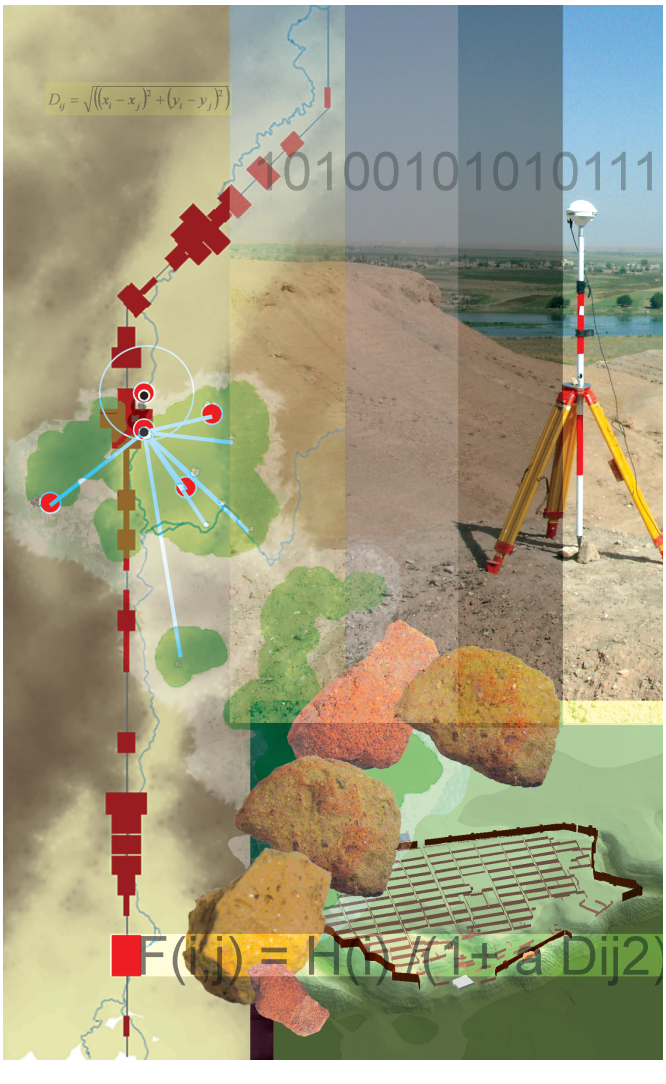
ECOLE THEMATIQUE

Géomatique, Analyse & Modélisation Spatiale en Archéologie

Réseau ISA
Information Spatiale et Archéologie
http://www.univ-tours.fr/isa

du 12 au 17 septembre 2005
à Messigny et Vantoux (Côte d'Or)

Support de cours - **Programmation Visual Basic pour ArcGIS (T. Lohro, X. Rodier)**



CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
Chrono-Ecologie, UMR 6565, Besançon
Archéologie, Culture et Société, UMR 5594, Dijon

UNIVERSITÉ DE BOURGOGNE

Maison des sciences de l'homme Dijon
Patrimoines, Espaces, Mutations

UMS 2913
MSH LEDOUX

Institut national de recherches archéologiques préventives

UNIVERSITÉ DE FRANCHE-COMTÉ

Atelier programmation ArcGis en visual basic.

Introduction :

Dans cet atelier notre ambition n'est pas de faire de vous en 3 heures de parfaits connaisseurs du langage Visual basic appliqué à Arcgis. Il va surtout s'agir de vous permettre d'en connaître les grands principes de fonctionnement (l'interface VBE, la norme COM, le modèle d'objet) ainsi que les différentes manières de s'en servir depuis la personnalisation de son document de travail par la récupération de fichiers déjà développés (.dll, .exe…), en passant par la réalisation de calculs et jusqu'à l'écriture de son propre code.

Qu'il s'agisse d'ArcInfo ou Arcgis il y a deux manières de personnaliser ces applications :

- Soit en interne par la réalisation de script
- Soit en externe en utilisant un environnement de développement standard

ArcInfo Workstation :

Les scripts sont développés en AML (Arc Macro Langage)

L'interface de développement est le kit ODE (Open Development Environnement) qui permet de mettre à disposition des bibliothèques d'objets (DLL, activeX). Le langage de développement est le visual basic ou le C++.

Arcgis Desktop :

Les scripts sont développés en VBA (Visual Basic for Application).

Les développements externes sont réalisés à partir de Visual basic qui permet l'utilisation de composant active X (DLL ou OCX).

Dans les deux cas on va utiliser la bibliothèque d'objets ArcObject, mais avec VBA même si la personnalisation de l'application peut-être très poussée, le code est dit interprété donc beaucoup moins rapide que celui généré par Visual basic qui est dit compilé et qui permet de créer des applications indépendantes.

Remarque : *l'utilisation de bibliothèque ArcObject pour la création de scripts et d'applications autonomes nécessite bien évidemment de posséder une licence Arcgis (Arcview, Arceditor ou Arcinfo. Il est aussi nécessaire de posséder ces licences pour utiliser ces applications*

Dans cet atelier nous nous limiterons à l'environnement ArcGis desktop et à l'utilisation de son module de programmation Visual Basic Editor (VBE).

I. Généralités :

A.Définitions :

1.Notion d'objet :

Visual basic est un langage dit « **orienté objet** ».

L'objet est donc l'élément de base de ce langage. Il peut s'agir d'un contrôle, d'un formulaire, d'une mise en page, etc.…

Chacun d'entre eux se caractérise par des **propriétés**, des **méthodes** et des **événements**.

