FrmZnegativa
FR.Show
End Sub
Private Sub Zmayor3000_Click()
Dim FR As New FrmZmayor3000
FR.Show
End Sub
Private Sub Zfugados_Click()
Dim FR As New FrmZfugados
FR.Show
End Sub
Private Sub TINcomparar_Click()
'hacer por código es abrir el TIN y buscar la cota que le
corresponde a los vértices de los elementos
Dim FR As New FrmTin
FR.Show
End Sub
```

Formulario FrmDeteccptosfugados:

```
Private Sub CmdZ0_Click()
Dim FR As New FrmZ0
FR.Show
End Sub
Private Sub CmdZmayor3000_Click()
Dim FR As New FrmZmayor3000
FR.Show
End Sub
Private Sub CmdZnegativa_Click()
Dim FR As New FrmZnegativa
FR.Show
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
End Sub
Private Sub CmdZfugados_Click()
Dim FR As New FrmZfugados
FR.Show
End Sub
```

Formulario FrmZ0:

```
'************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS
'************************************
Public Sub CmdImprVertFCPtos_click()
ImprimirVerticesFeatureClassPuntos
End Sub
Public Sub ImprimirVerticesFeatureClassPuntos(NUMLAYER As
Integer, LAYERFINAL As Integer)
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features

    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
    While Not pFeature Is Nothing
        Debug.Print vbTab + "Feature OID " +
CStr(pFeature.OID)
        Dim pPoint As IPoint
        Set pPoint = pFeature.Shape

        If pPoint.Z = 0 Then


            Me.LsbLayer.AddItem pFeatureLayer.Name
            Me.LsbOID.AddItem pFeature.OID
            Me.LsbX.AddItem pPoint.X
            Me.LsbY.AddItem pPoint.Y
            Me.LsbZ.AddItem pPoint.Z
```


```
            Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
            End If

            Set pFeature = pFeatureCursor.NextFeature
             Wend

          Set pFeature = pFeatureCursor.NextFeature


End Sub
'*********************************************
' IMPRIMIR VÉRTICES CAPAS DE LINEAS Y POLÍGONOS
'*********************************************
Public Sub CmdImprVertFCLinPol_click()
ImprimirVerticesFeatureClassLinPol
End Sub
Public Sub ImprimirVerticesFeatureClassLinPol(NUMLAYER As
Integer, LAYERFINAL As Integer)
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature

    While Not pFeature Is Nothing
        Debug.Print "Feature OID " + CStr(pFeature.OID)
        ' Accede as la geometría con la interfaz
IPointCollection
        Dim pPointCollection As IPointCollection
        Set pPointCollection = pFeature.Shape
            Dim i As Integer
            For i = 0 To pPointCollection.PointCount - 1
            Dim pPoint As IPoint
            Set pPoint = pPointCollection.Point(i)

            If pPoint.Z = 0 Then
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
            Me.LsbLayer.AddItem pFeatureLayer.Name
            Me.LsbOID.AddItem pFeature.OID
            Me.LsbX.AddItem pPoint.X
            Me.LsbY.AddItem pPoint.Y
            Me.LsbZ.AddItem pPoint.Z

           Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
            End If

        Next i
        Set pFeature = pFeatureCursor.NextFeature
   Wend

End Sub


'*********************************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS, LINEAS Y POLÍGONOS
'*********************************************************

Private Sub CmdImprVertFCPtosLinPol_Click()
ImprimirVertices
End Sub


Public Sub ImprimirVertices()
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Dim c As Integer
    Dim pcapaclass As IFeatureClass

    Dim f As Integer


    'bucle para recorrer todas las capas
     For c = 0 To pMap.LayerCount - 1

    ' bucle para elegir el código, según sea la capa de ptos o
de líneas o polígonos

     Set pFeatureLayer = pMap.layer(c)
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
     Set pcapaclass = pFeatureLayer.FeatureClass

             If pcapaclass.ShapeType = esriGeometryPoint Then
            ImprimirVerticesFeatureClassPuntos c, f
            MsgBox "Terminada la capa de puntos: " &
pFeatureLayer.Name
             Else
            ImprimirVerticesFeatureClassLinPol c, f
            MsgBox "Terminada la capa de líneas o polígonos: "
& pFeatureLayer.Name

        End If

       Next c
       MsgBox "Terminado"

End Sub


'************************************
' IMPRIMIR VÉRTICES CON CURSOR
'************************************
Public Sub CmdImprimirVerticesConCursor_click()
ImprimirVerticesConCursor
End Sub
Public Sub ImprimirVerticesConCursor()
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(0)

    ' Crea la nueva feature class donde se van a guardar los
vértices.
    Dim pFeatureClassDestino As IFeatureClass
    Set pFeatureClassDestino = CrearFeatureClass("Z0_pun",
esriGeometryPoint, pFeatureLayer.FeatureClass.FeatureDataset)
    ' Crea una nueva layer, para que nos muestre en ArcMap la
featureclass creada
        Dim pFeatureLayerNueva As IFeatureLayer
        Set pFeatureLayerNueva = New FeatureLayer
    ' Se hace que la layer apunte a la featureclass
        Set pFeatureLayerNueva.FeatureClass =
pFeatureClassDestino
        ' Se le da nombre
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
        pFeatureLayerNueva.Name =
pFeatureClassDestino.AliasName
    ' Y se añade al mapa
        pMap.AddLayer pFeatureLayerNueva
    ' Se obtiene el cursor de inserción de la feature class
nueva.
    Dim pCursorInsercion As IFeatureCursor
    Set pCursorInsercion = pFeatureClassDestino.Insert(True)

    ' Se crea un feature buffer para ir guardando los
vertices. El feature buffer es una feature temporal,
    ' que no existe físicamente en la feature class hasta que
no se guarda con el cursor de inserción.
    Dim pFeatureBuffer As IFeatureBuffer
    Set pFeatureBuffer =
pFeatureClassDestino.CreateFeatureBuffer

    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)

    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
   ' While Not pFeature Is Nothing

        ' Accede as la geometría con la interfaz
IPointCollection
        'Dim pPointCollection As IPointCollection
        'Set pPointCollection = pFeature.Shape

        ' Recorre los vértices
        Dim i As Long
        For i = 0 To LsbOID.ListCount - 1

            Dim pPoint As IPoint

            'Set pPoint = pPointCollection.Point(0)
            Set pPoint = pFeature.Shape

            ' Al crear la nueva feature class, se a definido
como que tiene Zs en la línea:
            ' pGeometryDefEdit.HasZ = True
            ' Así que en cualquier geometría que vayamos a
añadir tenemos que asegurarnos de que tenga Z

              pPoint.X = puntoporcoma(Me.LsbX.List(i))
              pPoint.Y = puntoporcoma(Me.LsbY.List(i))
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
              pPoint.Z = puntoporcoma(Me.LsbZ.List(i))
            EstablecerZ pPoint

            ' Asigna el vértice a la geometría de la nueva
feature
            Set pFeatureBuffer.Shape = pPoint

            ' Guarda la feature en la tabla
            pCursorInsercion.InsertFeature pFeatureBuffer

        Next i

        Set pFeature = pFeatureCursor.NextFeature

    ' Guarda los cambios pendientes
    pCursorInsercion.Flush

End Sub

Private Function EstablecerZ(pPoint As IPoint)
    Dim pZAware As IZAware
    Set pZAware = pPoint
    ' Establece que la geometria pueda tener Zs
    pZAware.ZAware = True
    ' Comprueva que la Z sea válida
    If pZAware.ZSimple = False Then
        ' Si no lo es la deja a cota cero
        pPoint.Z = 0
    End If
End Function

