

## A 3D sphere with various programming and web technologies written on it, with 'code' prominently displayed in the center. The sphere is white with black text for most terms and blue text for 'code'. The terms include: php, cms, sql, rss, web, cvs, sem, ssl, xhtml, css, java, seo, html, and www.

*Il futuro dell'ambiente di sviluppo Microsoft è molto incerto. Meglio prepararsi alla transizione?*



**N**on ci siamo fatti prendere dal millenarismo, tantomeno adesso che il primo millennio è passato senza danni e il secondo anche. Ci siamo, invece, posti il problema di cosa fare con una base di codice Visual Basic 6 ereditata dai tempi d'oro di questo linguaggio, tempi purtroppo trascorsi da un pezzo. Visual Basic 6, infatti è decisamente fuori tempo massimo. Non solo, ma mantenere in vita certi progetti può rendere difficoltoso l'acquisto di nuovi computer, dato che può essere necessario fare il downgrade del sistema operativo, almeno sulle macchine di sviluppo. Microsoft continua a offrire supporto per il suo ex cavallo di battaglia, forse fino alla prossima release del sistema operativo, ma è un fatto che da più di qualche anno sta tentando di scoraggiare la sopravvivenza del suo ex pupillo, ed è una buona cosa non farsi cogliere impreparati.

L'azienda di Redmond ha avuto ottime ragioni per passare dalla vecchia architettura a quella corrente. Riassumiamole brevemente. In primo luogo,

l'architettura del Com è basata sull'esportazione di proprietà e funzioni a livello binario. Chi ha pensato a questa soluzione ha creato un ottimo modello per il link dinamico di codice C, ma non ha sì è reso conto per tempo di aver generato falle nella sicurezza, che lasciano un sacco di opzioni ai

creatori di malware. In secondo luogo, l'architettura di Com è troppo povera per supportare in modo pulito versioni diverse del software, dando origine al famoso Dll hell, ossia al problema di trovare un insieme di librerie dinamiche che non mettesse in crisi nessuna delle applicazioni installate, man mano che ci sono aggiornamenti e iniziano a convivere applicazioni realizzate in tempi diversi.

La nuova architettura, .net, si fonda su una macchina virtuale proprio come Java. In parte ci sono motivazioni storiche, dato che l'autore dell'architettura di .net ha iniziato la sua carriera in Microsoft con VisualJ++, dopo una lunga esperienza in Borland con Delphi. VisualJ++ fu la prova che si può realizzare un ambiente virtualizzando l'intera piattaforma, senza dare accesso diretto alle librerie dinamiche Dll. La macchina virtuale può eseguire controlli di congruenza e applicare regole di sicurezza al codice binario, cosa che è fuori della portata del codice compilato.

Come aveva dimostrato ampiamente Java, si può realizzare una piattaforma molto più sicura usando una macchina virtuale, senza una evidente penalizzazione in termini di performance. In secondo luogo, la macchina virtuale comporta l'indipendenza

Il diagramma illustra la strategia di migrazione basata sulla qualità dell'applicazione (X-asse) e sulla strategia (Y-asse).

Alta	<b>RISCRIVERE</b>	<b>MIGRARE</b>
Bassa	<b>SOSTITUIRE</b>	<b>RIUTILIZZARE</b>
	Bassa	Alta

Qualità dell'applicazione

158 PC Professionale &gt; Novembre 2014

dalla piattaforma hardware e software. Quindi, anche se tutti vogliono bene a Intel, la grande disponibilità di chip potenti e veloci e la maggiore varietà di piattaforme consentono l'apertura di mercati nuovi, che è una buona cosa. Microsoft si è aperta la strada verso gli Arm che muovono i telefoni e i tablet in circolazione, proprio grazie alla disponibilità di una piattaforma su una macchina virtuale.