Une **propriété** est un attribut de l'objet qui définit une de ses caractéristiques (le nom, la couleur, la position, la taille, etc.…)

Remarque : *Selon leur nature certaines propriétés ne pourront être définies que pendant la phase de conception, d'autres pendant la phase de conception et d'exécution, d'autres encore ne pourront être que lues. D'une manière générale elles sont toutes lisibles.*

Une **méthode** est une action qu'un objet peut réaliser, par exemple *Add* associée à un contrôle de type « listbox » va permettre d'ajouter un élément à la liste qu'il présente.

Les **événements** sont des actions que reconnaît l'objet. Un bouton détecte le clic de la souris

2.Notion de classe :

Tous les objets qui possèdent les mêmes propriétés et les mêmes méthodes, sont des instances d'une même entité qu'on appelle **classe**. On parle de classe d'objets.

3.Le modèle d'objet :

Les relations qui existent entre les méthodes et les propriétés appartenant à une même classe et celles entre les différentes classes constituent le **modèle** d'objet et sont représentées par un diagramme qui utilise la modélisation UML. On parle **d'Object Model Diagram (O.M.D.)**

Chaque application possède un certain nombre d'objets qui lui sont propre, on parle de bibliothèque d'objets. Celle d'Arcgis est appelée **ARCOBJECTS.**

4. Structuration du Code visual basic :

Les instructions :

Les **instructions** sont l'unité de base du langage visual basic. Elles sont classées en trois grandes catégories qui sont :

- les instructions de **déclarations** : ***Private*** *myVar* ***As*** *string*
- les instructions d'**affectations** : *MyVar* **=** *« Ecole thématique »*
- les instructions **exécutables** :

```
If MyVar = Empty then
    msgbox "Vous n'avez rien saisi"
else
msgbox « Vous assistez à l' » & Myvar
end
```

Les procédures :

Une instruction peut rester seule ou être combinée à d'autres pour constituer une **procédure** afin de pouvoir être exécutée(s).
Il existe trois catégories de procédures :

- **les procédures événementielles**. Elles codent pour les événements correspondant à des actions qui affectent les objets. (click par exemple)

```
Private Sub mybouton_Click()
End Sub
```

- **les procédures routinières** : Il s'agit d'une série d'instructions qui n'est pas rattachée à un objet contrairement au cas précédent et qu'il va falloir appeler pour qu'elle s'exécute. Ceci sert notamment pour toutes les actions qui se produisent à différents endroits afin de ne pas avoir à saisir plusieurs fois le code qui leur correspond.

```
Private Sub Message_de_fin()
End Sub
```

- **les fonctions** : Il s'agit de procédures routinières qui peuvent renvoyer une valeur.

```
Private Sub conversion_m²_hectare()
End Sub
```

Les modules :

Ces procédures ou fonctions présentes sont organisées en **modules** selon leur rôle et leur relation au sein du document Arcgis.
Il y a 3 catégories de module :

- **Le module de feuille** : Chaque formulaire présent dans le document Arcgis possède un module de ce type. Ce module regroupe toutes les procédures événementielles associées au formulaire ainsi que celles codant pour les contrôles contenus par ce formulaire. Il contient en plus les données associées aux contrôles.
- **Le module standard** : Il réunit toutes les procédures et fonctions qui ne sont pas rattachées à un formulaire ou un contrôle. Il ne contient que du code et pas de donnée car pas de contrôle..

- **Le module de classe** : Il correspond à du code et des données associées à une classe d'objets qui n'ont pas d'interface graphique ou qui ont été créés par le développeur.

Le code contenu dans ces trois modules peut-être exporté séparément pour resservir dans d'autres documents Arcgis.
Les modules de feuille génèrent des fichiers « **.frm** », les modules standards génèrent des fichiers « **.bas** » et les modules de classe des fichiers « **.cls** ».

