Benefits of using Vulcan.NET for application development

Overview

Vulcan.NET is the next generation of the xBase family of languages. It provides a high level of backwards-compatibility with the Visual Objects language, while at the same time bringing it into the 21st century with all the features you would expect in a modern programming language. In addition, Vulcan.NET brings all the benefits of the .NET platform to the Visual Objects language, opening up a whole new world of opportunities for xBase programmers.

There is much to be gained by adopting Vulcan.NET as the language of choice for application development. This document highlights and explains some of the reasons it is important to consider this approach. In summary, some of the key benefits are:

- Use of core part of current and future versions of Windows
- Access to new technologies provided with updates to Windows
- Improved memory management system and garbage collector
- Ability to use a familiar language
- Potential for speed improvements in your application
- A more stringent, consistent and extensible compiler
- New language features
- New data types
- Built-in localisation features
- Integrated code access security
- High level of compatibility with existing Visual Objects code
- Able to use existing Visual Objects classes
- Ability to use the thousands of classes from the .NET framework
- With VO class/functions wrappers had to be written – the .NET framework the classes can be used directly
- Ability to use legacy 32-bit Windows functionality when necessary
- Ability to use both newer .NET techniques and older VO-style code in the one application
- Object-oriented programming is more flexible, maintainable than the Win32 functional programming
- Ability to use assemblies provided by third party software vendors
- Easy use of assemblies written in other .NET compliant languages, e.g. C#. VB.NET
- Assemblies written in Vulcan.NET can be used anywhere a .NET assembly is allowed
- Choice of development environment
- More stable use of COM
- Ability to generate a Visual Studio solution from code in the VO IDE.
- Major VO third party tools have been or are being converted to Vulcan
- Obfuscation tools can be used to protect code
- More efficient deployment of applications, avoiding "DLL-hell"
- 64-bit Windows support provided through .NET
- Ability to develop code targeting the desktop, for ASP.NET, for web services, for portable devices

The following section provides more details about these points. Information about the availability of Vulcan.NET and its developer, GrafX, are at the end of the document. We trust you will see why Vulcan.NET is the path to your future application development.

Use of core part of current and future versions of Windows

The .NET framework is a key part of Microsoft's strategies for current and future versions of Windows. Microsoft Vista includes version 3.0 of the framework, which is also available on Windows XP and Windows Server 2003. Microsoft's design goals for the .NET framework include:

- Interoperability: because interaction between new applications and legacy systems is important, the .NET framework provides mechanism to interface with code outside of the .NET environment. Use of COM components is provided, and the ability to call functions in dynamic-link libraries is possible using the platform-invoke feature
- Common Runtime: .NET compliant languages all compile to a Common Intermediate Language. The intermediate code is then compiled by the Just-In-Time compiler to native code for the target machine.
- Language Independence: The .NET framework has a Common Type System, making it easy for code produced using different .NET languages to work together seamlessly.
- Base Class Library: The framework has a huge library of classes encapsulating a wide expanse of common features, available to all .NET languages
- Simplified deployment: The framework is designed so that installation of new software does not interfere with current software.
- Security: The .NET security model addresses system vulnerabilities and controls program execution to minimise exploitation by malicious or dangerous code.
- Portability: Applications written using the .NET framework can target different platforms; applications can be developed to run on the desktop or on mobile devices.

See http://msdn2.microsoft.com/en-au/netframework/default.aspx.

Access to new technologies provided with updates to Windows

Development of the .NET framework is ongoing; version 3.5 was released in late 2007. The update built incrementally on the features supplied with .NET Framework 3.0. In addition, .NET Framework 3.5 contains a number of new features in several technology areas which have been added as new assemblies to avoid breaking changes. They include:

- Deep integration of Language Integrated Query (LINQ) and data awareness. LINQ-enabled languages enable you to filter, enumerate, and create projections of several types of SQL data, collections, XML, and DataSets by using the same syntax. LINQ support in Vulcan is currently being developed.
- ASP.NET AJAX provides more efficient, more interactive, and highly-personalized Web experiences that work across all the most popular browsers.
- Full tooling support in Visual Studio 2008 for WF (Workflow Foundation), WCF (Windows Communication Foundation), and WPF (Windows Presentation Foundation), including the new workflow-enabled services technology.
- New Web protocol support for building WCF services including AJAX, JSON, REST, POX, RSS, ATOM, and several new WS-* standards.
- New classes in .NET Framework 3.5 base class library (BCL) that address many common requests.

See http://www.microsoft.com/downloads/details.aspx?FamilyId=333325FD-AE52-4E35-B531-508D977D32A6.

Improved memory management system and garbage collector

Vulcan.NET makes use of the same memory management system as all .NET compliant languages, including C# and VB.NET. The system, developed by Microsoft and finely tuned to work efficiently and effectively with Windows, is a generation further on compared to the memory management used in Visual Objects. See http://msdn.microsoft.com/msdnmag/issues/1100/gci.