Un gruppo di developer particolarmente dinamico ha creato un clone della piattaforma su Linux, il progetto Mono, e adesso disponiamo di tool di sviluppo open source, come Mono-Develop e ambienti compatibili con .net su Android e iOS, come Xamarin, che sono un'opportunità in più per gli sviluppatori Windows.

Microsoft ha visto tutto questo e ha imboccato la strada della piattaforma .net lasciandosi alle spalle il Com e le Dll, con tutti i problemi di compatibilità e sicurezza che hanno afflitto Redmond per un decennio.

Sulla nuova piattaforma, Microsoft ha scelto di creare un nuovo linguaggio, più semplice del C++, come Java, e più completo e flessibile di Visual Basic, mentre il linguaggio più immediato, si è adattato alla piattaforma, assumendo molti dei meccanismi di C# e diventando Visual basic.net.

La semantica più evoluta si è riflessa in un linguaggio maggiormente complesso, diventato ulteriormente verboso rispetto alla versione precedente. D'altra

parte, Visual Basic non è mai stato un linguaggio in grado di creare una piattaforma: chi ha provato a strutturare un'applicazione con qualche centinaio di oggetti, se ne è pentito amaramente. Il risultato è stato interessante: molte nuove possibilità per gli sviluppatori più evoluti e un piccolo ghetto per chi non ha seguito per tempo la direzione del vento. Per qualche ragione Microsoft non ha voluto creare un linguaggio limitato, con meno opzioni, adatto per lo scripting di controlli scritti da altri, ma ha tentato di rendere Visual Basic un linguaggio sullo stesso piano di C# realizzando alla fine qualcosa che non è né più semplice né più coerente del linguaggio di riferimento.

## STRATEGIE EVOLUTIVE

Eric Nelson, che ha un blog su Msdn dedicato a Visual Basic dal significativo titolo di GOTO 100, suggeriva nel 2008 un modo di affrontare il problema del futuro di un'applicazione legacy, adottando per la valutazione lo schema della figura 1: il valore di business (la specificità del problema che risolve) e la qualità del prodotto. Si crea una matrice con quattro zone: nella zona vicina all'origine, quella in cui situiamo applicazioni per uso generico di bassa qualità, l'opzione migliore è sostituire l'applicazione, trovare un altro prodotto che fa lo stesso lavoro e non ha piombo sulle ali. Se l'applicazione è generica, esiste sicuramente

un ventaglio di offerte equivalenti e il cambiamento non sarà pesante.

Se ci spostiamo verso destra sull'asse della qualità, entriamo nella zona in cui conviene continuare a usare l'applicazione com'è. Se la qualità è soddisfacente, possiamo fidarci della continuità del supporto per le applicazioni legacy da parte di Microsoft e non investire fino a quando non sarà strettamente necessario.

Se invece un'applicazione è fortemente specifica, quindi ha un valore di business elevato, e la qualità è bassa, vale la pena di considerare di riscrivere l'applicativo. Prima o poi toccherà

## UN ESEMPIO DI AMBIGUITÀ

```
Function MyFunction( var1, var2 )
    MyFunction = var1 + var2
End Function
```

Per quanto semplice pone grandi problemi. Non sappiamo infatti se esegua una concatenazione di stringhe, un'addizione di numeri o il calcolo di una data. Tutti questi sono usi legittimi della funzione, come mostra il codice seguente

```
Debug.Print MyFunction("ciao ", " a tutti")
Debug.Print MyFunction(3, 2)
Debug.Print MyFunction(Now(), 1)
```

```
Function MyFunction( var1 As String, var2
As String ) As String
    MyFunction = var1 & var2
End Function
```

```
Function MyFunction( var1 As Integer,
var2 As Integer ) As Integer
    MyFunction = var1 + var2
End Function
```

```
Private Function myFunction(theDate As
System.DateTime, offset As Double)
    myFunction = theDate.AddDays(offset)
End Function
```

Eseguendo il codice abbiamo nella finestra di output

```
ciao a tutti
5
01/10/2014 19:08:48
```

