Lenguaje de Cuarta Generación de Microsoft: .NET BACHILLERATO EN CIENCIAS DE LA COMPUTACIÓN E INFORMÁTICA DE LA UNIVERSIDAD DE COSTA RICA

Carlos Gourzong Gómez A32314 E-mail: cgourzongg@yahoo.es

Alonso Ramírez Araya A44254 E-mail: aramireza@gmail.com

Escuela de Ciencias de la Computación e Informática

Universidad de Costa Rica

Abstract

Microsoft .Net is a group of programs and services, or a platform as it is called, which provides a set of tools to develop and execute software applications. Its main objective is to allow a fast development which is independent from the platform where it executes on.

.Net promotes to take a greater benefit to the increasing development in the capacities from the networking connections, the new standards of communication and the potential of the personal computers.

The platform of .Net is made up of three main components: the surroundings of execution, the library of classes and the surroundings of development which are included in .Net Framework and is necessary for the execution of the applications .Net.

Key words:

.Net, Framework, Base Class Library, Common Language Runtime, Common Language Specification

Resumen

Microsoft .Net es un conjunto de programas y servicios o una plataforma como lo llaman, la cual brinda herramientas para producción y ejecución de aplicaciones de software. Su objetivo principal es permitir un desarrollo veloz e independiente de la plataforma donde corra.

.Net promueve sacar un mayor provecho al creciente desarrollo en las capacidades de las conexiones de red, los nuevos estándares de comunicación y el potencial de las computadoras personales.

La plataforma de .Net se compone de tres componentes principales: el entorno de ejecución, la biblioteca de clases y el entorno de desarrollo los cuales se incluyen en el .Net Framework y son necesarios para la ejecución de las aplicaciones .Net

Palabras Claves:

.Net, Framework, Base Class Library, Common Language Runtime, Common Language Specification

1. Introducción

Microsoft .Net se introdujo al mercado a principios del año 2000 y se puede considerar como la competencia da Microsoft a la plataforma Java de Sun Microsystems y brinda a su usuarios entre otras cosas: un grupo de lenguajes para el desarrollo de aplicaciones cada uno con su respectivo compilador, Un único IDE con todas las utilidades necesarias para el desarrollo en cualquiera de los lenguajes soportados, una colección de módulos de software a disposición del desarrollador para ser reutilizados en nuevas aplicaciones y una maquina virtual que se encarga de ejecutar las aplicaciones desarrolladas en .Net

2. Componentes del .Net Framework

El Framework o Marco de Trabajo, es la pieza más importante dentro del engranaje de .Net. Es usado tanto en la etapa de desarrollo como en la etapa de ejecución de la aplicación.

El Framework es necesario para ejecutar cualquier aplicación desarrollada con .Net, únicamente para las aplicaciones web el cliente no necesita tener el Framework instalado de manera local, ya que tiene acceso a la aplicación a través de un navegador; sin embargo, el navegador debe tener el Framework instalado.

El Framework se puede instalar sobre cualquier sistema operativo Windows superior a Windows 98 y actualmente se puede obtener gratis en internet, en cualquiera de sus tres presentaciones:

Redistributable Package:

Es la versión "lite" del producto incluye únicamente los componentes necesarios para la ejecución de programas desarrollados bajo .Net, este paquete es el que generalmente se instala en las máquinas del usuario cuando se le entrega el producto.

SDK:

Es una versión más completa con herramientas para desarrollo (compiladores, depuradores, otros), este es el paquete que se instala en las máquinas de los desarrolladores.

Compact:

Este paquete es aun más compacto que el Redistributable Package, es una versión para instalar en dispositivos que utilicen el sistema Windows Mobile (Pocket PC's y SmartPhones).

2.1 Class Library:

La biblioteca de clases de .Net esta organizada de manera jerárquica en paquetes llamados namespaces, según su funcionalidad presenta cuatro subdivisiones:

- La Base Class Library (BCL - Biblioteca de Clases Base):

  Alberga componentes de funcionalidad más general como las colecciones de datos, manejo de hileras, operaciones matemáticas y control de flujos de entrada y salida entre otros. Abarca la mayoría de los namespaces incluidos en el namespace raíz "System".

- ADO.NET:

  Contiene los componentes necesarios para la interacción con almacenes de datos persistentes ya sean bases de datos relacionales o documentos XML. Se ubica en los namespaces "System.Data" y "System.Xml".

