**The Microsoft .NET Framework**

The .NET Framework is the infrastructure for the Microsoft .NET platform.

The .NET Framework is an environment for building, deploying, and running Web applications and Web Services, The .NET Framework represents a unified, object-oriented set of services and libraries that embrace the changing role of new network-centric and network-aware software. In fact, the .NET Framework is the first platform designed from the ground up with the Internet in mind.

ASP.NET was developed in direct response to the problems that developers had with classic ASP. Since ASP is in such wide use, however, Microsoft ensured that ASP scripts execute without modification on a machine with the .NET Framework (the ASP engine, ASP.DLL, is not modified when installing the .NET Framework). Thus, IIS can house both ASP and ASP.NET scripts on the same machine.

**.NET Architecture**



**Common Language Runtime**

The Common Language Runtime (CLR) is a layer between an application and the operating system it executes on. The CLR simplifies an application's design and reduces the amount of code developers need to write because it provides a variety of execution services that include memory management, thread management, component lifetime management, and default error handling.

The key benefit of the CLR is that it transparently provides these execution services to all applications, regardless of what programming language they're written in and without any additional effort on the part of the developer. The CLR is also responsible for compiling code just before it executes. Instead of producing a binary representation of your code, as traditional compilers do, .NET compilers produce a representation of your code in a language common to the .NET Framework: Microsoft Intermediate Language (MSIL), often referred to as IL.

When your code executes for the first time, The CLR invokes a special compiler called a Just In Time (JIT) compiler, which transforms the IL into executable instructions that are specific to the type and model of your system's processor. Because all .NET languages have the same compiled representation, they all have similar performance characteristics. This means that a program written in Visual Basic .NET can perform as well as the same program written in Visual C++ .NETCommon Type System

**The Common Type System (CTS)**

is a component of the CLR and provides a common set of data types, each having a common set of behaviors. for example, the "String data type" maps to the CTS System. String class. Therefore, if a JScript .NET client needs to communicate with a component implemented in C# .NET, the client doesn't have to do any additional work to exchange information because it's using a type common to both JScript .NET and VB .NET. The CTS eliminates many interoperability problems that exist outside .NET.

**.NET Class Library**

.NET Class Library contains hundreds of classes that model the system and services it provides. To make the .NET Class Library easier to work with and understand, it's divided into namespaces. The root namespace of the .NET Class Library is called System, and it contains core classes and data types, such as Int32, Object, Array, and Console.

Secondary namespaces reside within the System namespace. Examples of nested namespaces include the following:

*§ System.Diagnostics: Contains classes for working with the Event Log*

*§ System.Data: Makes it easy to work with data from multiple data sources*

*§ System.IO: Contains classes for working with files and data streams*

The figure below illustrates the relationship between some of the major namespaces in the.NET Class Library.

****

**unifying components**

The unifying components are the means by which you can access the services the .NET Framework provides:

*§ ASP.NET*

*§ Windows Forms*

*§ Visual Studio .NET*