Visual Studio

HOME PAGE ESEMPI LINGUAGGI ESTENSIONI DOCUMENTAZIONE FORUM

Free Book - Upgrading Microsoft Visual Basic 6.0 to Microsoft Visual Basic .NET

### Free Book - Upgrading Microsoft Visual Basic 6.0 to Microsoft Visual Basic .NET

Upgrading Microsoft Visual Basic 6.0 to Microsoft Visual Basic .NET is the complete technical guide to upgrading Visual Basic 6 applications to Visual Basic .NET, covering all upgrade topics from APIs to ZOrders. It shows how to fix upgrade issues with forms, language, data access, and COM+ Services, and how to upgrade applications with XML Web services, ADO.NET, and .NET remoting. It also provides big-picture architectural advice, a reference of function and object model changes, hundreds of before-and-after code samples, and a CD packed with useful examples.

C'è un libro elettronico sul sito Microsoft, che affronta in dettaglio i problemi di porting delle applicazioni Visual Basic ([bit.ly/vb6upgradebook](http://bit.ly/vb6upgradebook))

comunque riscriverlo e non conviene perdere tempo, inoltre, potendo usare l'applicativo esistente come una specifica corretta delle necessità di business, avremo un progetto ideale, quello in cui le uniche difficoltà sono quelle tecniche. Naturalmente, ci sarà da decidere in quale direzione muoversi nelle scelte architetturelle.

Infine, quando la qualità è buona e il valore di business alto, può convenire, secondo Nelson, sperimentare con una migrazione dell'applicazione sulla nuova piattaforma. Se la qualità del codice è alta, incontreremo meno trappole sulla strada, ma la domanda è se e come si può considerare un processo di traduzione. Analizziamo la situazione caso per caso.

## CASO 1 NON FARE NULLA



Lasciare il mondo come sta non è sempre sinonimo di ignavia o vigliaccheria. A volte può trattarsi di saggezza e prudenza, soprattutto quando si tratta di scelte che comportano investimenti. Microsoft supporterà Visual Basic per tutta la vita di Windows 8 (bit.ly/vb6support e bit.ly/vb6official) e ha rilasciato un aggiornamento cumulativo per l'ambiente run time di Visual Basic (support.microsoft.com/kb/957924). Con queste premesse, le applicazioni Visual Basic sono al sicuro per qualche anno, anche se Windows 8 sarà due numeri di release indietro fra breve. Abbastanza per consentire agli sviluppatori più stagionati di andare in pensione, come qualcuno notava sul web.

L'ambiente di sviluppo non è supportato dal 2008, ma non è obbligatorio fare il downgrade delle macchine a Windows XP, si può benissimo usare Windows 7 o 8, basta seguire i consigli riportati in un articolo apparso sul sito per sviluppatori Microsoft, (bit.ly/vb6onwin8). Si noti che Msdn ha un pulsante per tradurre gli articoli in Italiano. Più o meno le stesse istruzioni si trovano su vb6andwindows8.

## UNA FORM VB.NET

Questo è il codice della form principale di un semplice visualizzatore di immagini con una finestra di dialogo per caricare una foto e una checkbox per cambiare la modalità di visualizzazione. Basta un'occhiata per trovare qualcosa di familiare e qualcosa di nuovo, soprattutto nella dichiarazione delle funzioni e nelle liste di parametri.

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    ' No code needed here for this sample.
End Sub

Private Sub checkBox1_CheckedChanged(sender As Object, e As EventArgs) Handles checkBox1.CheckedChanged
    ' If the user selects the Stretch check box,
    ' change the PictureBox's SizeMode property to "Stretch".
    ' If the user clears the check box, change it to "Normal".
    ' Choosing Stretch shows the entire image in the available space.
    If checkBox1.Checked Then
        pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
    Else
        pictureBox1.SizeMode = PictureBoxSizeMode.Normal
    End If
End Sub

Private Sub closeButton_Click(sender As Object, e As EventArgs) Handles closeButton.Click
    ' Close the form.
    Me.Close()
End Sub

