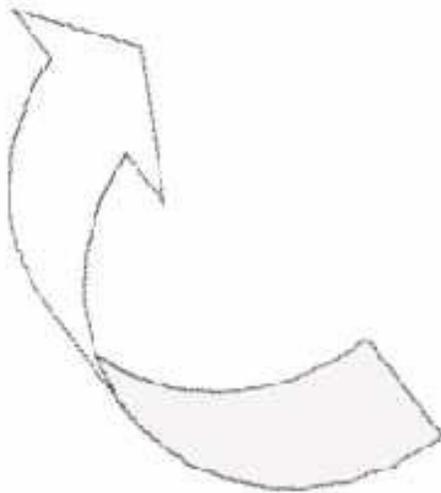
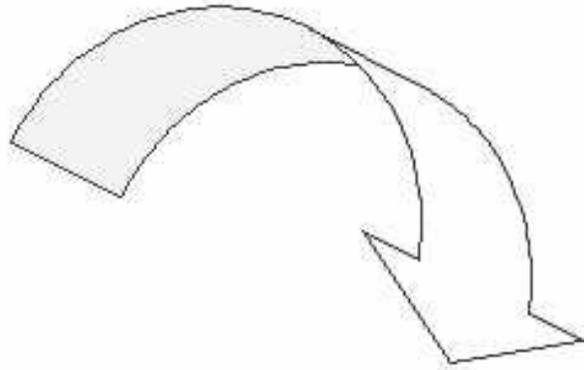


**C.I.S.A.**  
**Conferenza Italiana per Sviluppatori**  
Arezzo 4/5 giugno 2005

**Automazione Excel da Access**



([http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/vccore/html/\\_core\\_ole\\_ba ckground.asp](http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/vccore/html/_core_ole_ba ckground.asp))

## OLE - Object Linking and Embedding

### Introduzione a OLE

OLE è un meccanismo che permette agli utenti di creare e modificare documenti contenenti elementi o "oggetti" creati da più applicazioni.

Inizialmente OLE era l'acronimo di Object Linking and Embedding. Ora invece l'estensione dell'acronimo non viene più utilizzata e si indica semplicemente OLE. Le funzionalità di OLE che non riguardano il collegamento e l'incorporamento sono incluse nella tecnologia Active.

I documenti OLE, denominati precedentemente documenti composti, integrano vari tipi di dati o componenti. Clip audio, fogli di calcolo e bitmap sono esempi tipici di componenti presenti nei documenti OLE. L'inclusione del supporto OLE nell'applicazione consente agli utenti di utilizzare i documenti OLE senza la necessità di attivare le diverse applicazioni. Tale passaggio, infatti, viene eseguito automaticamente.

Per creare documenti composti viene utilizzata un'applicazione contenitore. Per creare invece gli elementi all'interno del documento contenitore viene creata un'applicazione server o un'applicazione componente. Le applicazioni scritte possono essere un contenitore, un server o entrambi.

OLE racchiude numerosi concetti diversi, tutti accomunati dall'obiettivo di ottenere un'interazione uniforme tra le applicazioni. I più importanti sono elencati di seguito:

#### Collegamento e incorporamento

Il collegamento e l'incorporamento sono i due metodi utilizzati per l'archiviazione di elementi creati all'interno di un documento OLE, a sua volta creato in un'altra applicazione. L'interfaccia dell'applicazione contenitore si modifica per includere le funzionalità dell'applicazione componente con la quale è stato creato l'elemento incorporato. Gli elementi collegati non sono mai attivati sul posto in quanto i dati reali dell'elemento sono contenuti in un file separato, al di fuori del contesto dell'applicazione contenente il collegamento.

**Nota** *Il collegamento, l'incorporamento e l'attivazione sul posto rappresentano le funzionalità principali della modifica visiva OLE.*

#### Automazione

L'automazione rende possibile il controllo di un'applicazione da parte di un'altra applicazione. L'applicazione con il controllo viene definita client di automazione, mentre l'applicazione controllata viene definita server o componente di automazione.