Public Function CrearFeatureClass(Nombre As String, geomType
As esriGeometryType, pFeatureDataset As IFeatureDataset) As
IFeatureClass
On Error GoTo EH
    ' El nombre del campo shape se define aquí ya que se va a
usar en más de un sitio.
    Dim nombreCampoShape As String
    nombreCampoShape = "Shape"
    Dim pCLSID As UID
    Set pCLSID = New UID
    pCLSID.Value = "esriGeoDatabase.Feature"
    Dim pFields As IFields
    Set pFields = PrepararCampos(geomType, nombreCampoShape,
pFeatureDataset)
    Set CrearFeatureClass =
pFeatureDataset.CreateFeatureClass(Nombre, pFields, pCLSID,
Nothing, esriFTSimple, nombreCampoShape, "")
    Exit Function
EH:
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
    ' En caso de error muestra un mensaje con la descripción.
    MsgBox Err.Description, vbInformation, "CrearFeatureClass"
End Function

' Aquí se definen los campos que tendrá la capa, en este caso
solo van a ser el campo 'Shape' y el 'OBJECTID'
' que son los campos mínimos que puede tener una feature
class. Dentro del campo Shape se guarda la geometría,
' en el campo OBJECTID un identificador único para esta.
Private Function PrepararCampos(pEsriGeometryType As
esriGeometryType, nombreCampoShape As String, pFeatureDataset
As IFeatureDataset) As IFields
    Dim pFields As IFieldsEdit
    Set pFields = New esriGeoDatabase.Fields
    pFields.AddField CrearCampoGeometria(nombreCampoShape,
pFeatureDataset, pEsriGeometryType)
    pFields.AddField CrearCampo("OBJECTID", esriFieldTypeOID)
    ' Si se quiere añadir un campo personalizado se hace a
continuacion.
    ' P. ej. un campo de texto con el nombre de la geometría.
    ' pFields.AddField CrearCampo("NOMBRE",
esriFieldTypeString)

    Set PrepararCampos = pFields
End Function

Private Function CrearCampo(Nombre As String, tipo As
esriFieldType, Optional pGeometryType As esriGeometryType) As
IFieldEdit
    Set CrearCampo = New esriGeoDatabase.Field
    CrearCampo.Name = Nombre
    CrearCampo.AliasName = Nombre
    CrearCampo.Type = tipo
End Function

Private Function CrearCampoGeometria(Nombre As String,
pFeatureDataset As IFeatureDataset, Optional pGeometryType As
esriGeometryType) As IFieldEdit
    Set CrearCampoGeometria = New esriGeoDatabase.Field
    CrearCampoGeometria.Name = Nombre
    CrearCampoGeometria.AliasName = Nombre
    CrearCampoGeometria.Type =
esriFieldType.esriFieldTypeGeometry
    Dim pGeoDataset As IGeoDataset
    Set pGeoDataset = pFeatureDataset
    Dim pGeometryDefEdit As IGeometryDefEdit
    Set pGeometryDefEdit = New GeometryDef
    Set pGeometryDefEdit.SpatialReference =
pGeoDataset.SpatialReference
    pGeometryDefEdit.GeometryType = pGeometryType
```


```
    pGeometryDefEdit.GridCount = 1
    pGeometryDefEdit.GridSize(0) = 10
    pGeometryDefEdit.AvgNumPoints = 2
    pGeometryDefEdit.HasM = False
    pGeometryDefEdit.HasZ = True
    Set CrearCampoGeometria.GeometryDef = pGeometryDefEdit
End Function


'************************************
' CAMBIAR PUNTO POR COMA
'************************************
Function puntoporcoma(d As String)
Dim g As String
For i = 1 To Len(d)
    If Mid(d, i, 1) = "." Then
        g = g + ","
    Else
        g = g + Mid(d, i, 1)
    End If
Next i
puntoporcoma = g
End Function

'*****************
' CONTADOR
'*****************
Private Sub CmdContador_Click()
Dim contador As Long
    contador = -1
        For i = 0 To LsbX.ListCount
        contador = contador + 1
        LblContador = contador
        Next i
End Sub


'*********************************
' SELECCIONAR PUNTOS FUGADOS
'*********************************
Private Sub CmdSeleccpuntosfugados_Click()


Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument

'Acceso al mapa actual
Dim pMapa As IMap
Set pMapa = pMxDoc.FocusMap
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
'Acceso a la 1ª capa
Dim pCapa1 As IFeatureLayer
Set pCapa1 = pMapa.layer(0)

'Acceso a la 2ª capa
Dim pCapa2 As IFeatureLayer
Set pCapa2 = pMapa.layer(i)
For i = 0 To pMapa.LayerCount - 1
Next i

'Creación del objeto de geoprocesamiento
Dim GP As Object
Set GP = CreateObject("esriGeoprocessing.GpDispatch.1")
'Acceso y ejecución de la herramienta
'argumentos: capa a seleccionar, relación espacial, capa en la
que se basará el filtro espacial
'En este caso la sintáxis sería:"selecciona los elementos de
la capa 2 que intersecten con...
'los elementos de la capa 1
GP.SelectLayerByLocation_management pCapa2, "INTERSECT",
pCapa1
 MsgBox "Vértices fugados seleccionados."
End Sub



'*********************************
' SELECCIONAR CAPA DE ERRORES
'*********************************
Private Sub CmdSelectlayer_Click()


' selects all of the layers in the first data frame in the
toc's display view

Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
Dim pTOC As IContentsView
Set pTOC = pMxDoc.ContentsView(0) ' Display View

Dim pMaps As IMaps
Set pMaps = pMxDoc.Maps
Dim pMap As IMap

Dim i As Integer
Set pMap = pMaps.Item(0) ' first data frame
Dim pEnumLayer As IEnumLayer
Set pEnumLayer = pMap.Layers
Dim pLayer As ILayer
Set pLayer = pEnumLayer.Next
```


```
pTOC.RemoveFromSelectedItems pTOC.SelectedItem
'Do While Not pLayer Is Nothing
pTOC.AddToSelectedItems pLayer
pTOC.Refresh pLayer
'Set pLayer = pEnumLayer.Next
'Loop
 MsgBox "Capa de errores seleccionada."
End Sub



'*********************************
' DESCARGAR CAPA DE ERRORES
'*********************************
Public Sub CmdDescargarcapaerror_Click()

  ' Get the map
  Dim pDoc As IMxDocument
  Dim pMap As IMap
  Set pDoc = ThisDocument
  Set pMap = pDoc.FocusMap

  ' Get the selected layer or table
  Dim pSelItem As IUnknown
  Set pSelItem = pDoc.SelectedItem
  If pSelItem Is Nothing Then
    MsgBox "No Feature layer selected"
    Exit Sub
  ' Remove a Layer and refresh the map
  ElseIf TypeOf pSelItem Is IFeatureLayer Then
    pMap.DeleteLayer pDoc.SelectedLayer
    Dim pActiveView As IActiveView
    Set pActiveView = pMap
    pActiveView.Refresh
   'Remove a table
  ElseIf TypeOf pSelItem Is IStandaloneTable Then
    Dim pStTab As IStandaloneTable
    Dim pStTabColl As IStandaloneTableCollection
    Set pStTab = pSelItem
    Set pStTabColl = pMap
    pStTabColl.RemoveStandaloneTable pStTab
  Else
    MsgBox "Selected item is not a table or layer"
    Exit Sub
  End If

  ' refresh the TOC
  pDoc.UpdateContents
MsgBox "Ahora conviene comparar con un TIN"
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
End Sub


'************************************
' IMPRIMIR FICHERO TXT DE ERRORES
'************************************

Private Sub CmdImprimirficherotxterrores_Click()

Dim Linea1 As String
Dim Linea2 As String
Dim Linea3 As String
Dim Linea4 As String
Dim Linea5 As String
Dim Arch As String
Dim i As Long

   Arch = " C:\Program Files\Z0.txt"
   Open Arch For Output Access Write As #5

   For i = 0 To LsbLayer.ListCount - 1
    Linea1 = ""
    Linea2 = ""
    Linea3 = ""
    Linea4 = ""
    Linea5 = ""
    Linea1 = Linea1 & LsbLayer.List(i)
    Linea2 = Linea2 & LsbOID.List(i)
    Linea3 = Linea3 & LsbX.List(i)
    Linea4 = Linea4 & LsbY.List(i)
    Linea5 = Linea5 & LsbZ.List(i)
   Print #5, "Capa: " & Linea1, "OID: " & Linea2, "X: " &
Linea3, "Y: " & Linea4, "Z: " & Linea5
   Next i
   Close #5
   Close #4
   Close #3
   Close #2
   Close #1