Private Sub backButton_Click(sender As Object, e As EventArgs) Handles backButton.Click
    ' Show the color picker dialog box. If the user chooses OK,
    change the
    ' PictureBox control's background to the color the user chose.
    If colorDialog1.ShowDialog() = DialogResult.OK Then
        pictureBox1.BackColor = colorDialog1.Color
    End If
End Sub

Private Sub clearButton_Click(sender As Object, e As EventArgs) Handles clearButton.Click
    ' Clear the picture.
    pictureBox1.Image = Nothing
End Sub

Private Sub showButton_Click(sender As Object, e As EventArgs) Handles showButton.Click
    ' Show the Open File dialog. If the user chooses OK, load the
    ' picture that the user chose.
    If openFileDialog1.ShowDialog() = DialogResult.OK Then
        pictureBox1.Load(openFileDialog1.FileName)
    End If
End Sub
End Class
```



The screenshot shows a blog post titled "Support Statement for Visual Basic 6.0 on Windows 8" by Nikosan, dated 29 Apr 2012 1:26 AM. The post content states that the Visual Basic team is committed to "It Just Works" compatibility for VB6 applications on Windows Vista, Windows Server 2008 including R2, Windows 7, and Windows 8. It also mentions that the Visual Basic 6.0 runtime files continue to be 32-bit only and all components must be hosted in 32-bit application processes. The post includes a "RATE THIS" section with 5 stars and a "COMMON TASKS" sidebar with links like Blog Home, Email Blog Author, About, Share this, RSS for comments, RSS for posts, and Atom. There is also a "SEARCH FORM" at the bottom.

Microsoft si è impegnata a garantire il supporto per VB6 per tutta la vita di Windows 8. Peraltro, siamo già a Windows 10. (<http://bit.ly/vb6support>)

codeplex.com. Attenzione a effettuare le installazioni con i privilegi di amministratore, controllare gli aggiornamenti richiesti e, soprattutto, non tentare l'installazione su una macchina con OS a 64 bit: ai tempi di Visual Basic solo i supercomputer avevano tutti quei bit in una word. Se anche funzionasse l'Ide, non ci sono molte speranze per i controlli.

Con questo, si dovrebbe poter usare Windows 8 come macchina di sviluppo e macchina target per l'installazione. C'è, naturalmente, un ma. Bisogna verificare se i componenti esterni da cui dipende l'applicazione sono compatibili col sistema e, in caso negativo, se sono disponibili aggiornamenti del fornitore per risolvere il problema. Comunque sia, c'è ancora speranza per chi vuole aspettare tempi migliori prima di prendere una decisione.

## CASO 2 RISCRIVERE



Questa è l'opzione radicale, quella che piace agli audaci e agli insofferenti per la staticità. Si tratta di una strada faticosa, ma i pericoli sono legati alle incognite della realizzazione di un sistema, quelle che si devono anticipare

qualunque sia il progetto che si intende realizzare, per il solo fatto di avere iniziato a realizzarlo. La scelta più importante è quale direzione prendere. Ci sono due variabili da considerare: la piattaforma e il linguaggio.

La piattaforma si suppone che rimanga Windows, a meno che uno si faccia attrarre da Linux o da Java, ma anche su Windows occorre scegliere fra diversi campi di sviluppo. Esistono tre bersagli diversi per la creazione di applicazioni: il vecchio desktop, il nuovo e, all'interno del vecchio desktop, WinForms e Wpf.

