Introducción a .NET

.NET

- Conjunto de tecnologías construidas a partir de una estrategia de Microsoft.
- .NET Incluye:
  - Aplicaciones de servidor
  - Entorno de desarrollo
  - Componentes clave que se integran al sistema operativo.
  - Servicios
  - Plataforma de desarrollo (.NET framework)

Objetivos de .NET

- Ejecutar software realizado en cualquier lenguaje sobre cualquier dispositivo (teléfonos, PDAs, Tablet PCs).
- Uso de internet en las aplicaciones. Proporcionar servicios mediante:
  - Un nuevo modelo para compartir componentes.
  - Un concepto y modelo de programación: Servicio Web XML

¿Por qué .NET?

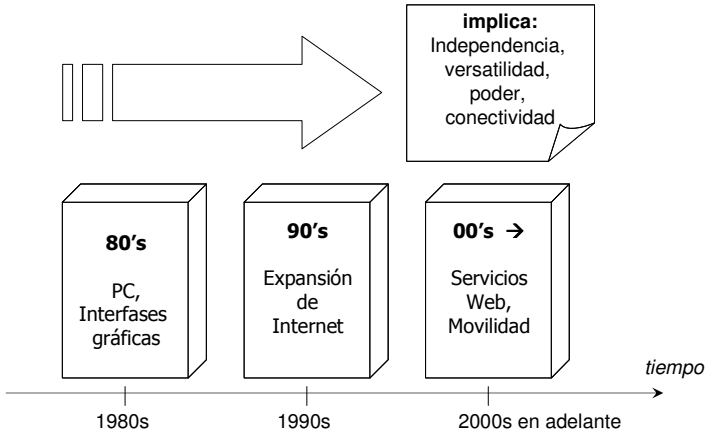
→ La programación Windows (y el propio S.O.) se estaban tornando complejos.

→ Multitud de servicios duplicados.

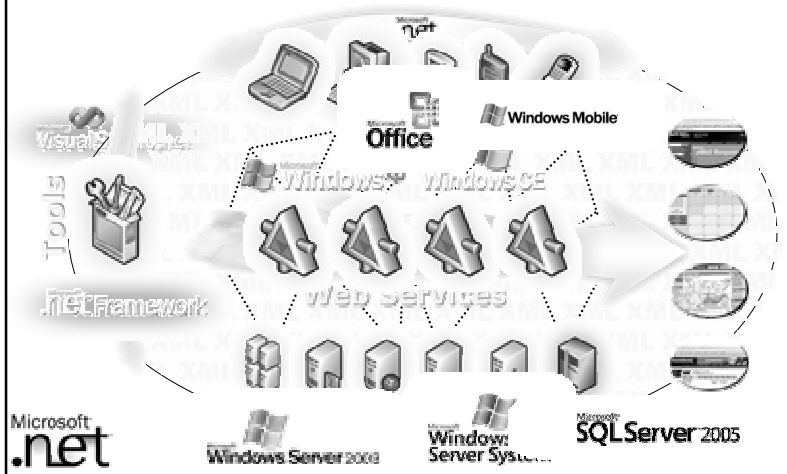
→ Limitada reutilización del código.

→ "El infierno de las DLL Win32" (Conflictos entre aplicaciones con una librería común en diferentes versiones).

…Historia…



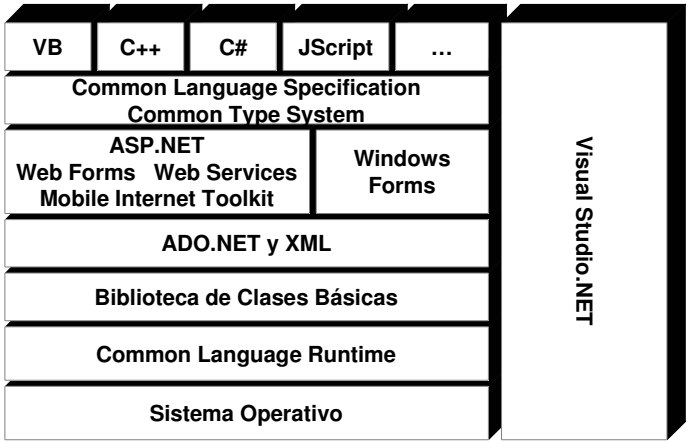
Plataforma.NET



.NET Framework

- Infraestructura de desarrollo compuesta por diversos recursos como:
  - CLR (Common Language Runtime)
  - BCL (Base Class Library)
  - ASP.NET (Active Server Pages .NET)