   MsgBox "El fichero Z0.txt de errores ha sido generado en: "
& Arch & ". Ya tiene localizados los elementos a corregir."
End Sub
```

Formulario FrmZnegativa:

```
'************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS
'************************************
Public Sub CmdImprVertFCPtos_click()
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
ImprimirVerticesFeatureClassPuntos
End Sub
Public Sub ImprimirVerticesFeatureClassPuntos(NUMLAYER As
Integer, LAYERFINAL As Integer)
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features

    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
    While Not pFeature Is Nothing
        Debug.Print vbTab + "Feature OID " +
CStr(pFeature.OID)
        Dim pPoint As IPoint
        Set pPoint = pFeature.Shape

        If pPoint.Z < 0 Then


            Me.LsbLayer.AddItem pFeatureLayer.Name
            Me.LsbOID.AddItem pFeature.OID
            Me.LsbX.AddItem pPoint.X
            Me.LsbY.AddItem pPoint.Y
            Me.LsbZ.AddItem pPoint.Z


           Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
           End If

           Set pFeature = pFeatureCursor.NextFeature
            Wend

         Set pFeature = pFeatureCursor.NextFeature


End Sub
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
'*********************************************
' IMPRIMIR VÉRTICES CAPAS DE LINEAS Y POLÍGONOS
'*********************************************
Public Sub CmdImprVertFCLinPol_click()
ImprimirVerticesFeatureClassLinPol
End Sub
Public Sub ImprimirVerticesFeatureClassLinPol(NUMLAYER As
Integer, LAYERFINAL As Integer)
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature

    While Not pFeature Is Nothing
        Debug.Print "Feature OID " + CStr(pFeature.OID)
        ' Accede as la geometría con la interfaz
IPointCollection
        Dim pPointCollection As IPointCollection
        Set pPointCollection = pFeature.Shape
            Dim i As Integer
            For i = 0 To pPointCollection.PointCount - 1
            Dim pPoint As IPoint
            Set pPoint = pPointCollection.Point(i)

            If pPoint.Z < 0 Then

             Me.LsbLayer.AddItem pFeatureLayer.Name
             Me.LsbOID.AddItem pFeature.OID
             Me.LsbX.AddItem pPoint.X
             Me.LsbY.AddItem pPoint.Y
             Me.LsbZ.AddItem pPoint.Z

            Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
            End If
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
        Next i
        Set pFeature = pFeatureCursor.NextFeature
   Wend

End Sub


'*********************************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS, LINEAS Y POLÍGONOS
'*********************************************************

Private Sub CmdImprVertFCPtosLinPol_Click()
ImprimirVertices
End Sub


Public Sub ImprimirVertices()
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Dim c As Integer
    Dim pcapaclass As IFeatureClass

    Dim f As Integer


    'bucle para recorrer todas las capas
     For c = 0 To pMap.LayerCount - 1

    ' bucle para elegir el código, según sea la capa de ptos o
de líneas o polígonos

     Set pFeatureLayer = pMap.layer(c)
     Set pcapaclass = pFeatureLayer.FeatureClass

             If pcapaclass.ShapeType = esriGeometryPoint Then
            ImprimirVerticesFeatureClassPuntos c, f
            MsgBox "Terminada la capa de puntos: " &
pFeatureLayer.Name
             Else
            ImprimirVerticesFeatureClassLinPol c, f
            MsgBox "Terminada la capa de líneas o polígonos: "
& pFeatureLayer.Name

        End If
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
        Next c
        MsgBox "Terminado"

End Sub


'*************************************
' IMPRIMIR VÉRTICES CON CURSOR
'*************************************
Public Sub CmdImprimirVerticesConCursor_click()
ImprimirVerticesConCursor
End Sub
Public Sub ImprimirVerticesConCursor()
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(0)

    ' Crea la nueva feature class donde se van a guardar los
vértices.
    Dim pFeatureClassDestino As IFeatureClass
    Set pFeatureClassDestino =
CrearFeatureClass("Znegativa_pun", esriGeometryPoint,
pFeatureLayer.FeatureClass.FeatureDataset)
    ' Crea una nueva layer, para que nos muestre en ArcMap la
featureclass creada
        Dim pFeatureLayerNueva As IFeatureLayer
        Set pFeatureLayerNueva = New FeatureLayer
    ' Se hace que la layer apunte a la featureclass
        Set pFeatureLayerNueva.FeatureClass =
pFeatureClassDestino
        ' Se le da nombre
        pFeatureLayerNueva.Name =
pFeatureClassDestino.AliasName
    ' Y se añade al mapa
        pMap.AddLayer pFeatureLayerNueva
    ' Se obtiene el cursor de inserción de la feature class
nueva.
    Dim pCursorInsercion As IFeatureCursor
    Set pCursorInsercion = pFeatureClassDestino.Insert(True)

    ' Se crea un feature buffer para ir guardando los
vertices. El feature buffer es una feature temporal,
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
    ' que no existe físicamente en la feature class hasta que
no se guarda con el cursor de inserción.
    Dim pFeatureBuffer As IFeatureBuffer
    Set pFeatureBuffer =
pFeatureClassDestino.CreateFeatureBuffer

    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)

    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
   ' While Not pFeature Is Nothing

        ' Accede as la geometría con la interfaz
IPointCollection
        'Dim pPointCollection As IPointCollection
        'Set pPointCollection = pFeature.Shape

        ' Recorre los vértices
        Dim i As Long
        For i = 0 To LsbOID.ListCount - 1

            Dim pPoint As IPoint

            'Set pPoint = pPointCollection.Point(0)
            Set pPoint = pFeature.Shape

            ' Al crear la nueva feature class, se a definido
como que tiene Zs en la línea:
            ' pGeometryDefEdit.HasZ = True
            ' Así que en cualquier geometría que vayamos a
añadir tenemos que asegurarnos de que tenga Z

              pPoint.X = puntoporcoma(Me.LsbX.List(i))
              pPoint.Y = puntoporcoma(Me.LsbY.List(i))
              pPoint.Z = puntoporcoma(Me.LsbZ.List(i))
            EstablecerZ pPoint

            ' Asigna el vértice a la geometría de la nueva
feature
            Set pFeatureBuffer.Shape = pPoint

            ' Guarda la feature en la tabla
            pCursorInsercion.InsertFeature pFeatureBuffer

        Next i
```


```
        Set pFeature = pFeatureCursor.NextFeature

    ' Guarda los cambios pendientes
    pCursorInsercion.Flush

End Sub

Private Function EstablecerZ(pPoint As IPoint)
    Dim pZAware As IZAware
    Set pZAware = pPoint
    ' Establece que la geometria pueda tener Zs
    pZAware.ZAware = True
    ' Comprueva que la Z sea válida
    If pZAware.ZSimple = False Then
        ' Si no lo es la deja a cota cero
        pPoint.Z = 0
    End If
End Function

Public Function CrearFeatureClass(Nombre As String, geomType
As esriGeometryType, pFeatureDataset As IFeatureDataset) As
IFeatureClass
On Error GoTo EH
    ' El nombre del campo shape se define aquí ya que se va a
usar en más de un sitio.
    Dim nombreCampoShape As String
    nombreCampoShape = "Shape"
    Dim pCLSID As UID
    Set pCLSID = New UID
    pCLSID.Value = "esriGeoDatabase.Feature"
    Dim pFields As IFields
    Set pFields = PrepararCampos(geomType, nombreCampoShape,
pFeatureDataset)
    Set CrearFeatureClass =
pFeatureDataset.CreateFeatureClass(Nombre, pFields, pCLSID,
Nothing, esriFTSimple, nombreCampoShape, "")
    Exit Function
EH:
    ' En caso de error muestra un mensaje con la descripción.
    MsgBox Err.Description, vbInformation, "CrearFeatureClass"
End Function

' Aquí se definen los campos que tendrá la capa, en este caso
solo van a ser el campo 'Shape' y el 'OBJECTID'
' que son los campos mínimos que puede tener una feature
class. Dentro del campo Shape se guarda la geometría,
' en el campo OBJECTID un identificador único para esta.
```