**Nota** *L'automazione funziona sia in ambito OLE che nei contesti della tecnologia Active. È possibile automatizzare qualsiasi oggetto basato su COM.*

#### File composti

I file composti costituiscono un formato di file standard che semplifica l'archiviazione strutturata di documenti composti per le applicazioni OLE. All'interno di un file composito, le archiviazioni presentano molte funzionalità delle directory e i flussi presentano molte funzionalità dei file. Questa tecnologia viene anche denominata archiviazione strutturata.

#### Trasferimento dati uniforme

Il trasferimento dati uniforme (UDT, Uniform Data Transfer) è un gruppo di interfacce che consentono l'invio e la ricezione dei dati in una modalità standard, indipendentemente dal metodo effettivamente scelto per il trasferimento dei dati. UDT rappresenta la base dei trasferimenti dati tramite il trascinamento degli elementi

selezionati e viene utilizzato ora anche per il trasferimento di dati Windows esistenti, quali gli Appunti e lo scambio dinamico di dati (DDE, dynamic data exchange).

#### Trascinamento della selezione

Il trascinamento della selezione è una tecnica semplice e a modifica diretta per il trasferimento di dati tra applicazioni, tra finestre all'interno di un'applicazione o anche all'interno di una sola finestra in un'applicazione. I dati da trasferire vengono selezionati e trascinati nella destinazione desiderata. Il trascinamento della selezione si basa sul trasferimento dati uniforme.

#### Component Object Model

Il modello COM (Component Object Model) costituisce l'infrastruttura utilizzata nelle comunicazioni tra oggetti OLE. Le classi OLE MFC rappresentano una semplificazione del modello COM per il programmatore. COM fa parte della tecnologia Active in quanto gli oggetti COM sono alla base sia di OLE che della tecnologia Active.

(<http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/modcore/html/deconunderstandingofficeobjectsubjectmodels.asp>)

## Il modello COM (Component Object Model)

### Oggetti, insiemi e modelli di oggetti: elementi fondamentali delle tecnologie

Le applicazioni di Microsoft® Office XP espongono le proprie funzionalità al linguaggio VBA (Microsoft® Visual Basic®, Applications Edition) tramite un sistema gerarchico di oggetti e insiemi di oggetti denominato modello di oggetti. Dopo aver compreso come creare un riferimento agli oggetti in un modello di oggetti di un'applicazione, è possibile utilizzare gli oggetti e le funzionalità disponibili per creare la propria soluzione.

Un'applicazione è fondamentalmente composta da due elementi, ovvero il contenuto e le funzionalità. Il contenuto si riferisce alle informazioni presenti in un'applicazione, ovvero documenti, fogli di lavoro, tabelle o diapositive e i relativi dati. Il contenuto fa riferimento alle informazioni sugli attributi dei singoli elementi di quell'applicazione, ad esempio la dimensione di una finestra, il colore di un elemento grafico o le dimensioni del carattere di una parola. Le funzionalità sono tutte le modalità di gestione e utilizzo del contenuto dell'applicazione, ad esempio la possibilità di aprire, chiudere, aggiungere, eliminare, inviare, copiare, incollare, modificare o formattare il contenuto dell'applicazione.

Il contenuto e le funzionalità che costituiscono un'applicazione sono presentati al linguaggio Visual Basic come unità discrete denominate oggetti.

Gli oggetti esposti da un'applicazione vengono ordinati in base a relazioni gerarchiche. L'oggetto di livello principale in un'applicazione di Microsoft® Office XP è l'oggetto Application, ovvero l'applicazione stessa. L'oggetto Application include a sua volta altri oggetti ai quali è possibile accedere solo se esiste l'oggetto Application, ovvero se è in esecuzione un'istanza dell'applicazione stessa. Ad esempio, l'oggetto Application di Excel include oggetti Workbook, mentre l'oggetto Application di Word contiene gli oggetti Document. L'oggetto Document può esistere solo se esiste l'oggetto Application di Word, pertanto viene definito figlio dell'oggetto Application. Viceversa, l'oggetto Application è definito padre dell'oggetto Document.

Molti oggetti figlio possono includere, a loro volta, altri figli.