Il vecchio desktop, quello di Windows 7 non molla e Microsoft lo sta sempre più riportando sul palco, dopo avere tentato – sbagliando – di sistemarlo dietro le quinte. In realtà, uniformare il computer al tablet è stata una mossa avventata e, in ultima analisi, sbagliata. L'interazione ideale su uno schermo da 10 pollici non è detto sia altrettanto sensata su un monitor da 23 pollici. Con Windows 8.1 il vecchio desktop è tornato l'ambiente principale di interazione. C'è ancora il cambio di ambiente quando si passa a un'applicazione per il nuovo desktop, ma questo scomparirà nel prossimo Windows 9, anzi 10, che vedrà convivere applicazioni vecchie e nuove in diverse finestre sul desktop.

Quindi, il desktop tradizionale non ci sta lasciando, ma è anche vero che le applicazioni con l'interfaccia moderna

non saranno più chiuse nel loro ambiente. I modelli di interazione delle nuove applicazioni sono interessanti e funzionali, quindi attenzione al senso di vecchio che daranno le applicazioni tradizionali.

Dovendo orientarsi nella scelta, il campo che richiede meno investimento è quello delle applicazioni desktop tradizionali (WinForms). Con WinForms lo sviluppatore VB6 va sul sicuro, portandosi dietro buona parte dei propri ricordi sul nome degli elementi di interfaccia e quello che fanno. Si tratta di una scelta prudente, ma bisogna considerare che prima o poi ci si dovrà muovere anche da qui e abbracciare del tutto lo sviluppo di interfacce utente con specificazione dell'interfaccia in un dialetto Xml chiamato Xaml, come Wpf.

La motivazione per scegliere Xaml è che si tratta della stessa tecnologia che muove le applicazioni per Windows Store e per Windows Phone, una scelta obbligata per chi vuole vendere applicazioni con l'interfaccia moderna nello shop di Microsoft.

Nel passaggio a VisualBasic.net ci sono diverse novità che riguardano il linguaggio, quindi sostanzialmente, si tratta di scegliere se fare un solo gradino o due insieme.

Può aiutare a decidere un progetto di prova realizzato seguendo le istruzioni di un ottimo e esteso tutorial sul sito Microsoft ([bit.ly/wpfindro](http://bit.ly/wpfindro)).

Infine, chi è pronto ad un salto nel vuoto può decidere di traslocare su C#. Il moderno linguaggio di Microsoft non è così complicato da non poter essere appreso in tempi brevi e ha una forte somiglianza con altri linguaggi moderni, come Java. Questo, è un vantaggio sia per chi Java lo conosce già, sia per chi vuole tenersi aperta una porta verso altre possibilità di lavoro.

Per capirci, se prendiamo uno dei siti più popolari di condivisione di conoscenza fra sviluppatori, StackOverflow e guardiamo la pagina che raggruppa le domande per linguaggio ([stackoverflow.com/tags](http://stackoverflow.com/tags)), possiamo osservare che ci sono 700.000 domande su C# e 74.000 su Visual Basic .net, mentre nell'indice Tiobe ([bit.ly/tiobeidx](http://bit.ly/tiobeidx)) C# è subito dietro C, Java, Objective-C e C++, mentre Visual

### Riscrittura

La scelta è radicale ma può essere l'occasione per scegliere una piattaforma che offra opportunità di crescita.

Può aiutare a decidere un progetto di prova realizzato seguendo le istruzioni di un ottimo e esteso tutorial sul sito Microsoft ([bit.ly/wpfindro](http://bit.ly/wpfindro)).

Infine, chi è pronto ad un salto nel vuoto può decidere di traslocare su C#. Il moderno linguaggio di Microsoft non è così complicato da non poter essere appreso in tempi brevi e ha una forte somiglianza con altri linguaggi moderni, come Java. Questo, è un vantaggio sia per chi Java lo conosce già, sia per chi vuole tenersi aperta una porta verso altre possibilità di lavoro.

