What is .Net?

The .NET Framework is a development and execution environment that allows different programming languages & libraries to work together seamlessly to create Windows-based applications that are easier to build, manage, deploy, and integrate with other networked systems. Currently there are over 22 different languages that are supported. Any of these languages can be used to write a .Net program and different languages can interoperate with in the same design. This is accomplished with the use of a multistage compiler that first compiles that code into the Microsoft Intermediate Language (MSIL). This is what allows all the different languages to work together. MSIL is a language that contains metadata that describes the code so that no type libraries or an Interface Definition Language is needed.

The .Net Framework is made up of three parts: The Common Language Runtime (CLR), The Framework Class Libraries (FCL) and ASP.NET. CLR is a language-neutral development & execution environment that provides services to help "manage" application execution similar to Java's Virtual Machine. So far this runtime is only available on Windows platforms although work is being done to port it to others. The FCL is a consistent, object-oriented library of prepackaged functionality. Together, the class libraries provide a common, consistent development interface across all languages supported by the .NET Framework. ASP.NET provides a programming model, and infrastructure, to make creating scalable, secure and stable applications faster, and easier than with previous Web technologies.

.Net Pros

.Net offers multiple language support currently with over 22 languages available. A programmer can use a combination of any of these languages to write their program and the framework will compile them natively regardless of the language or deployment method into a working .Net program. This enables developers to use the programming language that is most appropriate for a given task and to combine languages within a single application without any extra effort needed from the programmer.

The FCL provides a rich set of libraries similar to the Java API to make programming quicker. Over 350 third party tools are also available to aid in faster development times. Currently there is a lot of support and learning material available with more being written all the time. There are over 350 .Net books and over 750 .Net user groups available worldwide. .Net is becoming widely used and supported and therefore it is becoming a necessity to learn how to use it. There are over one million developers using .Net Visual Studio and thousands of leading companies are developing and deploying there applications using .Net. Web services are extremely efficient when used with .Net as .Net is optimized for XML and architected around XMLWeb Services.

Microsoft has developed the new language C# for use in conjunction with the .Net framework. It is Microsoft's answer to Java. It is a strongly-typed object-oriented language designed to give the optimum blend of simplicity, expressiveness, and performance. .Net also offers improved performance over J2EE in many applications such as web application hosting, web services and distributed transactions. As well the new version 1.1 Framework offers improved performance with the use of just in time compilation.

.Net Cons

.Net is yet another platform to consider and thus means more learning is necessary. New tricks and tweaks must also be learned. .Net is made by Microsoft and this can be bad for a variety of reasons besides the obvious ones. All support for .Net comes from Microsoft and thus it is easy for them to cover up bugs and issues that exist or simply pretend that they do not exist at all. Since .Net is only available on Windows platforms, Windows security problems become your security problems. Also this causes problems if you aren't running any Windows machines or if you have no servers running ASP.NET server. With new Microsoft software comes new hardware upgrades. “Ok you want a new Microsoft product the first thing you need is MORE RAM”. Large organizations with legacy hardware will not be able to port existing systems into .Net due to the hardware constraints. This is also a cost factor.

Many people have argued that multiple language support as Microsoft has advertised is actually a problem. It is said that by providing access to many different languages for one problem that solutions to problems become less elegant and more towards a hack. A jumble of different code segments in different languages might be used instead of a simple elegant solution using one language.

It appears that .NET applications consume huge amounts of memory. This is due to the fact that the runtime takes massive chunks of memory and then returns it to the OS when called for. The problem has been reported that this can become an issue when it slows web servers.

.Net programs are compiled into MSIL. Unlike native code binaries, MSIL contains metadata that makes it easy to decompile. Utilities exist that can convert compiled code back into C#. This is a very big problem for any developer who wants to keep source code confidential.

Lastly, .Net is very expensive. Because .NET is propriety, a license must be purchased for each developer on your team. This gets really pricey really fast considering an enterprise architect version of Visual Studio .Net has a suggested retail price of $2500 USD. Also, bulk licensing doesn't kick in until you reach 250 copies. 250 * $2500 = $62500 and that's just the cost of the software not to mention the cost of machines and operating systems etc.