```
Private Function PrepararCampos(pEsriGeometryType As
esriGeometryType, nombreCampoShape As String, pFeatureDataset
As IFeatureDataset) As IFields
    Dim pFields As IFieldsEdit
    Set pFields = New esriGeoDatabase.Fields
    pFields.AddField CrearCampoGeometria(nombreCampoShape,
pFeatureDataset, pEsriGeometryType)
    pFields.AddField CrearCampo("OBJECTID", esriFieldTypeOID)
    ' Si se quiere añadir un campo personalizado se hace a
continuacion.
    ' P. ej. un campo de texto con el nombre de la geometría.
    ' pFields.AddField CrearCampo("NOMBRE",
esriFieldTypeString)

    Set PrepararCampos = pFields
End Function

Private Function CrearCampo(Nombre As String, tipo As
esriFieldType, Optional pGeometryType As esriGeometryType) As
IFieldEdit
    Set CrearCampo = New esriGeoDatabase.Field
    CrearCampo.Name = Nombre
    CrearCampo.AliasName = Nombre
    CrearCampo.Type = tipo
End Function

Private Function CrearCampoGeometria(Nombre As String,
pFeatureDataset As IFeatureDataset, Optional pGeometryType As
esriGeometryType) As IFieldEdit
    Set CrearCampoGeometria = New esriGeoDatabase.Field
    CrearCampoGeometria.Name = Nombre
    CrearCampoGeometria.AliasName = Nombre
    CrearCampoGeometria.Type =
esriFieldType.esriFieldTypeGeometry
    Dim pGeoDataset As IGeoDataset
    Set pGeoDataset = pFeatureDataset
    Dim pGeometryDefEdit As IGeometryDefEdit
    Set pGeometryDefEdit = New GeometryDef
    Set pGeometryDefEdit.SpatialReference =
pGeoDataset.SpatialReference
    pGeometryDefEdit.GeometryType = pGeometryType
    pGeometryDefEdit.GridCount = 1
    pGeometryDefEdit.GridSize(0) = 10
    pGeometryDefEdit.AvgNumPoints = 2
    pGeometryDefEdit.HasM = False
    pGeometryDefEdit.HasZ = True
    Set CrearCampoGeometria.GeometryDef = pGeometryDefEdit
End Function
```


```
'************************************
' CAMBIAR PUNTO POR COMA
'************************************
Function puntoporcoma(d As String)
Dim g As String
For i = 1 To Len(d)
    If Mid(d, i, 1) = "." Then
        g = g + ","
    Else
        g = g + Mid(d, i, 1)
    End If
Next i
puntoporcoma = g
End Function

'*****************
' CONTADOR
'*****************
Private Sub CmdContador_Click()
Dim contador As Long
    contador = -1
        For i = 0 To LsbX.ListCount
        contador = contador + 1
        LblContador = contador
        Next i
End Sub


'*********************************
' SELECCIONAR PUNTOS FUGADOS
'*********************************
Private Sub CmdSeleccpuntosfugados_Click()


Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument

'Acceso al mapa actual
Dim pMapa As IMap
Set pMapa = pMxDoc.FocusMap

'Acceso a la 1ª capa
Dim pCapa1 As IFeatureLayer
Set pCapa1 = pMapa.layer(0)

'Acceso a la 2ª capa
Dim pCapa2 As IFeatureLayer
Set pCapa2 = pMapa.layer(i)
For i = 0 To pMapa.LayerCount - 1
Next i
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
'Creación del objeto de geoprocesamiento
Dim GP As Object
Set GP = CreateObject("esriGeoprocessing.GpDispatch.1")
'Acceso y ejecución de la herramienta
'argumentos: capa a seleccionar, relación espacial, capa en la
que se basará el filtro espacial
'En este caso la sintáxis sería:"selecciona los elementos de
la capa 2 que intersecten con...
'los elementos de la capa 1
GP.SelectLayerByLocation_management pCapa2, "INTERSECT",
pCapa1
 MsgBox "Vértices fugados seleccionados."
End Sub



'*********************************
' SELECCIONAR CAPA DE ERRORES
'*********************************
Private Sub CmdSelectlayer_Click()


' selects all of the layers in the first data frame in the
toc's display view

Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
Dim pTOC As IContentsView
Set pTOC = pMxDoc.ContentsView(0) ' Display View

Dim pMaps As IMaps
Set pMaps = pMxDoc.Maps
Dim pMap As IMap

Dim i As Integer
Set pMap = pMaps.Item(0) ' first data frame
Dim pEnumLayer As IEnumLayer
Set pEnumLayer = pMap.Layers
Dim pLayer As ILayer
Set pLayer = pEnumLayer.Next

pTOC.RemoveFromSelectedItems pTOC.SelectedItem
'Do While Not pLayer Is Nothing
pTOC.AddToSelectedItems pLayer
pTOC.Refresh pLayer
'Set pLayer = pEnumLayer.Next
'Loop
 MsgBox "Capa de errores seleccionada."
End Sub
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
'*********************************
' DESCARGAR CAPA DE ERRORES
'*********************************
Public Sub CmdDescargarcapaerror_Click()

  ' Get the map
  Dim pDoc As IMxDocument
  Dim pMap As IMap
  Set pDoc = ThisDocument
  Set pMap = pDoc.FocusMap

  ' Get the selected layer or table
  Dim pSelItem As IUnknown
  Set pSelItem = pDoc.SelectedItem
  If pSelItem Is Nothing Then
    MsgBox "No Feature layer selected"
    Exit Sub
  ' Remove a Layer and refresh the map
  ElseIf TypeOf pSelItem Is IFeatureLayer Then
    pMap.DeleteLayer pDoc.SelectedLayer
    Dim pActiveView As IActiveView
    Set pActiveView = pMap
    pActiveView.Refresh
   'Remove a table
  ElseIf TypeOf pSelItem Is IStandaloneTable Then
    Dim pStTab As IStandaloneTable
    Dim pStTabColl As IStandaloneTableCollection
    Set pStTab = pSelItem
    Set pStTabColl = pMap
    pStTabColl.RemoveStandaloneTable pStTab
  Else
    MsgBox "Selected item is not a table or layer"
    Exit Sub
  End If

  ' refresh the TOC
  pDoc.UpdateContents
MsgBox "Ahora hay que ejecutar Create TIN, Edit TIN y TIN to
Ráster/TIN to Features"
End Sub


'*********************************
' IMPRIMIR FICHERO TXT DE ERRORES
'*********************************
Private Sub CmdImprimirficherotxterrores_Click()
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
Dim Linea1 As String
Dim Linea2 As String
Dim Linea3 As String
Dim Linea4 As String
Dim Linea5 As String
Dim Arch As String
Dim i As Long

   Arch = "C:\Program Files\Zmenor0.txt"
   Open Arch For Output Access Write As #5

   For i = 0 To LsbLayer.ListCount - 1
    Linea1 = ""
    Linea2 = ""
    Linea3 = ""
    Linea4 = ""
    Linea5 = ""
    Linea1 = Linea1 & LsbLayer.List(i)
    Linea2 = Linea2 & LsbOID.List(i)
    Linea3 = Linea3 & LsbX.List(i)
    Linea4 = Linea4 & LsbY.List(i)
    Linea5 = Linea5 & LsbZ.List(i)
   Print #5, "Capa: " & Linea1, "OID: " & Linea2, "X: " &
Linea3, "Y: " & Linea4, "Z: " & Linea5
   Next i
   Close #5
   Close #4
   Close #3
   Close #2
   Close #1

   MsgBox "El fichero Zmenor0.txt de errores ha sido generado
en: " & Arch & ". Ya tiene localizados los elementos a
corregir."
End Sub
```

Formulario FrmZmayor3000:

```
'*************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS
'*************************************
Public Sub CmdImprVertFCPtos_click()
ImprimirVerticesFeatureClassPuntos
End Sub
Public Sub ImprimirVerticesFeatureClassPuntos(NUMLAYER As
Integer, LAYERFINAL As Integer)
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features