.NET Framework



CLR (Common Language Runtime)

- **RUNTIME** .- Componente que provee todos los recursos necesarios para que las aplicaciones escritas en un lenguaje de programación sean ejecutadas.
- Todos los lenguajes .NET utilizan un **único** RUNTIME para su ejecución.
- El CLR asegura que **cualquier** lenguaje .NET soporte orientación a objetos y no requiera manejar la memoria.

BCL (Base Class Library)

Biblioteca de aproximadamente 3000 clases ya compiladas en forma de ensamblados, las cuales forman parte de la infraestructura, y pueden ser utilizadas libremente por el desarrollador sin importar el lenguaje utilizado.

La BCL incluye tipos de datos comunes para todos los lenguajes .NET

Otros recursos…

- Formularios Windows (Windows Forms) Elementos gráficos para las formas en aplicaciones de "escritorio".
- Formularios WEB (Para su ejecución en internet).
- ASP.Net (Active Server Pages).
- ADO.Net (Modelo de acceso a datos).

.NET en resumen

.NET es el conjunto de nuevas **tecnologías** en las que **Microsoft** ha estado trabajando durante los últimos años con el objetivo de obtener una **plataforma sencilla** y **potente** para distribuir el software en forma de **servicios** que puedan ser suministrados **remotamente** y que puedan comunicarse y combinarse unos con otros de manera totalmente **independiente de** la plataforma, lenguaje de programación y modelo de componentes con los que hayan sido desarrollados.

¿Qué se necesita?

- Para **desarrollar** aplicaciones para .NET
  - Microsoft .NET Framework SDK
- Para **ejecutar** aplicaciones .NET
  - Microsoft .NET Framework Redistributable
- Donde obtenerlos (gratuitamente)
  - msdn.microsoft.com/net/
  - Mas información: www.microsoft.com/net
- Versiones
  - .NET Framework
    - v1.0, 1.1
    - **v2.0**
  - .NET Compact Framework
    - v1.0 y **v2.0**

Visual Studio Versiones Express 2005

- Versiones plenamente funcionales y gratuitas:
  - Visual Basic 2005
  - Visual C# 2005
  - Visual C++ 2005
  - Visual J# 2005
  - Visual Web Developer 2005
  - SQL Server 2005
- Descargarlas de:
  http://msdn.microsoft.com/vstudio/express/

Sharp Develop

- Una alternativa a Visual Studio es SharpDevelop
  - Entorno Visual
  - Funcional
  - Gratuito
  - Open Source
  - En constante desarrollo
  - Ocupa menos recursos del sistema
- Se puede descargar desde:
  http://www.icsharpcode.net

Introducción a C#

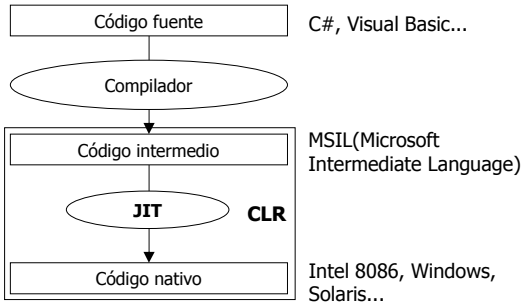
C# .Net

- Lenguaje moderno, orientado a objetos.
- Sintaxis similar a C++ y Java.
- Desarrollado por Anders Hejlsberg (Creador de Turbo Pascal).
- Combina la facilidad de Visual Basic y el poder y la flexibilidad de C++
- Se llama C# pues se considera una versión avanzada de C++ que llamarían C++++
- Se pueden crear muchos tipos de aplicaciones (Consola, Windows, Web, Moviles, etc.).
- Acceso completo a la plataforma .NET

¿Qué pasa al compilar / ejecutar un programa en C# ?