B. L'interface Visual Basic editor :

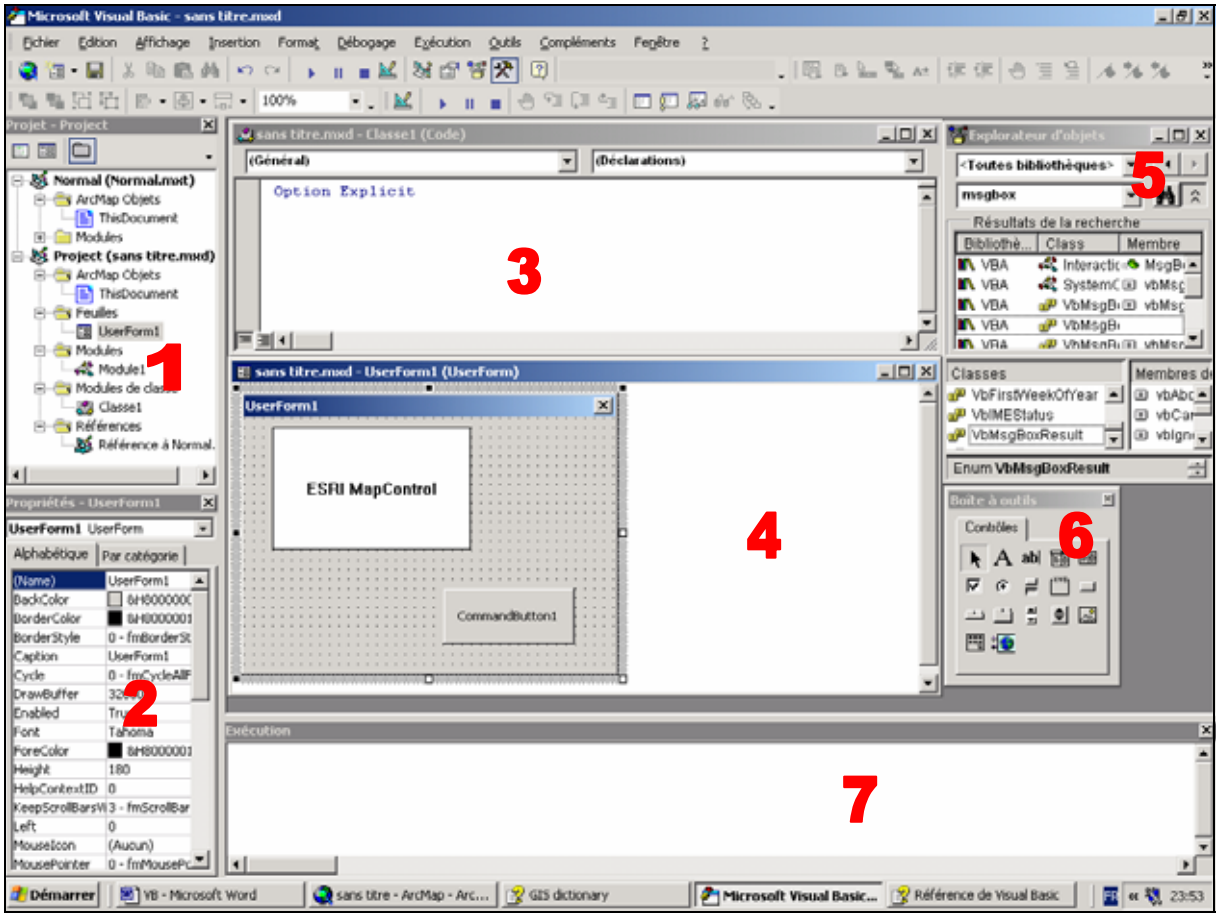


Figure 1.

L'éditeur visual basic représenté figure 1/ci-dessus est accessible dans ArcMap à partir du menu Outils/Macros/Editeurs_visual_basic.
Il est composé de 7 fenêtres principales :

- **L'explorateur de projet (1)** : Elle énumère les différents modules qui sont contenus dans le document Arcgis et qui sont autant de fichiers qui constituent le projet.
- **La fenêtre de propriété (2)** : Elle énumère les propriétés de l'objet sélectionné, dans le cas présent il s'agit du formulaire.
- **La fenêtre de code (3)** : Elle permet de taper le texte du code du projet. Il en existe une pour chaque module.
- **La fenêtre de conception de feuille (4)** : Elle permet la réalisation de l'interface graphique de chaque formulaire. Il y en a une par formulaire.
- **La fenêtre explorateur d'objet (5)** : Elle permet de parcourir les objets et leur méthode disponible pour notre projet et offre une aide précieuse pour la rédaction du code.
- **La boite à outil (6)** : Elle possède tous les contrôles nécessaires à la conception d'un formulaire.
- **La fenêtre d'exécution (7)** : Elle permet de tester les lignes de code tapées. Elle constitue avec les fenêtres variables locales et espions des outils très utiles pour le débogage de l'application (ces 2 fenêtres ne sont pas présentes sur la figure 1).

Remarque : Dans le menu outils/options de l'éditeur de visual basic il est possible de personnaliser l'éditeur en y ajoutant des fonctionnalités d'aide à la rédaction du code, telles que la saisie semi-automatique ou l'obligation de déclaration des variables.

II. Utilisation de code déjà rédigé et rédaction de macros :

A. Enregistrement des modifications :

Avant d'entamer une personnalisation de notre application il est important de savoir comment la sauvegarder.
Il existe trois niveaux d'enregistrement des personnalisations effectuées dans l'application :

- Le **niveau document Arcmap** en cours, qui correspond au document ArcMap ouvert. Les modifications apportées n'existeront que dans ce seul fichier « **.mxd** ». Ceci se fait en utilisant le menu fichier/enregistrer sous et en choisissant comme type de fichier « **ArcMap documents** ».

- Le **niveau modèle de base** qui est un document qui possède des mises en pages prédéfinies et qui permet de reproduire de nouvelles cartes rapidement. Tous les nouveaux documents qui utiliseront ce modèle se verront affecter ses modifications. Ceci se fait en utilisant le menu fichier/enregistrer sous et en choisissant comme type de fichier « **ArcMap template** ». Ils ont comme extension **« .mxt »**.

- **Le niveau modèle normal**, modèle spécial qui est automatiquement chargé à l'ouverture d'ArcMap. Ceci se fait en utilisant le menu fichier enregistrer sous. Ainsi tous les documents ArcMap ouverts quel que soit le modèle utilisé posséderont les modifications. Ils ont comme extension « **.mxt** », la même que précédemment car il s'agit aussi d'un modèle et non pas d'un document.

B. Personnalisation du document :

Dans cette partie nous allons voir différents types de fichiers pouvant contenir du code en visual basic et les manières de l'incorporer à notre application.
De nombreux fichiers de ce type sont en ligne sur le site d'esrifrance **http://support.esrifrance.fr**

1. Création d'une barre d'outil :

La première chose à faire quand on veut rajouter une fonctionnalité à son application est de réfléchir à l'endroit ou elle va se positionner. Il peut s'agir d'une barre d'outil, d'un menu, etc.…

Exercice : Création d'une barre d'outil

1. Lancer ArcMap
2. Ouvrir un nouveau document vide.
3. L'enregistrer sous en l'appelant Ecole_thématique.
4. Cliquer sur outils/personnaliser.
5. Cliquer sur nouvelle dans l'onglet barre d'outil.
6. Saisir un nom « Ma barre d'outil à moi » et choisir de l'enregistrer dans le document Ecole thematique.
7.Positionner la nouvelle barre d'outil dans la fenêtre ArcMap.