    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
    While Not pFeature Is Nothing
        Debug.Print vbTab + "Feature OID " +
CStr(pFeature.OID)
        Dim pPoint As IPoint
        Set pPoint = pFeature.Shape

        If pPoint.Z >= 3000 Then


            Me.LsbLayer.AddItem pFeatureLayer.Name
            Me.LsbOID.AddItem pFeature.OID
            Me.LsbX.AddItem pPoint.X
            Me.LsbY.AddItem pPoint.Y
            Me.LsbZ.AddItem pPoint.Z


           Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
           End If

           Set pFeature = pFeatureCursor.NextFeature
            Wend

         Set pFeature = pFeatureCursor.NextFeature


End Sub
'*********************************************
' IMPRIMIR VÉRTICES CAPAS DE LINEAS Y POLÍGONOS
'*********************************************
Public Sub CmdImprVertFCLinPol_click()
ImprimirVerticesFeatureClassLinPol
End Sub
Public Sub ImprimirVerticesFeatureClassLinPol(NUMLAYER As
Integer, LAYERFINAL As Integer)
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature

    While Not pFeature Is Nothing
        Debug.Print "Feature OID " + CStr(pFeature.OID)
        ' Accede as la geometría con la interfaz
IPointCollection
        Dim pPointCollection As IPointCollection
        Set pPointCollection = pFeature.Shape
            Dim i As Integer
            For i = 0 To pPointCollection.PointCount - 1
            Dim pPoint As IPoint
            Set pPoint = pPointCollection.Point(i)

            If pPoint.Z >= 3000 Then

             Me.LsbLayer.AddItem pFeatureLayer.Name
             Me.LsbOID.AddItem pFeature.OID
             Me.LsbX.AddItem pPoint.X
             Me.LsbY.AddItem pPoint.Y
             Me.LsbZ.AddItem pPoint.Z

            Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
            End If

        Next i
        Set pFeature = pFeatureCursor.NextFeature
   Wend

End Sub


'*****************************************************
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS, LINEAS Y POLÍGONOS
'*********************************************************

Private Sub CmdImprVertFCPtosLinPol_Click()
ImprimirVertices
End Sub


Public Sub ImprimirVertices()
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Dim c As Integer
    Dim pcapaclass As IFeatureClass

    Dim f As Integer


    'bucle para recorrer todas las capas
     For c = 0 To pMap.LayerCount - 1

    ' bucle para elegir el código, según sea la capa de ptos o
de líneas o polígonos

     Set pFeatureLayer = pMap.layer(c)
     Set pcapaclass = pFeatureLayer.FeatureClass

              If pcapaclass.ShapeType = esriGeometryPoint Then
            ImprimirVerticesFeatureClassPuntos c, f
            MsgBox "Terminada la capa de puntos: " &
pFeatureLayer.Name
             Else
            ImprimirVerticesFeatureClassLinPol c, f
            MsgBox "Terminada la capa de líneas o polígonos: "
& pFeatureLayer.Name

        End If

       Next c
       MsgBox "Terminado"

End Sub


'*************************************
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
' IMPRIMIR VÉRTICES CON CURSOR
'************************************
Public Sub CmdImprimirVerticesConCursor_click()
ImprimirVerticesConCursor
End Sub
Public Sub ImprimirVerticesConCursor()
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(0)

    ' Crea la nueva feature class donde se van a guardar los
vértices.
    Dim pFeatureClassDestino As IFeatureClass
    Set pFeatureClassDestino =
CrearFeatureClass("Zmayor3000_pun", esriGeometryPoint,
pFeatureLayer.FeatureClass.FeatureDataset)
    ' Crea una nueva layer, para que nos muestre en ArcMap la
featureclass creada
        Dim pFeatureLayerNueva As IFeatureLayer
        Set pFeatureLayerNueva = New FeatureLayer
    ' Se hace que la layer apunte a la featureclass
        Set pFeatureLayerNueva.FeatureClass =
pFeatureClassDestino
        ' Se le da nombre
        pFeatureLayerNueva.Name =
pFeatureClassDestino.AliasName
    ' Y se añade al mapa
        pMap.AddLayer pFeatureLayerNueva
    ' Se obtiene el cursor de inserción de la feature class
nueva.
    Dim pCursorInsercion As IFeatureCursor
    Set pCursorInsercion = pFeatureClassDestino.Insert(True)

    ' Se crea un feature buffer para ir guardando los
vertices. El feature buffer es una feature temporal,
    ' que no existe físicamente en la feature class hasta que
no se guarda con el cursor de inserción.
    Dim pFeatureBuffer As IFeatureBuffer
    Set pFeatureBuffer =
pFeatureClassDestino.CreateFeatureBuffer

    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
```


```
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)

    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
   ' While Not pFeature Is Nothing

        ' Accede as la geometría con la interfaz
IPointCollection
        'Dim pPointCollection As IPointCollection
        'Set pPointCollection = pFeature.Shape

        ' Recorre los vértices
        Dim i As Long
        For i = 0 To LsbOID.ListCount - 1

            Dim pPoint As IPoint

            'Set pPoint = pPointCollection.Point(0)
            Set pPoint = pFeature.Shape

            ' Al crear la nueva feature class, se a definido
como que tiene Zs en la línea:
            ' pGeometryDefEdit.HasZ = True
            ' Así que en cualquier geometría que vayamos a
añadir tenemos que asegurarnos de que tenga Z

              pPoint.X = puntoporcoma(Me.LsbX.List(i))
              pPoint.Y = puntoporcoma(Me.LsbY.List(i))
              pPoint.Z = puntoporcoma(Me.LsbZ.List(i))
            EstablecerZ pPoint

            ' Asigna el vértice a la geometría de la nueva
feature
            Set pFeatureBuffer.Shape = pPoint

            ' Guarda la feature en la tabla
            pCursorInsercion.InsertFeature pFeatureBuffer

        Next i

        Set pFeature = pFeatureCursor.NextFeature

    ' Guarda los cambios pendientes
    pCursorInsercion.Flush
      MsgBox "Vértices copiados a capa."
End Sub

Private Function EstablecerZ(pPoint As IPoint)
```


```
    Dim pZAware As IZAware
    Set pZAware = pPoint
    ' Establece que la geometria pueda tener Zs
    pZAware.ZAware = True
    ' Comprueva que la Z sea válida
    If pZAware.ZSimple = False Then
        ' Si no lo es la deja a cota cero
        pPoint.Z = 0
    End If
End Function

Public Function CrearFeatureClass(Nombre As String, geomType
As esriGeometryType, pFeatureDataset As IFeatureDataset) As
IFeatureClass
On Error GoTo EH
    ' El nombre del campo shape se define aquí ya que se va a
usar en más de un sitio.
    Dim nombreCampoShape As String
    nombreCampoShape = "Shape"
    Dim pCLSID As UID
    Set pCLSID = New UID
    pCLSID.Value = "esriGeoDatabase.Feature"
    Dim pFields As IFields
    Set pFields = PrepararCampos(geomType, nombreCampoShape,
pFeatureDataset)
    Set CrearFeatureClass =
pFeatureDataset.CreateFeatureClass(Nombre, pFields, pCLSID,
Nothing, esriFTSimple, nombreCampoShape, "")
    Exit Function
EH:
    ' En caso de error muestra un mensaje con la descripción.
    MsgBox Err.Description, vbInformation, "CrearFeatureClass"
End Function

' Aquí se definen los campos que tendrá la capa, en este caso
solo van a ser el campo 'Shape' y el 'OBJECTID'
' que son los campos mínimos que puede tener una feature
class. Dentro del campo Shape se guarda la geometría,
' en el campo OBJECTID un identificador único para esta.
Private Function PrepararCampos(pEsriGeometryType As
esriGeometryType, nombreCampoShape As String, pFeatureDataset
As IFeatureDataset) As IFields
    Dim pFields As IFieldsEdit
    Set pFields = New esriGeoDatabase.Fields
    pFields.AddField CrearCampoGeometria(nombreCampoShape,
pFeatureDataset, pEsriGeometryType)
    pFields.AddField CrearCampo("OBJECTID", esriFieldTypeOID)
    ' Si se quiere añadir un campo personalizado se hace a
continuacion.
    ' P. ej. un campo de texto con el nombre de la geometría.
```