Oltre a contenere gli oggetti figlio, ogni oggetto della gerarchia include contenuti e funzionalità correlati sia all'oggetto stesso che a tutti gli oggetti figlio. Più alta è la posizione occupata da un oggetto in una gerarchia di oggetti nidificati, ovvero maggiore è il numero di oggetti figlio inclusi in un oggetto, più vasto sarà l'ambito del contenuto e delle funzionalità. Ad esempio, in Excel l'oggetto Application include la dimensione della finestra dell'applicazione e la possibilità di uscire dall'applicazione, l'oggetto Workbook include il nome del file, il formato della cartella di lavoro e la possibilità di salvare tale cartella, mentre l'insieme Worksheets contiene i nomi dell'oggetto Worksheet e la possibilità di aggiungere ed eliminare i fogli di lavoro.

Per gestire il contenuto e le funzionalità esposti da un oggetto, si utilizzano le proprietà e i metodi di tale oggetto. Le proprietà consentono di determinare o modificare le caratteristiche di un oggetto, quali il colore, le dimensioni o lo stato. Ad esempio, è possibile impostare la proprietà Visible di un oggetto Worksheet di Excel per specificare se rendere visibile il foglio di lavoro all'utente. I metodi, invece, consentono di eseguire azioni particolari su un oggetto. Ad esempio, il metodo PrintOut dell'oggetto Document di Word consente di stampare il documento.

**Nota** Per utilizzare al meglio l'automazione nelle soluzioni per Office, è necessario conoscere a fondo le applicazioni che si desidera integrare. Questo tipo di conoscenza ed esperienza può essere acquisita solo tramite esercitazioni sulle applicazioni specifiche e con l'esperienza pratica.

(<http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/modcore/html/deconsettinreferencesworkingwithobjectvariables.asp>)

### **Impostazione di riferimenti**

Per automatizzare un'applicazione di Microsoft® Office da un'altra applicazione, è necessario innanzitutto fare riferimento all'applicazione che si desidera automatizzare. Questo riferimento consente all'applicazione di individuare gli oggetti esposti dall'altra applicazione. In genere, questa operazione consiste nell'impostare un riferimento alla libreria dei tipi dell'applicazione utilizzando la finestra di dialogo Riferimenti.

Prima di utilizzare gli oggetti esposti da un'applicazione di Office, è opportuno impostare un riferimento a quell'applicazione utilizzando la finestra di dialogo Riferimenti.

L'aggiunta di riferimenti non necessari causa l'aumento del tempo necessario per il caricamento della soluzione e consuma ulteriori risorse della memoria.

Per utilizzare gli oggetti di un'altra applicazione di Office (o gli oggetti esposti da un'altra applicazione o componente che supporta l'automazione) senza impostare un riferimento nella finestra di dialogo Riferimenti, utilizzare la funzione CreateObject o GetObject e dichiarare le variabili oggetto come tipo Object generico.

Quando si utilizza questa procedura, gli oggetti del codice vengono associati in modo tardivo. Ne consegue che non sarà possibile utilizzare gli strumenti in fase di progettazione, ad esempio il completamento automatico dell'istruzione o Visualizzatore di oggetti, e il codice non verrà eseguito con la stessa velocità.

(<http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/modcore/html/decondeclaringobjectvariables.asp>)

## Dichiarazione di una variabile oggetto

Affinché un'applicazione possa utilizzare gli oggetti esposti dalla libreria dei tipi di un'altra applicazione, è necessario innanzitutto stabilire il tipo di informazioni contenute in quella libreria dei tipi. Il processo di esecuzione di query sugli oggetti, i metodi e le proprietà esposti da un'altra applicazione viene denominato associazione. La programmazione VBA (Microsoft® Visual Basic®, Applications Edition) nelle applicazioni di Microsoft® Office supporta due tipi di associazione, ovvero l'associazione anticipata e l'associazione tardiva. I tempi e le modalità relativi alle associazioni possono influenzare pesantemente le prestazioni di una soluzione.

### Dichiarazioni di associazioni anticipate

Le associazioni anticipate consentono di dichiarare una variabile oggetto come identificatore programmatico o nome di classe, invece che come tipo di dati Object o Variant. L'identificatore programmatico di un'applicazione viene archiviato nel Registro di sistema di Microsoft® Windows® come sottochiave nella sottostruttura \HKEY\_CLASSES\_ROOT. Ad esempio, l'identificatore programmatico per Microsoft® Access è rappresentato da "Access.Application", quello per Microsoft® Excel è "Excel.Application".