Cette barre d'outil va servir à positionner de nouvelles fonctionnalités dans notre document.

2. Fichiers exe ou dll :

Ces deux types de fichiers sont réalisés à l'aide du logiciel visual basic, et ne peuvent en aucun cas être faits avec l'éditeur visual basic de ArcMap. Il s'agit de code compilé.

*Exercice : Création d'un contrôle à partir d'un fichier « **. dll** ».*

1. *Sous le répertoire Atelier_Vb/fichiers_téléchargés_zippés, ouvrir le fichier « outildeflux.zip »*
2. *Le décompresser sous le répertoire Atelier_VB/fichiers_dezippés.*
3. *Dans ArcMap, cliquer sur outils/personnaliser.*
4. *Sélectionner l'onglet commande et sélectionner dans la liste l'outil Uicontrols.*
5. *Cliquer sur ajouter depuis un fichier.*
6. *Parcourir l'arborescence et sélectionner le fichier outildeflux.dll que vous venez de décompresser.*
7. *Une nouvelle commande apparaît dans la boîte de dialogue.*
8. *Sélectionner la et glisser la jusque sur la barre de contrôle que vous avez créée dans l'exercice précédent.*
9. *Enregistrer votre document.*
10. *Cette nouvelle fonctionnalité est désormais opérationnelle et fait partie de votre document « Ecole thématique ».*

*Exercice : Création d'un contrôle à partir d'un fichier « **. exe** ».*

1. *Sous le répertoire Atelier_Vb/fichiers_téléchargés_zippés, ouvrer le fichier « transformationDAO.zip ».*
2. *Le décompresser sous le répertoire Atelier_VB/fichiers_dézippés.*
3. *Cliquer sur le fichier install.*
4. *Cliquer sur outils/personnaliser.*
5. *Parcourir le menu déroulant des barres d'outil, trouver la barre d'outil géoréférencer un fichier DAO.*
6. *Cocher-la. Elle apparaît prête à l'emploi. Il vous suffit de la positionner dans votre fenêtre ArcMap et puis d'enregistrer votre document.*

3. Fichiers.bas.frm et.cls :

Il s'agit de fichiers qui peuvent être générés par l'éditeur visual basic.
Ces 3 types de fichiers correspondent aux 3 types de modules de code décrits précédemment. Ils ont été rédigés et enregistrés dans un de ces 3 modules puis exportés.

*Exemple : Création d'un contrôle à partir d'un fichier « **.bas** ».*

1. *Sous le répertoire Atelier_VB/fichiers_téléchargés_zippés, ouvrir le fichier « Dropshadows.zip ».*
2. *Le décompresser sous le répertoire Atelier_VB/fichiers_dézippés.*
3. *Dans ArcMap, cliquer sur outils/macros/editeur visuel basic.*
4. *Dans la fenêtre de projet, faire un clique droit sur module et sélectionner importer un fichier.*
5. *Parcourir l'arborescence et sélectionner le fichier « dropshadows.bas ». Dans la fenêtre de projet un nouveau module est apparu, il s'agit d'un module standard correspondant à l'extension du fichier importé, .bas et portant le même nom. Si vous double-cliquer dessus vous verrez apparaître la fenêtre de code lui correspondant.*
6. *Fermer l'éditeur visual basic.*
7. *Cliquer sur outils/personnaliser.*
8. *Sélectionner l'onglet commande et faire défiler la liste des contrôles jusqu'à macro.*
9. *Une commande apparaît portant le nom du fichier.bas.*
10. *Il suffit de la sélectionner et de la faire glisser jusqu'à la barre d'outil.*
11. *Sauvegarder ensuite votre document ArcMap.*

Le fonctionnement est le même avec les fichiers « .frm » et « .cls »

C. Rédaction d'une macro :

Jusqu'à présent toutes les fonctionnalités rajoutées à notre document ont été réalisées à partir de code récupéré.

Dans cette partie il va s'agir de rédiger son propre code et de l'associer à la fenêtre ArcMap.
Ce code correspondant à une série d'instructions utilisée comme une commande unique, est appelé aussi une **Macro**.

Exercice Rédaction d'une macro :

1. *Dans Arcmap, cliquer sur outils/macros/macros.*
2. *Dans la boite de dialogue saisir le nom de votre macro, indispensable avant de pouvoir la créer. Taper « Macro.date_du_jour.*
3. *Cliquer sur créer.*
4. *L'éditeur visual basic s'est ouvert et un nouveau module intitulé macro. Sa fenêtre de code s'est également ouverte et il pointe sur la procédure date_du_jour.*
5. *Taper dans cette procédure le code :*
   Msgbox « Aujourd'hui nous sommes le » & « Format(Date, « ddd d mmmm YYYY) »
6. *Enregistrer le tout et rebasculer dans ArcMap.*
7. *Cliquer sur outils/personnaliser et dans l'onglet commandes.*
8. *Sélectionner Macro et dans la fenêtre de droite doit apparaître la macro « date_du_jour ».*
9. *Faites-la glisser jusqu'à votre barre d'outils.*
10. *Cliquer sur le nouveau bouton et la macro s'exécute.*

D. Utilisation du langage Visual Basic avec la calculatrice :

Lors de la mise à jour des valeurs d'une table, il est possible d'utiliser la calculatrice. Celle-ci permet de rédiger des formules de calcul utilisant les valeurs des autres champs de la table, des tests conditionnels ou encore des fonctions internes d'Arcgis. L'utilisateur peut rédiger ses propres formules ou charger des formules préalablement enregistrées, fichiers « **.cal** ».