- ASP.NET:

  Almacena los Componentes necesarios para el desarrollo de aplicaciones web. Se ubica en el namespace "System.Web".

- Windows Forms:

Almacena los Componentes necesarios para el desarrollo de aplicaciones de escritorio. Se ubica en el namespace "System.Windows.Forms".

2.2 Common Language Runtime:

La plataforma .NET presenta un modelo de ejecución de "maquina virtual" ya que el sistema operativo no es el encargado de la ejecución y el control de los programas. El Common Language Runtime (CLR) es un programa que corre sobre el sistema operativo y se encarga de controlar, ejecutar y brindar servicios a las aplicaciones .Net.

Las aplicaciones desarrolladas en .Net una vez compiladas producen archivos de extensiones .exe o .dll, sin embargo estos archivos no contienen código maquina para ser ejecutadas por el sistema operativo, sino mas bien un están en un lenguaje intermedio llamado MSIL (Microsoft Intermediate Language). EL MSIL es un lenguaje independiente de la arquitectura donde se compile la aplicación y tiene un formato que facilita su traducción a código maquina. A estos archivos se les conoce con el nombre de Assemblies (Ensamblados) y son los unicos archivos que reconose el CLR.

Entre los servicios que el CRL brinda a los programas que administra estan:

- Compilación Just In Time (o Justo A Tiempo):

  El CLR debe traducir las instrucciones al código maquina de la arquitectura sobre la que esta ejecutando, esta traducción no se da en un solo paso, si no que traduce bloques a medida que los va necesitando.

- Gestión Automática de Memoria:

  El CLR tiene un servicio llamado Garbage Collector (Recolector de Basura) que se encarga de liberar la memoria que no esta siendo utilizada por ningún programa, evitando que los programadores tengan que solicitar y liberar memoria de manera explicita, además elimina el uso de punteros.

- Gestión de Errores Consistente:

  Al no correr directamente sobre el sistema operativo los errores de un programa pueden ser atrapados por el CLR evitando que afecten otros programas en ejecución y manteniendo la estabilidad del sistema.

- Gestión de Seguridad:

  El CLR permite establecer lineamientos de seguridad que las programas que corran sobre le deben cumplir a cabalidad.

- Multithreading:

  El CLR pone a disposición de los programas que corre características multihilos y mecanismos de sincronización para acceso concurrente a datos.

Una característica de los assemblies es que además del código MSIL contienen una sección llamada "MetaData" en la cual se incluye información acerca de ellos y de los recursos externos que necesitan para su funcionamiento correcto. Esto permite facilidad de reutilización y manejo de versiones.

Los assemblies no necesitan estar registrados en la Registry de Windows, por lo tanto, para instalar una aplicación .NET basta copiar todos los assemblies necesarios a la computadora de destino, para desinstalar solamente es necesario borrarlos.

2.3 Common Language Specification (CLS):

La especificación de lenguaje común (CLS) es un estándar donde se especifican todas las características que soporta el CLR, la mayoría necesarias para implementar cualquier aplicación. Indistintamente del lenguaje de programación en que sean escritos, todos los componentes desarrollados y compilados de acuerdo al CLS pueden interactuar entre si. El .Net Framework incluye cuatro lenguajes de programación de alto nivel especificados de acuerdo al CLS:

- Microsoft Visual Basic .NET
- Microsoft Visual C# .NET
- Microsoft Visual J#.NET
- Microsoft Visual C++.NET

Esto permite por ejemplo que se pueda incrustar dentro de un programa escrito en Visual Basic.NET fragmentos escritos en C# o C++ .Net.

El estándar CLS esta a disposición publica, esto ha permitido que diseñadores de otros lenguajes y compiladores diseñen lenguajes compatibles con la especificación y por ende compatibles entre si y con los lenguajes de .Net.

Es importante destacar que la escogencia de un lenguaje especifico para el desarrollo de una aplicación en .Net se ha convertido en una cuestión que depende directamente de el gusto del desarrollador, los aspectos de eficiencia del compilador no son tomados en cuenta dado que todos los compiladores generan los assemblies de acuerdo al CLS y los asemblies son ejecutados por el CLR sin tomar en cuenta el lenguaje en que se ensamblaron.