Ability to use a familiar language

Microsoft encourages the development of additional languages for .NET. It recognizes that since there is a significant amount of existing code, and a large number of programmers with existing language skills, that it is often pragmatic to migrate the code and skill-set rather than completely re-write applications in a new language. There are numerous language offerings for the .NET framework, some of which can be found here: http://www.dotnetlanguages.net/DNL/Resources.aspx.

Potential for speed improvements in your application

When using the base data types of .NET, Vulcan applications have been demonstrated to operate faster than the equivalent Visual Objects versions. For example, consider a loop such as this:

```
LOCAL i, j AS INT
FOR i := 1 UPTO 1000000000
        j += 1
NEXT
```

In Visual Objects this might take 8 seconds to execute, whereas in Vulcan it would take 3 seconds.
In addition, there are .NET classes available that offer speed improvements: for example, the StringBuilder class provides great speed gains when a large amount of string concatenation is required.

A more stringent, consistent and extensible compiler

The Vulcan.NET compiler, while offering high Visual Objects compatibility, is a brand new compiler. It is rule-based, providing higher consistency than VO and designed to be extensible, allowing for future updates, including support of a number of emerging .NET features. Some of the early indications of the compiler's flexibility are the support of the use of interfaces, and ability to use generic types.

New language features

Vulcan introduces a number of language features that are new to VO: abstract, static and sealed classes, attributes, delegates. These are discussed in detail in the Vulcan documentation, included with the trial version of Vulcan.NET, downloadable from http://www.govulcan.net. Further features are described in the Getting Started book, also downloadable from http://www.govulcan.net.

New data types

The .NET Framework types are the foundation on which .NET applications, components, and controls are built. The .NET Framework includes types that perform the following functions:

- Represent base data types and exceptions.
- Encapsulate data structures.
- Perform I/O.
- Access information about loaded types.
- Invoke .NET Framework security checks.
- Provide data access, rich client-side GUI, and server-controlled, client-side GUI.

The types include 64-bit integer and 96-bit decimal types.
See http://msdn2.microsoft.com/en-us/library/hfa3fa08(VS.71).aspx.

Built-in localisation features

The .NET framework has a number of features that make it easier to produce localised versions of your application. For example, when using Visual Studio, when you open a Windows Form in the designer you can set the Localisation property to True. Then you design your form using the default language. When you are ready, you switch the Language property of the form, for example to Mongolian. The designer will generate a resource for that language, and you can use either the designer or the resource editor to put in the appropriate Unicode strings. You can do this for each language you want to provide. When the application is run .NET will check the current locale of the machine executing the program and use the matching resources if it finds them.

Further access to culture-related information is provided through the System.Globalization namespace: http://msdn2.microsoft.com/en-us/library/system.globalization.aspx.

Integrated code access security

Code Access Security is an integral part of the .NET framework, intended to protect users from potentially malicious applications. System administrators control the security policy of the users' machines and so determine what code is allowed to run. Based on the origin and authorship of the application, the >NET framework can throw an exception when an illegal operation is requested. See http://msdn2.microsoft.com/en-us/library/aa302330.aspx.

High level of compatibility with existing Visual Objects code

As previously mentioned, the Visual Objects class libraries, including GUI, RDD and Internet classes, have been rebuilt using Vulcan.NET. This demonstrates a hign degree of

compatibility. Usually, it is only low-level code that requires manual attention, as documented in the Vulcan help.

Able to use existing Visual Objects classes

Since the VO class libraries have been rebuilt as .NET assemblies they can be easily used from Vulcan.NET applications. The coding techniques to use these assemblies are the same as those used to program native VO applications.

Ability to use the thousands of classes from the .NET framework

All of the .NET framework classes are available for use within Vulcan.NET applications. The samples page of http://www.govulcan.net shows how to use just some of the classes: printing, email, graphics, internet, registry, XML, COM automation, etc.

With VO class/functions wrappers had to be written – the .NET framework the classes can be used directly

The framework classes do not require any intermediate wrapper in order to be used by Vulcan.NET applications; they can be used directly. This is in contrast to the situation with Visual Objects where all Win32 features need to have function prototypes created, and ideally have an object-oriented wrapper class written.

Ability to use legacy 32-bit Windows functionality when necessary

Vulcan.NET supports the Visual Objects syntax, _DLL FUNC, for declaring prototypes of functions in dynamic-link libraries. Once declared, these functions can be called from a Vulcan application in the same manner as they are called from a Visual Objects application. Probably the most extensive example of this is action is the source for the VO class libraries, which has been recompiled in Vulcan.NET.

Ability to use both newer .NET techniques and older VO-style code in the one application

Since Vulcan.NET is highly compatible with the Visual Objects language it is straight-forward to write VO-style code, and compile it with Vulcan.NET. It is also possible to use the Vulcan-built version of the VO class libraries, and at your own pace, add newer .NET features. For example, one of the sample applications demonstrated at the recent Vulcan conference showed a traditional master-detail VO-style window which used a helper .NET Windows Form to display product images for the details.