```
    ' pFields.AddField CrearCampo("NOMBRE",
esriFieldTypeString)

    Set PrepararCampos = pFields
End Function

Private Function CrearCampo(Nombre As String, tipo As
esriFieldType, Optional pGeometryType As esriGeometryType) As
IFieldEdit
    Set CrearCampo = New esriGeoDatabase.Field
    CrearCampo.Name = Nombre
    CrearCampo.AliasName = Nombre
    CrearCampo.Type = tipo
End Function

Private Function CrearCampoGeometria(Nombre As String,
pFeatureDataset As IFeatureDataset, Optional pGeometryType As
esriGeometryType) As IFieldEdit
    Set CrearCampoGeometria = New esriGeoDatabase.Field
    CrearCampoGeometria.Name = Nombre
    CrearCampoGeometria.AliasName = Nombre
    CrearCampoGeometria.Type =
esriFieldType.esriFieldTypeGeometry
    Dim pGeoDataset As IGeoDataset
    Set pGeoDataset = pFeatureDataset
    Dim pGeometryDefEdit As IGeometryDefEdit
    Set pGeometryDefEdit = New GeometryDef
    Set pGeometryDefEdit.SpatialReference =
pGeoDataset.SpatialReference
    pGeometryDefEdit.GeometryType = pGeometryType
    pGeometryDefEdit.GridCount = 1
    pGeometryDefEdit.GridSize(0) = 10
    pGeometryDefEdit.AvgNumPoints = 2
    pGeometryDefEdit.HasM = False
    pGeometryDefEdit.HasZ = True
    Set CrearCampoGeometria.GeometryDef = pGeometryDefEdit
End Function


'************************************
' CAMBIAR PUNTO POR COMA
'************************************
Function puntoporcoma(d As String)
Dim g As String
For i = 1 To Len(d)
    If Mid(d, i, 1) = "." Then
        g = g + ","
    Else
        g = g + Mid(d, i, 1)
    End If
```


```
Next i
puntoporcoma = g
End Function

'*****************
' CONTADOR
'*****************
Private Sub CmdContador_Click()
Dim contador As Long
    contador = -1
        For i = 0 To LsbX.ListCount
        contador = contador + 1
        LblContador = contador
        Next i
End Sub


'********************************
' SELECCIONAR PUNTOS FUGADOS
'********************************
Private Sub CmdSeleccpuntosfugados_Click()


Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument

'Acceso al mapa actual
Dim pMapa As IMap
Set pMapa = pMxDoc.FocusMap

'Acceso a la 1ª capa
Dim pCapa1 As IFeatureLayer
Set pCapa1 = pMapa.layer(0)

'Acceso a la 2ª capa
Dim pCapa2 As IFeatureLayer
Set pCapa2 = pMapa.layer(i)
For i = 0 To pMapa.LayerCount - 1
Next i

'Creación del objeto de geoprocesamiento
Dim GP As Object
Set GP = CreateObject("esriGeoprocessing.GpDispatch.1")
'Acceso y ejecución de la herramienta
'argumentos: capa a seleccionar, relación espacial, capa en la
que se basará el filtro espacial
'En este caso la sintáxis sería:"selecciona los elementos de
la capa 2 que intersecten con...
'los elementos de la capa 1
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
GP.SelectLayerByLocation_management pCapa2, "INTERSECT",
pCapa1
 MsgBox "Vértices fugados seleccionados."
End Sub



'*********************************
' SELECCIONAR CAPA DE ERRORES
'*********************************
Private Sub CmdSelectlayer_Click()


' selects all of the layers in the first data frame in the
toc's display view

Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
Dim pTOC As IContentsView
Set pTOC = pMxDoc.ContentsView(0) ' Display View

Dim pMaps As IMaps
Set pMaps = pMxDoc.Maps
Dim pMap As IMap

Dim i As Integer
Set pMap = pMaps.Item(0) ' first data frame
Dim pEnumLayer As IEnumLayer
Set pEnumLayer = pMap.Layers
Dim pLayer As ILayer
Set pLayer = pEnumLayer.Next

pTOC.RemoveFromSelectedItems pTOC.SelectedItem
'Do While Not pLayer Is Nothing
pTOC.AddToSelectedItems pLayer
pTOC.Refresh pLayer
'Set pLayer = pEnumLayer.Next
'Loop
   MsgBox "Capa de errores seleccionada."

End Sub



'*********************************
' DESCARGAR CAPA DE ERRORES
'*********************************
Public Sub CmdDescargarcapaerror_Click()

  ' Get the map
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
  Dim pDoc As IMxDocument
  Dim pMap As IMap
  Set pDoc = ThisDocument
  Set pMap = pDoc.FocusMap

  ' Get the selected layer or table
  Dim pSelItem As IUnknown
  Set pSelItem = pDoc.SelectedItem
  If pSelItem Is Nothing Then
    MsgBox "No Feature layer selected"
    Exit Sub
  ' Remove a Layer and refresh the map
  ElseIf TypeOf pSelItem Is IFeatureLayer Then
    pMap.DeleteLayer pDoc.SelectedLayer
    Dim pActiveView As IActiveView
    Set pActiveView = pMap
    pActiveView.Refresh
   'Remove a table
  ElseIf TypeOf pSelItem Is IStandaloneTable Then
    Dim pStTab As IStandaloneTable
    Dim pStTabColl As IStandaloneTableCollection
    Set pStTab = pSelItem
    Set pStTabColl = pMap
    pStTabColl.RemoveStandaloneTable pStTab
  Else
    MsgBox "Selected item is not a table or layer"
    Exit Sub
  End If

  ' refresh the TOC
  pDoc.UpdateContents
MsgBox "Ahora hay que ejecutar Create TIN, Edit TIN y TIN to
Ráster/TIN to Features"
End Sub


'************************************
' IMPRIMIR FICHERO TXT DE ERRORES
'************************************

Private Sub CmdImprimirficherotxterrores_Click()

Dim Linea1 As String
Dim Linea2 As String
Dim Linea3 As String
Dim Linea4 As String
Dim Linea5 As String
Dim Arch As String
Dim i As Long
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
   Arch = " C:\Program Files\Zmayor3000.txt"
   Open Arch For Output Access Write As #5

   For i = 0 To LsbLayer.ListCount - 1
    Linea1 = ""
    Linea2 = ""
    Linea3 = ""
    Linea4 = ""
    Linea5 = ""
    Linea1 = Linea1 & LsbLayer.List(i)
    Linea2 = Linea2 & LsbOID.List(i)
    Linea3 = Linea3 & LsbX.List(i)
    Linea4 = Linea4 & LsbY.List(i)
    Linea5 = Linea5 & LsbZ.List(i)
   Print #5, "Capa: " & Linea1, "OID: " & Linea2, "X: " &
Linea3, "Y: " & Linea4, "Z: " & Linea5
   Next i
   Close #5
   Close #4
   Close #3
   Close #2
   Close #1

   MsgBox "El fichero Zmayor3000.txt de errores ha sido
generado en: " & Arch & ". Ya tiene localizados los elementos
a corregir."
End Sub
```

Formulario FrmZfugados:

```
'*************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS
'*************************************
Public Sub CmdImprVertFCPtos_click()
ImprimirVerticesFeatureClassPuntos
End Sub
Public Sub ImprimirVerticesFeatureClassPuntos(NUMLAYER As
Integer, LAYERFINAL As Integer)
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features

    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
    While Not pFeature Is Nothing
        Debug.Print vbTab + "Feature OID " +
CStr(pFeature.OID)
        Dim pPoint As IPoint
        Set pPoint = pFeature.Shape

        If pPoint.Z >= 3000 Or pPoint.Z <= 0 Then


            Me.LsbLayer.AddItem pFeatureLayer.Name
            Me.LsbOID.AddItem pFeature.OID
            Me.LsbX.AddItem pPoint.X
            Me.LsbY.AddItem pPoint.Y
            Me.LsbZ.AddItem pPoint.Z


           Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
           End If

           Set pFeature = pFeatureCursor.NextFeature
            Wend

         Set pFeature = pFeatureCursor.NextFeature


End Sub
'*********************************************
' IMPRIMIR VÉRTICES CAPAS DE LINEAS Y POLÍGONOS
'*********************************************
Public Sub CmdImprVertFCLinPol_click()
ImprimirVerticesFeatureClassLinPol
End Sub
Public Sub ImprimirVerticesFeatureClassLinPol(NUMLAYER As
Integer, LAYERFINAL As Integer)
  ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
```


```
    Set pFeatureLayer = pMap.layer(NUMLAYER)
    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)
    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature

    While Not pFeature Is Nothing
        Debug.Print "Feature OID " + CStr(pFeature.OID)
        ' Accede as la geometría con la interfaz
IPointCollection
        Dim pPointCollection As IPointCollection
        Set pPointCollection = pFeature.Shape
            Dim i As Integer
            For i = 0 To pPointCollection.PointCount - 1
            Dim pPoint As IPoint
            Set pPoint = pPointCollection.Point(i)

            If pPoint.Z >= 3000 Or pPoint.Z <= 0 Then

             Me.LsbLayer.AddItem pFeatureLayer.Name
             Me.LsbOID.AddItem pFeature.OID
             Me.LsbX.AddItem pPoint.X
             Me.LsbY.AddItem pPoint.Y
             Me.LsbZ.AddItem pPoint.Z

            Debug.Print vbTab + "Vertice " + CStr(i) + " - X =
" + CStr(pPoint.X) + " Y = " + CStr(pPoint.Y) + " Z = " +
CStr(pPoint.Z)
            End If

        Next i
        Set pFeature = pFeatureCursor.NextFeature
   Wend

End Sub


'*********************************************************
' IMPRIMIR VÉRTICES CAPAS DE PUNTOS, LINEAS Y POLÍGONOS
'*********************************************************

Private Sub CmdImprVertFCPtosLinPol_Click()
ImprimirVertices
End Sub
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
Public Sub ImprimirVertices()
    ' Accede al documento
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Dim c As Integer
    Dim pcapaclass As IFeatureClass

    Dim f As Integer


    'bucle para recorrer todas las capas
     For c = 0 To pMap.LayerCount - 1

    ' bucle para elegir el código, según sea la capa de ptos o
de líneas o polígonos

     Set pFeatureLayer = pMap.layer(c)
     Set pcapaclass = pFeatureLayer.FeatureClass

              If pcapaclass.ShapeType = esriGeometryPoint Then
             ImprimirVerticesFeatureClassPuntos c, f
             MsgBox "Terminada la capa de puntos: " &
pFeatureLayer.Name
              Else
             ImprimirVerticesFeatureClassLinPol c, f
             MsgBox "Terminada la capa de líneas o polígonos: "
& pFeatureLayer.Name

        End If

       Next c
       MsgBox "Terminado"

End Sub


'************************************
' IMPRIMIR VÉRTICES CON CURSOR
'************************************
Public Sub CmdImprimirVerticesConCursor_click()
ImprimirVerticesConCursor
End Sub
Public Sub ImprimirVerticesConCursor()
  ' Accede al documento
    Dim pMxDoc As IMxDocument
```


```
    Set pMxDoc = ThisDocument
    ' Obtiene el primer mapa (en base cero)
    Dim pMap As IMap
    Set pMap = pMxDoc.Maps.Item(0)
    ' Obtiene la primera layer
    Dim pFeatureLayer As IFeatureLayer
    Set pFeatureLayer = pMap.layer(0)

    ' Crea la nueva feature class donde se van a guardar los
vértices.
    Dim pFeatureClassDestino As IFeatureClass
    Set pFeatureClassDestino =
CrearFeatureClass("Zfugados_pun", esriGeometryPoint,
pFeatureLayer.FeatureClass.FeatureDataset)
    ' Crea una nueva layer, para que nos muestre en ArcMap la
featureclass creada
        Dim pFeatureLayerNueva As IFeatureLayer
        Set pFeatureLayerNueva = New FeatureLayer
    ' Se hace que la layer apunte a la featureclass
        Set pFeatureLayerNueva.FeatureClass =
pFeatureClassDestino
        ' Se le da nombre
        pFeatureLayerNueva.Name =
pFeatureClassDestino.AliasName
    ' Y se añade al mapa
        pMap.AddLayer pFeatureLayerNueva
    ' Se obtiene el cursor de inserción de la feature class
nueva.
    Dim pCursorInsercion As IFeatureCursor
    Set pCursorInsercion = pFeatureClassDestino.Insert(True)

    ' Se crea un feature buffer para ir guardando los
vertices. El feature buffer es una feature temporal,
    ' que no existe físicamente en la feature class hasta que
no se guarda con el cursor de inserción.
    Dim pFeatureBuffer As IFeatureBuffer
    Set pFeatureBuffer =
pFeatureClassDestino.CreateFeatureBuffer

    ' Crea un cursor de busqueda de la feature class contenida
en la layer
    Dim pFeatureCursor As IFeatureCursor
    Set pFeatureCursor =
pFeatureLayer.FeatureClass.Search(Nothing, True)

    ' Recorre las features
    Dim pFeature As IFeature
    Set pFeature = pFeatureCursor.NextFeature
   ' While Not pFeature Is Nothing
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
        ' Accede as la geometría con la interfaz
IPointCollection
        'Dim pPointCollection As IPointCollection
        'Set pPointCollection = pFeature.Shape

        ' Recorre los vértices
        Dim i As Long
        For i = 0 To LsbOID.ListCount - 1

            Dim pPoint As IPoint

            'Set pPoint = pPointCollection.Point(0)
            Set pPoint = pFeature.Shape

            ' Al crear la nueva feature class, se a definido
como que tiene Zs en la línea:
            ' pGeometryDefEdit.HasZ = True
            ' Así que en cualquier geometría que vayamos a
añadir tenemos que asegurarnos de que tenga Z

              pPoint.X = puntoporcoma(Me.LsbX.List(i))
              pPoint.Y = puntoporcoma(Me.LsbY.List(i))
              pPoint.Z = puntoporcoma(Me.LsbZ.List(i))
            EstablecerZ pPoint

            ' Asigna el vértice a la geometría de la nueva
feature
            Set pFeatureBuffer.Shape = pPoint

            ' Guarda la feature en la tabla
            pCursorInsercion.InsertFeature pFeatureBuffer

        Next i

        Set pFeature = pFeatureCursor.NextFeature

    ' Guarda los cambios pendientes
    pCursorInsercion.Flush
      MsgBox "Vértices copiados a capa."
End Sub

Private Function EstablecerZ(pPoint As IPoint)
    Dim pZAware As IZAware
    Set pZAware = pPoint
    ' Establece que la geometria pueda tener Zs
    pZAware.ZAware = True
    ' Comprueva que la Z sea válida
    If pZAware.ZSimple = False Then
        ' Si no lo es la deja a cota cero
        pPoint.Z = 0
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
    End If
End Function

Public Function CrearFeatureClass(Nombre As String, geomType
As esriGeometryType, pFeatureDataset As IFeatureDataset) As
IFeatureClass
On Error GoTo EH
    ' El nombre del campo shape se define aquí ya que se va a
usar en más de un sitio.
    Dim nombreCampoShape As String
    nombreCampoShape = "Shape"
    Dim pCLSID As UID
    Set pCLSID = New UID
    pCLSID.Value = "esriGeoDatabase.Feature"
    Dim pFields As IFields
    Set pFields = PrepararCampos(geomType, nombreCampoShape,
pFeatureDataset)
    Set CrearFeatureClass =
pFeatureDataset.CreateFeatureClass(Nombre, pFields, pCLSID,
Nothing, esriFTSimple, nombreCampoShape, "")
    Exit Function
EH:
    ' En caso de error muestra un mensaje con la descripción.
    MsgBox Err.Description, vbInformation, "CrearFeatureClass"
End Function