Per capirci, se prendiamo uno dei siti più popolari di condivisione di conoscenza fra sviluppatori, StackOverflow e guardiamo la pagina che raggruppa le domande per linguaggio ([stackoverflow.com/tags](http://stackoverflow.com/tags)), possiamo osservare che ci sono 700.000 domande su C# e 74.000 su Visual Basic .net, mentre nell'indice Tiobe ([bit.ly/tiobeidx](http://bit.ly/tiobeidx)) C# è subito dietro C, Java, Objective-C e C++, mentre Visual

Basic .net è all'undicesimo posto. Non ci sono differenze di qualità di codice e velocità fra Visual Basic .net e C#, ma è indiscutibile che C# è mainstream. Tirando le somme, la via più breve è passare a Visual Basic .net con WinForms. Chi è interessato a vendere le applicazioni nello shop di Microsoft deve convertirsi a Xaml e chi vuole aprirsi tutte le opzioni può tentare di fare un salto definitivo sul linguaggio più diffuso.

## CASO 3 IL PORTING



Portare un progetto dalla vecchia versione sarebbe bello e, per un po', Microsoft ha anche offerto un tool di migrazione, sparito dall'offerta diversi anni fa dopo una lunga serie di valutazioni negative da parte degli utenti. Ci sono parecchie classi di problemi nel passaggio dal vecchio al nuovo ambiente.

In primo luogo il linguaggio è cambiato e ci sono conflitti dove prima non c'erano. Ad esempio, VB6 aveva un sistema dinamico di tipizzazione dei dati, che liberava il programmatore

dall'onere di dichiarare un tipo. Questo rendeva Visual Basic deliziosamente ecumenico, oltre che lento e ambiguo. I variant sono scomparsi e questo è certamente lo scoglio numero uno con cui si confronterà chi affronta il porting di un progetto VB6.

Curiosamente, i linguaggi di ora, come Swift, Go, Java e C# hanno algoritmi di tipizzazione dinamica che mantengono la tipizzazione stretta delle variabili, ma la deducono dall'uso che se ne fa quando non ci sono ambiguità, mentre Visual Basic .net è rimasto indietro.

Un'analisi dettagliata dei problemi di aggiornamento di un progetto Visual Basic sono in un ebook che è a disposizione sul sito Microsoft (bit.ly/vb6upgradebook). Il libro non è breve, così come non è poco lo sforzo, questo già basta per classificare l'impresa di un porting come un'attività da intraprendere solo se il gioco vale veramente la candela.

Microsoft non offre sostegno esplicito, ma esiste un forum dedicato ai problemi di porting su Msdn (social.msdn.microsoft.com/Forums?forum=vbinterop).

Apparentemente ci sono società specializzate nel porting, come Artinsoft (www.mobilize.net/solution/vb-upgrade-companion) e Migration Partner (www.vbmigration.com). La prima offre una versione gratuita del tool di conversione. Entrambe offrono consulenza, che non sarà economica, ma ha un costo confrontabile, a spanne, con sei mesi di un posto di lavoro tecnologico. Non è poco, ma

può essere conveniente per una grande azienda. Per le piccole software house avventurose ci sono tool di migrazione verso Java: uno che gira come add-on dell'ambiente di sviluppo VB (bit.ly/vb62java) e l'altro come applicazione Java (code.google.com/p/vb6-to-java). Entrambi sono gratuiti. Il secondo sembra abbastanza elementare, ma ha il pregio di essere open source.