Object-oriented programming is more flexible, maintainable than the Win32 functional programming

The features of the .NET framework are provided as classes, and most object-oriented programming techniques can be used to make the most of them. Many classes can be subclassed when necessary to achieve a high degree of customisation.

Ability to use assemblies provided by third party software vendors

Since a goal of the framework is to be language-agnostic – smooth interoperability of code written in different languages – this means that most tools and class libraries that are targeted at one language can also be used easily by other languages. So many third party tools, or open source assemblies, or public domain assemblies that were originally developed with C#, or VB.NET, in mind can also be used by Vulcan.NET.

Easy use of assemblies written in other .NET compliant languages, e.g. C#. VB.NET

.NET compliant languages use common types, and compile to MS Intermediate Language. Once the MSIL is produced the language of origin used to code the assembly is not a major factor when it comes to executing the code. Because of this it is usually straight-forward to be able to use assemblies authored in different languages. It is possibly to have parts of applications written in different languages and still be part of the one Visual Studio solution.

Assemblies written in Vulcan.NET can be used anywhere a .NET assembly is allowed

Vulcan.NET assemblies can be used by applications written in other .NET languages, and can be used wherever a .NET assembly is required. As such, it is possible to use Vulcan.NET assemblies as part of ASP.NET applications, or to provide web services. A link to an example of this is provided at http://www.govulcan.net.

Choice of development environment

The default development environment for Vulcan.NET is Microsoft's Visual Studio. If you already have a professional version of Visual Studio, Vulcan is installed as an additional language alongside C#/VB.NET. If you do not already have Visual Studio a Vulcan-only version is provided.

A more VO-like development environment is available from Chris Pyrgas: Vulcan IDE. Details about it are listed at http://www.govulcan.net.

In some situations it might be useful to use any text editor to write code and then compile from the command prompt. It should also be possible to integrate Vulcan into editors such as Multi-edit, Ed for Windows or Sharp Develop.

More stable use of COM

The Microsoft implementation of COM is more robust than the COM interface layer provided by Visual Objects. Support for the use of COM and building of automation servers is provided as part of .NET, and can be used by Vulcan.NET. Making of the .NET COM support rather than VO's promises more stable COM applications.

Ability to generate a Visual Studio solution from code in the VO IDE.

Vulcan.NET is provided with a tool, called Transporter, that can generate a Visual Studio solution from the code in the VO IDE. Transporter applies a small set of updates to the code,

but mainly extracts the source from VO and writes it out as PRG files. The resulting solution can be opened and built in Visual Studio. This generated solution can in turn be used by Vulcan IDE or other tools to provide further features.

Major VO third party tools have been or are being converted to Vulcan

Pre-release versions of Vulcan compatible bBrowser, ReportPro, Classmate and RightSLE were demonstrated at the German conference at the end of 2007. Plans for a Vulcan version of VO2ADO were also announced.

Obfuscation tools can be used to protect code

All applications, even those created by Visual Objects, can be hacked or examined by use of low-level tools. .NET applications can be obfuscated, making decompilation difficult.
A community edition of an obfuscator is provided with professional versions of Visual Studio. A trial version is also available from http://www.preemptive.com.

More efficient deployment of applications, avoiding "DLL-hell"

The .NET Framework provides features including application isolation and side-by-side components for applications built with managed code on the .NET platform. .NET applications are self-describing, removing registry dependency. Versioning is built into and enforced by the platform, as is support for side-by-side versions of components. See http://msdn2.microsoft.com/en-us/library/ms973843.aspx.

64-bit Windows support provided through .NET

The 64-bit version of the .NET Framework enables tools and applications to run on 64-bit workstations and servers which provide increased performance and scalability by addressing more memory, processing more data per clock cycle and performing faster numeric calculations. See http://msdn2.microsoft.com/en-au/netframework/aa496329.aspx.

Ability to develop code targeting the desktop, for ASP.NET, for web services, for portable devices

.NET applications can be used in a wider range of situations than VO applications. As well as running of the desktop, Vulcan.NET applications can integrate with web servers and so be accessible from ASP.NET, or provide web services. They can also execute on portable devices. See http://www.govulcan.net for examples.

More Information

A sixty-day evaluation version of Vulcan.NET can be downloaded from http://www.govulcan.net. A Getting Started guide can also be downloaded from the site. There are competitive upgrade discounts available to users of Visual Objects and other xBase products. Vulcan.NET licences are provided with one year free maintenance releases. The maintenance releases will provide new functionality as it becomes available. Vulcan.NET licences are perpetual, and applications produced with it are royalty free. Vulcan.NET is also

available as part of the VOPS plan, providing early access to development versions and priority support. Visit http://www.govulcan.net for more details.