*Exercice: Générer et utiliser des fichiers de type « **.cal** ».*

1. *Ouvrir un nouveau document ArcMap.*
2. *Ajouter à ce document la couche « country.shp ».*
3. *Ouvrir sa table et y ajouter 3 champs de type numérique d'une longueur de 10 avec 2 chiffres après la virgule, intitulés : X, Y et Area.*
4. *Sélectionner le champ area et lancer la calculatrice.*
5. *Taper dans la première fenêtre le code suivant :*

   Dim dblArea as double
   Dim pArea as IArea
   Set pArea = [shape]
   dblArea = pArea.area
6. *Taper dans la deuxième fenêtre le nom de la variable*

   DblArea
7. *La surface de chaque polygone a ainsi été calculée.*
8. *Enregistrer ensuite ce code sous le répertoire atelier_VB/formules en l'intitulant Area.*
9. *Fermer la calculatrice, puis l'ouvrir à nouveau et charger la formule area en cliquant sur le bouton « charger ». L'expression tapée précédemment a donc bien été enregistrée et est réutilisable. L'extension de cette formule area est « **.cal** »*
10. *Recommencer avec les champs X et Y en rédigeant les formules et en les sauvegardant :*

    Dim pArea as IArea
    Dim pPoint as IPoint
    Dim dblX as Double
    Set pArea = [Shape]
    Set pPoint = pArea.Centroid
    dblX = pPoint.X

    Dim pArea as IArea
    Dim pPoint as IPoint
    Dim dblY as Double
    Set pArea = [Shape]

Set pPoint = pArea.Centroid
dblY = pPoint.Y

III. La Norme COM et le diagramme du modèle d'objet Arcobject :

A. La Norme COM :

La bibliothèque d'objet d'Arcgis ArcObject s'appuie sur la technologie **Microsoft's Component Object Model.(COM)**

La norme COM n'est pas un langage de programmation mais un protocole. Elle permet de mettre en relation les classes d'objets constitutives de la bibliothèque d'objet d'un logiciel, mais aussi les classes des différentes bibliothèques des différents logiciels installés sur votre ordinateur entre elles pour peu qu'ils soient compatibles avec cette norme.

L'utilisateur va donc pouvoir construire de nouvelles classes d'objets qui pourront être à leur tour échangées. Ceci indépendamment de tout langage de programmation, ce qui assure une pérennité de l'application qui les utilise et lui garantissant la possibilité d'évoluer au-delà des évolutions des langages et des logiciels.

Il s'agit d'une approche orientée objet et non objet comme arcview3 qui impose tout développement avec un langage propriétaire Avenue.

Avec cette norme l'utilisateur ne « traite » pas avec les objets directement mais avec leurs méthodes et propriétés qui sont réunies au sein d'une même classe. L'accessibilité à ces méthodes et propriétés se fait par l'intermédiaire d'**interfaces**.

Une interface réunit les propriétés et les méthodes d'une classe d'objet en fonction de leur caractéristique. Chaque classe **implémente** (possède) plusieurs interfaces qui comprennent chacune une partie de leurs méthodes et de leurs propriétés en les regroupant de façon de façon logique.

Une fois « récupérées » ces méthodes et propriétés pourront être exploitées à travers un langage de développement choisi par l'utilisateur (Visual basic, Delphi, etc.…) et ainsi permettront d'**instancier** (créer) un objet.

Si le besoin se fait sentir de faire évoluer les propriétés ou méthodes d'un objet, les nouvelles n'effaceront pas les précédentes, garantissant la pérennité des développements antérieurs, mais seront réunies au sein d'une nouvelle interface afin d'être utilisables.

Il peut arriver que plusieurs classes d'objets partagent une même interface, on parle alors de **polymorphisme**.

B. Le Diagramme du modèle d'objet Arcobject :

La bibliothèque d'objet d'ArcGis, ArcObject, est composée de nombreuses classes d'objet. Les relations qu'entretiennent ces classes entre elles et les méthodes et propriétés que leurs interfaces possèdent sont représentées à l'aide du **ArcObject Object Model Diagram. (OMD)** Compte tenu de sa grande taille et de sa complexité, il a été découpé par grandes classes d'objets. Ces différents OMD sont fournis avec Arcgis au format « .pdf » et sont stockés sous le répertoire » C:\…\ArcGIS\DeveloperKit\Diagrams\Desktop.

Ils sont représentés selon la nomenclature **UML. (Unified Modeling Language)** Il est indispensable de savoir les lire et d'y circuler si on veut faire du développement avec ArcObject.