Quando si utilizza un'associazione anticipata, per inizializzare una variabile oggetto è possibile utilizzare la funzione CreateObject o GetObject oppure la parola chiave New, se supportata dall'applicazione. Tutte le applicazioni di Office XP possono essere inizializzate tramite la parola chiave New. Poiché l'ambiente di programmazione di Microsoft® Outlook® per gli elementi di Outlook supporta solo script, non è possibile utilizzare le dichiarazioni di associazioni anticipate nel relativo ambiente di programmazione VBScript. Tuttavia, è possibile utilizzare l'associazione anticipata nel codice VBA di un progetto VBA locale di Outlook o di un componente aggiuntivo COM oppure nel codice di automazione collegato a Outlook da un'altra applicazione host.

Nel frammento di codice seguente è inclusa la dichiarazione di una variabile Application utilizzando l'identificatore programmatico per Word (Word.Application), quindi la creazione di una nuova istanza di Word utilizzando l'istruzione Set con la parola chiave New:

```
Dim wdApp As Word.Application  
Set wdApp = New Word.Application
```

Se nel codice che segue queste righe non è inclusa l'impostazione su True della proprietà Visible dell'oggetto Application, la nuova istanza di Word risulterà nascosta. Tutte le applicazioni di Office sono nascoste per impostazione predefinita quando automatizzate da un'altra applicazione.

È consigliabile utilizzare l'associazione anticipata quando possibile, in quanto essa presenta i vantaggi seguenti:

**Controllo sintassi** - Quando si utilizza l'associazione anticipata, VBA controlla la sintassi delle istruzioni in relazione alla sintassi archiviata nella libreria degli oggetti durante la compilazione e non in fase di esecuzione. Pertanto, è possibile rilevare ed eliminare gli errori in fase di progettazione. Ad esempio, VBA è in grado di stabilire se le proprietà e i metodi utilizzati per un oggetto sono validi, così come gli argomenti passati a tali proprietà e metodi.

**Supporto di strumenti per la creazione di istruzioni** - Quando si utilizza l'associazione anticipata, Visual Basic Editor supporta alcune funzionalità che semplificano la scrittura del codice riducendo il numero di possibili errori, ad esempio l'elencazione automatica delle proprietà e dei metodi di un oggetto e i suggerimenti popup per gli argomenti predefiniti.

**Supporto per costanti incorporate** - Quando si utilizza l'associazione anticipata, il codice fa riferimento alle costanti incorporate per gli argomenti dei metodi e le impostazioni delle proprietà perché queste informazioni sono disponibili dalla libreria dei tipi in fase di progettazione. Se invece si utilizza l'associazione tardiva, è necessario definire queste costanti nel codice ricercando i valori nella documentazione dell'applicazione.

**Migliori prestazioni** - L'associazione anticipata consente di ottenere prestazioni significativamente più veloci rispetto all'associazione tardiva.

#### Dichiarazioni di associazioni tardive

L'associazione tardiva consente di dichiarare una variabile come un tipo di dati Object o Variant. Per inizializzare la variabile, richiamare la funzione GetObject o CreateObject e specificare l'identificatore programmatico dell'applicazione. Ad esempio, nel frammento di codice seguente viene innanzitutto dichiarata una variabile Object che viene poi impostata su un'istanza di Microsoft® Access tramite la funzione CreateObject:

```
Dim objApp As Object  
Set objApp = CreateObject("Access.Application")
```

L'espressione associazione tardiva è il nome descrittivo dell'associazione IDispatch del linguaggio C, ovvero il primo metodo di associazione implementato nelle applicazioni che possono controllare le altre applicazioni tramite l'automazione. Per questo motivo, è possibile utilizzare l'associazione tardiva per mantenere la compatibilità con le versioni precedenti delle applicazioni. L'associazione tardiva comporta, tuttavia, un considerevole overhead. È più veloce dello scambio dinamico dei dati (DDE), ma più lenta dell'associazione anticipata.