A titolo di curiosità, segnaliamo anche Jabaco, un ambiente integrato con un linguaggio che evoca VB6 e un compilatore che genera codice da eseguire con la Jvm. Questo ambiente permette di scrivere codice simile al Visual Basic e eseguirlo ovunque c'è una macchina virtuale Java. Jabaco sembra il prodotto di un singolo sviluppatore e l'ultima release è del 2009, ma è gratuito e richiede solo una registrazione.

Ricapitolando, tutti gli strumenti di porting automatico del codice che abbiamo potuto vedere nella nostra carriera, hanno sempre lasciato trucoli lungo i bordi che dovevano essere essere limati a mano. È inevitabile, perché tradurre da un linguaggio all'altro è un compito impossibile da automatizzare. Ogni dialetto ha frasi idiomatiche e costruzioni sintattiche che possono essere innaturali o inesistenti in altri linguaggi.

In sintesi, un progetto di traduzione automatica va tentato solo con supporto specializzato, risorse a disposizione e quando davvero ne vale la pena. In tutti gli altri casi, una riscrittura da zero, a specifiche bloccate può essere un processo più controllabile.

**CODEARCHITECTS**  
TOMORROW'S SOFTWARE. TODAY

HOME CHI SIAMO SVILUPPO MIGRAZIONE DA VB6 CONSULENZA FORMAZIONE TECNOLOGIE CLIENTI CONTATTI

Perché sceglierli? Ecco 10 buone ragioni

**Microsoft Regional Director PROGRAM**

Per otto anni i fondatori di Code Architects **Francesco Balena** e **Giuseppe Dimairo** sono stati gli unici **Microsoft Regional Director** per l'Italia; partecipano come speaker a conferenze tecniche in Europa e America e sono autori di **7 best-seller** Microsoft Press tradotti in una dozzina di lingue, tra cui il testo sulle **coding guidelines** per .NET Framework.

## Migrazione da VB6

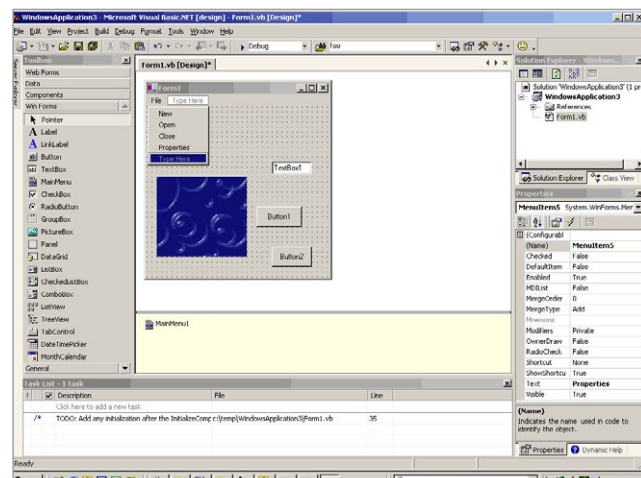
Oltre ad aver realizzato **VB Migration Partner**, il miglior strumento per migrare applicazioni Visual Basic verso .NET, Code Architects è in grado di offrire servizi di migrazione chiavi in mano per convertire applicazioni esistenti VB6 verso .NET Framework, convenienti.

**Perché scegliere i servizi di migrazione di Code Architects?**

- Decine di milioni di righe di codice migrate con successo verso .NET in oltre 15 nazioni in tutto il mondo
- Una squadra di programmatori esperti in VB6 e .NET, guidata da Francesco Balena, guru di Visual Basic e autore di questo linguaggio per Microsoft Press
- La nostra comprovata metodologia convert-test-fix, che permette di modificare ed estendere il codice di migrazione a .NET.

**VB Migration Partner**  
Code Architects è l'unica azienda italiana che partecipa ai programmi mondiali **ISV NXT** (Microsoft ISV modernizzazione di applicazioni legacy VB6) e **VSIP** (Visual Studio Industry Partner). Dal 2008, anno del lancio commerciale, VB Migration Partner ha raggiunto i seguenti risultati:

- copertura completa delle features delle features di Visual Basic 6. Per maggiori informazioni vedere il sito [www.vbmigration.com](http://www.vbmigration.com)
- stretta collaborazione con Microsoft in Europa, Stati Uniti, Giappone e Australia
- rete di partner in Europa, Stati Uniti, Giappone, Israele e India, comprendente aziende come Packard, EDS, Fujitsu, CSC, Matrix e Datamatic
- case-study di migrazione da VB6 sul sito di **Microsoft USA**



La migrazione può essere assistita da strumenti sofisticati, ma occorre un intervento umano. Diversi consulenti se ne occupano, anche in Italia.

Una visione familiare, sempre più difficile da trovare. Oramai accessibile soltanto sui supporti di Msdn.