' Aquí se definen los campos que tendrá la capa, en este caso
solo van a ser el campo 'Shape' y el 'OBJECTID'
' que son los campos mínimos que puede tener una feature
class. Dentro del campo Shape se guarda la geometría,
' en el campo OBJECTID un identificador único para esta.
Private Function PrepararCampos(pEsriGeometryType As
esriGeometryType, nombreCampoShape As String, pFeatureDataset
As IFeatureDataset) As IFields
    Dim pFields As IFieldsEdit
    Set pFields = New esriGeoDatabase.Fields
    pFields.AddField CrearCampoGeometria(nombreCampoShape,
pFeatureDataset, pEsriGeometryType)
    pFields.AddField CrearCampo("OBJECTID", esriFieldTypeOID)
    ' Si se quiere añadir un campo personalizado se hace a
continuacion.
    ' P. ej. un campo de texto con el nombre de la geometría.
    ' pFields.AddField CrearCampo("NOMBRE",
esriFieldTypeString)

    Set PrepararCampos = pFields
End Function
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
Private Function CrearCampo(Nombre As String, tipo As
esriFieldType, Optional pGeometryType As esriGeometryType) As
IFieldEdit
    Set CrearCampo = New esriGeoDatabase.Field
    CrearCampo.Name = Nombre
    CrearCampo.AliasName = Nombre
    CrearCampo.Type = tipo
End Function

Private Function CrearCampoGeometria(Nombre As String,
pFeatureDataset As IFeatureDataset, Optional pGeometryType As
esriGeometryType) As IFieldEdit
    Set CrearCampoGeometria = New esriGeoDatabase.Field
    CrearCampoGeometria.Name = Nombre
    CrearCampoGeometria.AliasName = Nombre
    CrearCampoGeometria.Type =
esriFieldType.esriFieldTypeGeometry
    Dim pGeoDataset As IGeoDataset
    Set pGeoDataset = pFeatureDataset
    Dim pGeometryDefEdit As IGeometryDefEdit
    Set pGeometryDefEdit = New GeometryDef
    Set pGeometryDefEdit.SpatialReference =
pGeoDataset.SpatialReference
    pGeometryDefEdit.GeometryType = pGeometryType
    pGeometryDefEdit.GridCount = 1
    pGeometryDefEdit.GridSize(0) = 10
    pGeometryDefEdit.AvgNumPoints = 2
    pGeometryDefEdit.HasM = False
    pGeometryDefEdit.HasZ = True
    Set CrearCampoGeometria.GeometryDef = pGeometryDefEdit
End Function


'************************************
' CAMBIAR PUNTO POR COMA
'************************************
Function puntoporcoma(d As String)
Dim g As String
For i = 1 To Len(d)
    If Mid(d, i, 1) = "." Then
        g = g + ","
    Else
        g = g + Mid(d, i, 1)
    End If
Next i
puntoporcoma = g
End Function

'*****************
' CONTADOR
```


```
'*****************
Private Sub CmdContador_Click()
Dim contador As Long
    contador = -1
        For i = 0 To LsbX.ListCount
        contador = contador + 1
        LblContador = contador
        Next i
End Sub


'*********************************
' SELECCIONAR PUNTOS FUGADOS
'*********************************
Private Sub CmdSeleccpuntosfugados_Click()


Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument

'Acceso al mapa actual
Dim pMapa As IMap
Set pMapa = pMxDoc.FocusMap

'Acceso a la 1ª capa
Dim pCapa1 As IFeatureLayer
Set pCapa1 = pMapa.layer(0)

'Acceso a la 2ª capa
Dim pCapa2 As IFeatureLayer
Set pCapa2 = pMapa.layer(i)
For i = 0 To pMapa.LayerCount - 1
Next i

'Creación del objeto de geoprocesamiento
Dim GP As Object
Set GP = CreateObject("esriGeoprocessing.GpDispatch.1")
'Acceso y ejecución de la herramienta
'argumentos: capa a seleccionar, relación espacial, capa en la
que se basará el filtro espacial
'En este caso la sintáxis sería:"selecciona los elementos de
la capa 2 que intersecten con...
'los elementos de la capa 1
GP.SelectLayerByLocation_management pCapa2, "INTERSECT",
pCapa1
 MsgBox "Vértices fugados seleccionados."
End Sub
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
'*********************************
' SELECCIONAR CAPA DE ERRORES
'*********************************
Private Sub CmdSelectlayer_Click()


' selects all of the layers in the first data frame in the
toc's display view

Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
Dim pTOC As IContentsView
Set pTOC = pMxDoc.ContentsView(0) ' Display View

Dim pMaps As IMaps
Set pMaps = pMxDoc.Maps
Dim pMap As IMap

Dim i As Integer
Set pMap = pMaps.Item(0) ' first data frame
Dim pEnumLayer As IEnumLayer
Set pEnumLayer = pMap.Layers
Dim pLayer As ILayer
Set pLayer = pEnumLayer.Next

pTOC.RemoveFromSelectedItems pTOC.SelectedItem
'Do While Not pLayer Is Nothing
pTOC.AddToSelectedItems pLayer
pTOC.Refresh pLayer
'Set pLayer = pEnumLayer.Next
'Loop
   MsgBox "Capa de errores seleccionada."

End Sub



'*********************************
' DESCARGAR CAPA DE ERRORES
'*********************************
Public Sub CmdDescargarcapaerror_Click()

  ' Get the map
  Dim pDoc As IMxDocument
  Dim pMap As IMap
  Set pDoc = ThisDocument
  Set pMap = pDoc.FocusMap

  ' Get the selected layer or table
  Dim pSelItem As IUnknown
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
  Set pSelItem = pDoc.SelectedItem
  If pSelItem Is Nothing Then
    MsgBox "No Feature layer selected"
    Exit Sub
  ' Remove a Layer and refresh the map
  ElseIf TypeOf pSelItem Is IFeatureLayer Then
    pMap.DeleteLayer pDoc.SelectedLayer
    Dim pActiveView As IActiveView
    Set pActiveView = pMap
    pActiveView.Refresh
   'Remove a table
  ElseIf TypeOf pSelItem Is IStandaloneTable Then
    Dim pStTab As IStandaloneTable
    Dim pStTabColl As IStandaloneTableCollection
    Set pStTab = pSelItem
    Set pStTabColl = pMap
    pStTabColl.RemoveStandaloneTable pStTab
  Else
    MsgBox "Selected item is not a table or layer"
    Exit Sub
  End If

  ' refresh the TOC
  pDoc.UpdateContents
MsgBox "Ahora hay que ejecutar Create TIN, Edit TIN y TIN to
Ráster/TIN to Features"
End Sub


'************************************
' IMPRIMIR FICHERO TXT DE ERRORES
'************************************

Private Sub CmdImprimirficherotxterrores_Click()

Dim Linea1 As String
Dim Linea2 As String
Dim Linea3 As String
Dim Linea4 As String
Dim Linea5 As String
Dim Arch As String
Dim i As Long

   Arch = "G C:\Program Files\Zfugados.txt"
   Open Arch For Output Access Write As #5

   For i = 0 To LsbLayer.ListCount - 1
    Linea1 = ""
    Linea2 = ""
    Linea3 = ""
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

```
    Linea4 = ""
    Linea5 = ""
    Linea1 = Linea1 & LsbLayer.List(i)
    Linea2 = Linea2 & LsbOID.List(i)
    Linea3 = Linea3 & LsbX.List(i)
    Linea4 = Linea4 & LsbY.List(i)
    Linea5 = Linea5 & LsbZ.List(i)
   Print #5, "Capa: " & Linea1, "OID: " & Linea2, "X: " &
Linea3, "Y: " & Linea4, "Z: " & Linea5
   Next i
   Close #5
   Close #4
   Close #3
   Close #2
   Close #1

   MsgBox "El fichero Zfugados.txt de errores ha sido generado
en: " & Arch & ". Ya tiene localizados los elementos a
corregir."
End Sub
```


Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo

Iciar Martínez González
Máster en Teledetección y Sistemas de Información Geográfica
Escuela Politécnica de Mieres – Universidad de Oviedo