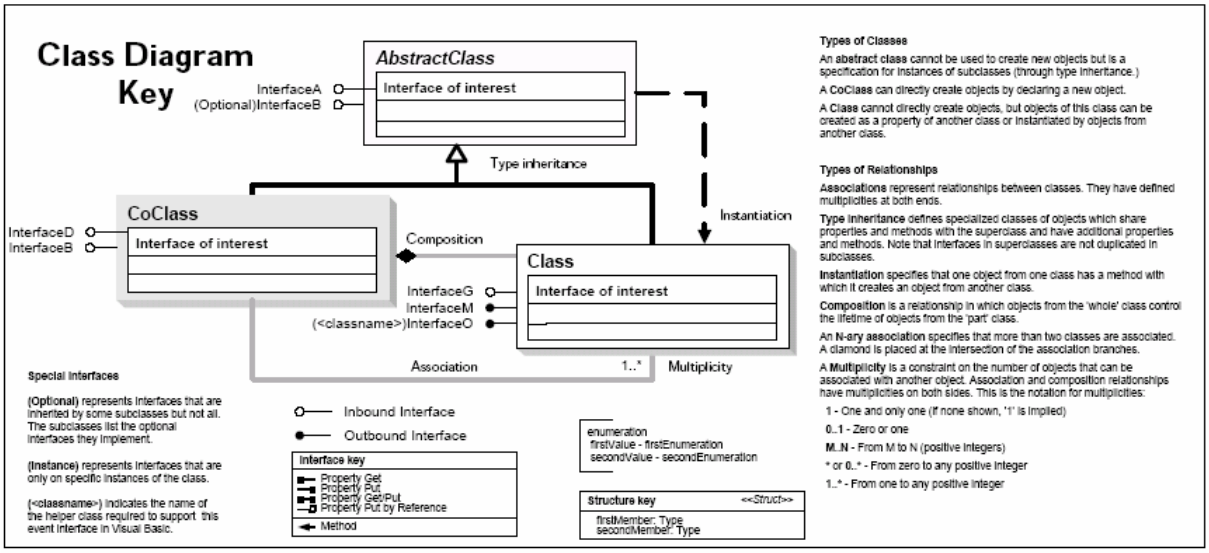


Figure 2 Légende des diagrammes (Classe diagram Key)
La figure 2 représente la légende des OMD.

Les classes

Il y a 3 types de classes :

- **Les Abstract class**, elles sont représentées par un rectangle blanc.

Elles réunissent les propriétés générales à plusieurs sous-classes.
Il n'est pas possible d'instancier des objets à partir de ce type de classe.

- **Les Class**, elles sont représentées par un rectangle 3D.

Il n'est pas possible d'instancier directement des objets à partir de ces class, ils ne peuvent l'être qu'à travers une méthode d'une autre classe ou bien comme propriété d'une autre classe.

- **Les CoClass** (Component Object Class), elles sont représentées par un rectangle 3D gris.

Il est possible d'instancier directement des objets à partir de ces CoClass.

Les relations

Il existe 4 grands types de relations entre les classes :

- **Les relations d'héritages**, elles sont représentées de cette manière 

Une « sous-class » hérite des propriétés et des méthodes de sa super classe.

- **Les relations d'instanciation**, elles sont représentées de cette manière 

Ceci signifie qu'une classe possède une méthode permettant de créer un objet d'une autre classe. (La classe dont on peut créer un objet se situe du coté de la pointe de la flèche).

- **Les relations de composition**, elles sont représentées de cette manière 

Dans ce type de relation les objets de la class parent contrôlent l'existence des objets de la classe enfants. S'ils sont détruits, les autres disparaissent également.

- **Les relations d'association**, elles sont représentées de cette manière

Cela signifie simplement que 2 classes sont associées entre elles.

Les cardinalités

Elles existent pour les relations de composition et d'association uniquement.

- **1** signifie qu'un objet peut être en relation avec un et un seul objet.

- **0..1** signifie qu'un objet peut être en relation avec zéro ou un objet.

- **M..N** signifie que M Objets peuvent être en relation avec N objets.

- *** ou 0..*** signifie qu'un objet peut être en relation avec zéro ou plusieurs objets

- **1..*** signifie qu'un objet peut être en relation avec un ou plusieurs objets

Les propriétés

-  ou  indique une propriété en lecture seule

-  indique une propriété en lecture écriture par valeur

-  indique une propriété en lecture écriture par référence

Les méthodes

- ⟵ indique une méthode.

Les interfaces

- I…  indique une interface.

Il existe un petit utilitaire intitulé EOBrowser.exe qui va permettre d'explorer de façon plus précise qu'avec l'explorateur d'objet VBA et avec plusieurs modes de représentation différents.
Il est situé dans le répertoire c:\… ArcGIS\DeveloperKit\tools

IV Manipulations d'object :

Réaliser un développement dans ArcGis sous visual basic va donc consister à naviguer au sein de cet OMD en passant d'une interface à l'autre afin d'accéder à la classe d'objet recherchée, de l'instancier et enfin de travailler sur l'objet souhaité.

Variables Globales

Dans ArcObject il existe 2 variables qui sont accessibles tout le temps, sans déclaration ni initialisation et qui vont servir de clés d'entrée dans l'OMD.

Application qui pointe sur l'interface Iapplication de la classe Application.
Thisdocument qui pointe sur l'interface Idocument de la Classe Mxdocument. (Document Arcmap)

Le « pointeur » étant placé sur un de ces deux objets, il va s'agir de le déplacer.

Exercice

1. Ouvrer un nouveau document ArcMap.
2 .Renommer le bloc de données.
3. Ouvrer l'éditeur visual basic.
4. Créer un nouveau module standard.
5. Ouvrer sa fenêtre de code.
6. Créer une nouvelle procédure intitulée recupnomcouches
7. Saisir le code suivant dans cette procédure :

```
Dim pMxDOc  as Imxdocument
 Set pMxDOc= thisdocument
Msgbox pMxDoc.focusmap.name
```

8. Basculer dans le document arcMap.
9. Rajouter un bouton à l'interface avec comme code la macro recupnomcouche.
10.Executer la macro.
11.Enregistrer le document arcMap en lui donnant un nom.