**Nota** *In alcune applicazioni e alcuni componenti che supportano l'automazione è possibile utilizzare solo l'associazione tardiva. Tutte le applicazioni di Office XP e quasi tutte le applicazioni recenti che supportano l'automazione supportano anche le associazioni anticipata e tardiva. Tuttavia, i linguaggi script, quali VBScript e Microsoft® JScript®, non supportano l'associazione anticipata, poiché non supportano i riferimenti né tipi di dati oggetto specifici. Ad esempio, VBScript supporta solo il tipo di dati Variant.*

(<http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/modcore/html/deconshuttimgdownobjectcreatedbyusingautomation.asp>)

### Chiusura di oggetti creati tramite automazione

Generalmente una variabile locale viene eliminata al termine dell'esecuzione della routine nella quale è stata dichiarata. Tuttavia, è consigliabile eliminare esplicitamente le variabili oggetto a livello di applicazione utilizzate per automatizzare un'altra applicazione, impostandole con la parola chiave Nothing. In questo modo, viene liberata tutta la memoria utilizzata dalla variabile. Per alcuni oggetti Application potrebbe essere necessario utilizzare il metodo Quit dell'oggetto per eliminare completamente una variabile oggetto e liberare la memoria utilizzata. In genere, è più sicuro eseguire entrambe le operazioni, ovvero utilizzare il metodo Quit e quindi impostare la variabile oggetto con la parola chiave Nothing.

```
Dim wdApp As Word.Application  
Set wdApp = New Word.Application  
...  
wdApp.Quit  
Set wdApp = Nothing
```

([http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/dv\\_wrcore/html/wrconexclobjectmodeloverview.asp](http://msdn.microsoft.com/library/ita/default.asp?url=/library/ITA/dv_wrcore/html/wrconexclobjectmodeloverview.asp))

## Cenni preliminari sul modello a oggetti di Excel

Per sviluppare soluzioni che utilizzano Microsoft Office Excel 2003, è necessario interagire con gli oggetti forniti dal modello a oggetti di Excel. Principalmente, il modello a oggetti emula direttamente l'interfaccia utente. L'oggetto Application fornisce ad esempio un wrapper per l'intera applicazione e ciascun oggetto Workbook contiene un insieme di oggetti Worksheet. Al livello immediatamente inferiore, la principale astrazione utilizzabile per rappresentare le celle è costituita dall'oggetto Range, che consente di utilizzare singole celle o gruppi di celle.

Sebbene in Excel siano disponibili centinaia di oggetti con cui è possibile interagire, per comprendere il modello a oggetti è sufficiente concentrarsi su una piccola parte degli oggetti disponibili. Tali oggetti comprendono:

- Oggetto Application
- Oggetto Workbook
- Oggetto Worksheet
- Oggetto Range

La maggior parte delle operazioni effettuate in Excel coinvolge queste quattro classi e i relativi membri.

### Oggetto Application

L'oggetto Application di Excel rappresenta l'applicazione Excel stessa. Ed espone numerose informazioni sull'applicazione in esecuzione, le opzioni applicate a tale istanza e gli oggetti dell'utente corrente aperti all'interno dell'istanza.

### Oggetto Workbook

La classe Workbook rappresenta una singola cartella di lavoro all'interno dell'applicazione Excel. Molti membri della classe Application possono essere rappresentati anche come membri della classe Workbook. In questo caso, le proprietà impostate vengono applicate a una cartella di lavoro specifica e non alla cartella di lavoro attiva.

### Oggetto Worksheet

Sebbene la classe Worksheet comprenda numerosi membri, quasi tutti i metodi, le proprietà e gli eventi sono identici o simili a quelli delle classi Application e/o Workbook.

Excel fornisce l'insieme *Sheets* come proprietà dell'oggetto Workbook, ma in Excel non è presente alcuna classe Sheet. I singoli membri dell'insieme Sheets sono invece oggetti Worksheet o Chart.

### Oggetto Range

L'oggetto Range è il più utilizzato nelle applicazioni basate su Excel. Per poter modificare un'area all'interno di Excel, è necessario rappresentarla come oggetto Range e utilizzare i metodi e le proprietà di tale oggetto. Un oggetto Range può rappresentare una cella, una riga, una colonna, una selezione di celle contenente uno o più blocchi di celle, contigue o non contigue, o anche un gruppo costituito da celle situate in più fogli.

Esistono molte altre classi utili, quali *PivotTable* e *Chart*. Grazie al modello a oggetti è possibile effettuare praticamente tutte le attività automatizzate necessarie.