- Los programas creados en C# se traducen a instrucciones "MSIL".
- Justo antes de su ejecución, las instrucciones en MSIL se traducen a instrucciones nativas de la plataforma en la que se esté trabajando.
- La conversión de instrucciones MSIL a instrucciones nativas se lleva a cabo por el JIT (Just in Time Compiler).
- El código MSIL generado por C# es idéntico a cualquier otro lenguaje .NET

¿Qué pasa al compilar / ejecutar un programa en C# ?



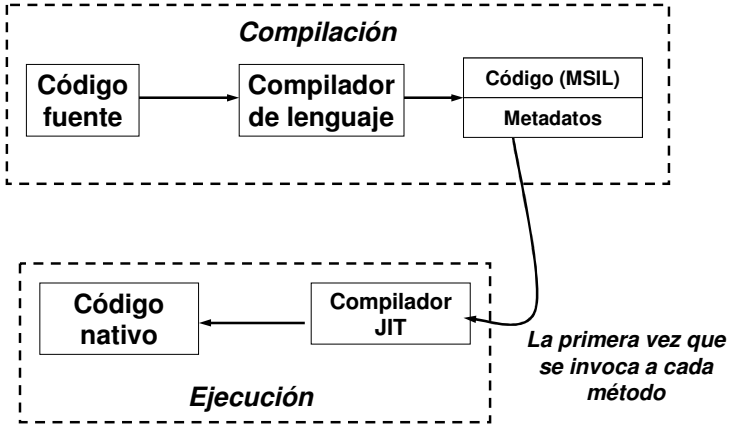
MSIL

- MicroSoft Intermediate Language (Lenguaje Intermedio de Microsoft).
- Es un lenguaje en formato binario con instrucciones de bajo nivel entendidas solamente por el compilador JIT.
- Puede pensarse en el MSIL como un "Lenguaje ensamblador intermedio orientado a objetos propietario de Microsoft"

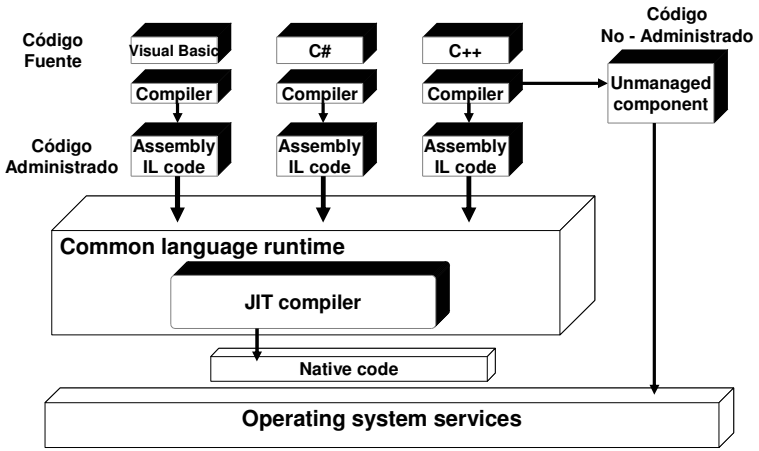
JIT

- Just In Time Compiler (Compilador Justo a Tiempo).
- El JIT :
  - Toma el código MSIL justo antes de ejecutarse.
  - Verifica que el código MSIL sea seguro y sin errores.
  - Lo convierte a código máquina optimizado para el procesador y sistema operativo específico.
  - Ejecuta el código final generado.

Compilación y ejecución de código



Código administrado y no-administrado



Tipos de código aceptado en los programas dentro del .NET framework

- **Código Administrado (Managed).-**
  - Son aquellas instrucciones que se ejecutan dentro del contexto ofrecido por el framework .NET
  - El manejo de los recursos y la memoria corren por cuenta del framework.
  - Un programa que utiliza solo clases y tipos del framework, genera código administrado (MSIL).
  - El código administrado es seguro, válido y se encuentra optimizado.
- **Código No Administrado (Unmanaged).-**
  - Son instrucciones que se salen del contexto del framework .NET
  - Es código inseguro y no optimizado. Generalmente de bajo rendimiento.
  - Se acepta en los programas, pero el framework no se responsabiliza de él.