REMARQUE :
Dans le code précédent le pointeur est passé de l'interface Imxdocument à l'interface Idocument, toutes les deux appartenant à la classe ImxDocument.
Ceci en plaçant le curseur sur une première interface et ensuite en créant une deuxième variable qui pointe sur la seconde interface vers laquelle on fait basculer le pointeur à l'aide de l'instruction SET
Ceci s'appelle une QUERY INTERFACE

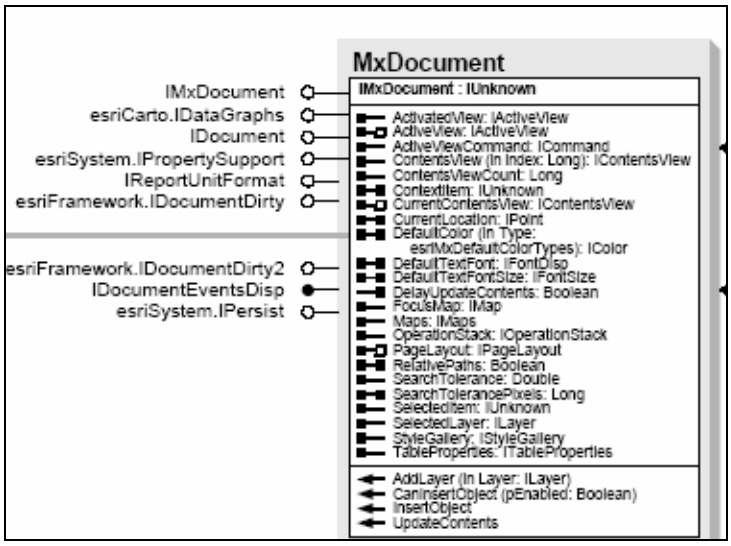


Figure 3 Extrait de l'OMG ArcMapUIModel

Exercice :

1. Dans le même document ArcMap que celui de l'exercice précédent, rajouter des couches au bloc de données.
2. Basculer dans l'éditeur visual basic et créer un nouveau module standard avec une nouvelle procédure intitulés listecouches
3.Taper le code suivant :

```
Dim  pMxDoc as Imxdocument
Dim Pmap as Imap
Dim pEnumlayer as IenumLayer
Dim player as Ilayer
Dim lareponse as string

Set pMxDoc = thisdocument
Set pMap =pMxDoc.FocusMap

Lareponse="La carte est constituée de "  & pmap.LayerCount  & "  couches : "&vbCrlf

Set pEnumLayer =pMap.Layers
Set pLayer = pEnumLayer.Next

Do While Not player is Nothing
        Lareponse = lareponse  & player.name  & vbCrlf
        Set player = pEnumlayer.Next
Loop

MsgBox lareponse
```

4. Basculer dans le document ArcMap.
5. Rajouter un bouton à l'interface avec comme code la macro *listecouches.*
6. Exécuter la macro.
7. Enregistrer le document ArcMap et le fermer.

REMARQUE :

Dans les lignes de codes précédents les changements d'interface se sont produits en utilisant la capacité d'une classe d'implémenter un objet d'une autre classe à l'aide d'une méthode qu'elle possède

Set pMap =pMxDoc.FocusMap

.

Exercice

Dans ce dernier exercice nous allons récapituler tout ce qui a été vu au tout au long de cet atelier. Il va s'agir de réaliser un formulaire qui va afficher la liste des cartes présentes dans un document ArcMap. Ce même formulaire, après sélection d'une de ces cartes affichera la liste des couches qu'elle possède et enfin dans une 3ème liste s'affichera la liste des champs de la table de la couche sélectionnée.

1. Ouvrer le document ArcMap, intitulé Formulaire situé sous le répertoire C:\Atelier_VB\Documents Arcmap.

2. Lancer l'éditeur Visual Basic.

3. Créer un module de feuille et le nommer listing.

4. Créer 3 contrôles de type listbox, que vous nommer respectivement lstmaps, lstlayers et lstfields. (cf. figure 4)

5. Créer 1 contrôle de type bouton de commande, que vous nommer fermer.

6. Renseigner la propriété « Caption » de ce bouton de commande par « Quitter » et par formulaire celle du formulaire.

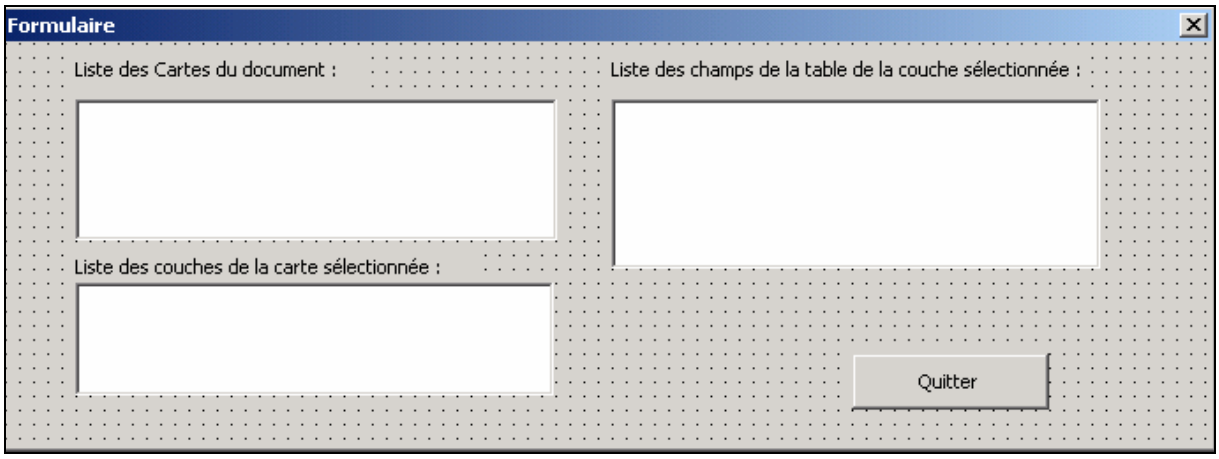


Figure 4 : le formulaire listing

7.Accéder à la fenêtre de code de ce formulaire et naviguer jusqu'à la section ***General Declaration*** *de la feuille.*

8. Saisir le code suivant :

```
Dim pMxDoc As IMxDocument
Dim pMaps As IMaps
Dim Pmap As IMap
Dim pLayers As IEnumLayer
Dim player As ILayer
Dim Txtmap As Variant
Dim TxtLayer As Variant
Dim TxtField As Variant
```

Attention à la syntaxe des variables et à l'alternance des majuscules et minuscules qui n'est pas tant pour faire joli que pour vous faciliter la tâche en évitant les fautes de frappe. En effet VB va les reconnaître dans les différentes procédures ou vous aller saisir du code et reproduire leur syntaxe telle qu'elle a été définie ici. Si ce n'est pas le cas cela signifie qu'il ne reconnaît pas la variable et donc qu'il y a vraisemblablement une faute de frappe.

Il s'agit de déclarer toutes les variables utilisées par ce formulaire.

9. Naviguer jusqu'à la procédure événementielle correspondant à ***l'événement Initialize*** *du formulaire et y saisir le code suivant :*

```
Dim i As Integer
Dim intMapcount As Integer
Set pMxDoc = ThisDocument
```

```
Set pMaps = pMxDoc.Maps
 i = 0
 intMapcount = pMaps.Count

For i = 0 To intMapcount - 1
        lstmaps.AddItem pMaps.Item(i).Name
 Next i
```

Ceci permet au formulaire à son ouverture, quand il s'initialise, de récupérer les cartes présentes dans le document ArcMap.

10. Naviguer jusqu'à la procédure ***événementielle Click*** *du contrôle Fermer et y inscrire le code suivant :*

```
Unload Me
```

11.Tester votre code en exécutant le formulaire

Normalement le formulaire doit s'ouvrir et afficher la liste des cartes présentes dans le document.

12. Cliquer sur le bouton Quitter.

13. Naviguer jusqu'à la procédure ***événementielle Click*** *du contrôle Listmaps et saisir le code suivant :*

```
Txtmap = ""
Txtmap = lstmaps.Value
Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
Dim pMaps As IMaps
Dim i As Integer
Set pMaps = pMxDoc.Maps
i = 0
Dim c As Integer
c = pMaps.Count
Dim Pmap As IMap
lstlayers.Clear
Lstfields.Clear
TxtLayer = ""
TxtField = ""

For i = 0 To c - 1

   If pMaps.Item(i).Name = Txtmap Then
        Dim pLayers As IEnumLayer
        Set pLayers = pMaps.Item(i).Layers
        Dim player As ILayer
        Set player = pLayers.Next

      Do Until player Is Nothing
        lstlayers.AddItem player.Name
        Set player = pLayers.Next
      Loop
   End If

Next i
```

Il s'agit de déterminer la valeur sélectionnée dans la liste des cartes pour ensuite récupérer la liste des couches qu'elle possède et les afficher dans ce contrôle.
Tester votre code en exécutant le formulaire à nouveau. Dans la liste des cartes qui s'affiche en sélectionner une ,Les couches qui la constituent doivent apparaitrent …

14. Naviguer jusqu'à la procédure évenementielle Click du contrôle listlayer et y saisir le code suivant :

```
TxtLayer = lstlayers.Value

Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
Dim pMaps As IMaps
Set pMaps = pMxDoc.Maps
Dim i As Integer
i = 0
Dim c As Integer
c = pMaps.Count
Dim Pmap As IMap

For i = 0 To c - 1
   If pMaps.Item(i).Name = Txtmap Then
      Dim pLayers As IEnumLayer
      Set pLayers = pMaps.Item(i).Layers
      Dim player As ILayer
      Set player = pLayers.Next

         Do Until player Is Nothing

               If player.Name = TxtLayer Then
                     Dim pfields As IFields
                     Dim pLayerclass As IClass
                     Set pLayerclass = player
                     Set pfields = pLayerclass.Fields

                     Dim fieldcount As Integer
                     fieldcount = pfields.fieldcount
                     Dim x As Integer
                     x = 0
                     Lstfields.Clear
                     TxtField = ""

                     For x = 0 To fieldcount - 1
                     Lstfields.AddItem pfields.Field(x).Name
                     Next x
                  End If
            Set player = pLayers.Next
         Loop
   End If
Next i
```

Il ne vous reste plus qu'à tester une dernière fois votre code en relançant le formulaire et faire afficher la liste des champs dans le 3ème listing.

Vous aller maintenant créer une procédure pour réaliser l'ouverture de votre formulaire à partir de votre document ArcMap et non depuis l'éditeur visual basic. Cette nouvelle procédure va être rédigée dans un module standard du projet que vous allez créer et ensuite vous l'associerez à un contrôle sur l'interface du document ArcMap.

15. Dans le projet, créer un module standard et le nommer « standard ».
16. Créer une procédure appeler ouvertureformulaire
17. Saisir le code suivant dans cette nouvelle procédure :

```
Load Listing
Listing.Show
```

18. Basculer dans Arcmap et rajouter une commande à l'interface d'Arcmap avec cette procédure associée. Tester votre nouvelle commande.

19. Retourner dans l'éditeur visual basic et à partir de la fenêtre de ^projet exporter les modules « standard » et « listing » dans le répertoire C:\Atelier_Vb\export.

20. Ouvrer un nouveau document ArcMap. Ouvrer l'éditeur visual basic et importer les 2 fichiers listing.frm et standard.bas que vous venez de créer. Exécuter le formulaire listing.