3. Funcionamiento Interno del CLR

3.1 Common Language Infrastructure (CLI):

La infraestructura de lenguaje común (CLI) es un estándar desarrollado en conjunto por Microsoft y diversas compañías productoras de hardware donde se especifica un entorno virtual para la ejecución de aplicaciones. Esto para lograr que aplicaciones escritas en distintos lenguajes de programación puedan correr sobre distintas plataformas de hardware sin necesidad de recompilar su código.

Las ventajas de la arquitectura CLI son:

- Desarrollo componentes ínter operables indistintamente de la arquitectura del hardware y del lenguaje de programación en que se desarrollen.
- Uso de un sistema unificado de tipos de datos.
- Los componentes de software se encapsulan en unidades totalmente auto descriptivas y portables.
- Las aplicaciones son independientes entre si en tiempo de ejecución, pero son capaces de compartir recursos.

- Las dependencias entre componentes se resuelven en tiempo de ejecución tomando en cuenta la versión, atributos de localización y políticas administrativas.
- Las aplicaciones corren bajo la supervisión de un entorno privilegiado que controla y hace cumplir políticas en tiempo de ejecución.

3.2 Application Domain:

Todo entorno de ejecución (virtual o no) debe manejar alguna forma de aislamiento entre aplicaciones que asegure que los procesos de una aplicación no afecten negativamente a otra aplicación. En el CLR, todos los asemblies de un mismo programa se cargan dentro de un Application Domain (Frontera de Dominio) el cual se encarga de garantizar que los recursos de una aplicación no puedan ser afectados directamente por otra aplicación, además controla que los errores de una aplicación no afecte la ejecución del resto de aplicaciones.

3.3 Common Type System (CTS):

El sistema común de tipos de datos (CTS) brinda una definición común de los tipos de datos básicos que utiliza el CLR, además, define la forma en que se declaran, usan y manejan en tiempo de ejecución los tipos de datos. El CTS permite que los lenguajes que compilan para una plataforma CLI compartan el mismo sistema de tipos de datos, de esta manera facilita la interoperabilidad.

3.4 Manejo de la Memoria:

El CLR cuenta con dos segmentos de memoria para asignar a las aplicaciones:

El primer segmento es una pila (Last In – First Out) donde almacena tipos de datos primitivos (su referencia y su valor se encuentran en la misma posición de memoria) como enteros, caracteres y booleanos, también se almacenan aquí las referencias a datos no primitivos. La memoria utilizada por este tipo de datos se libera cuando finaliza el bloque de código donde se declararon.

Al otro segmento se le conoce como heap y es donde se guardan los valores de los datos referenciados. La memoria ocupada por estos datos es liberada automáticamente por el Garbage Collector del CLR.

4. Características del CLR

Presenta un modelo de programación orientado a objetos.

Ofrece un manejo de versiones de DLLs efectivo.

Las aplicaciones de .Net pueden correr bajo cualquier plataforma que tenga implementado el CLR.

Se puede desarrollar en cualquier lenguaje que tenga un compilador para .Net, no obliga al desarrollador a aprender un nuevo lenguaje de programación.

Cuenta con un recolector de basura que facilita al desarrollador la programación, le ahorra el trabajo de solicitar y liberar la memoria.

Facilita la detección de errores de conversión de tipos en tiempo de ejecución, porque revisa que los tipos de origen y destino sean compatibles.

Se encarga de que desde el código de una aplicación no se pueda acceder a localidades arbitrarias de memoria o al código o datos de otra aplicación.

Los errores que se producen durante la ejecución de una aplicación disparan excepciones.

Brinda soporte para aplicaciones multihilos.

Es capas de manejar objetos remotos y acceder a ellos tal y como si se encontrasen en la máquina local.

Proporciona mecanismos para restringir la ejecución o los permisos asignados a ciertos códigos basándose en su procedendecia o el usuario que los ejecute.

5. Ventajas y Desventajas de Microsoft .Net

5.1 Ventajas:

El CRL esta a cargo de administrar las aplicaciones y controlar que se lleven a cabo correctamente.

El desarrollador tiene libertad de escoger el lenguaje que le parezca más cómodo para trabajar.

El código nativo generado por el compilador JIT es especifico para cada arquitectura de hardware lo que produce un mayor rendimiento.

Se pueden asignar políticas de seguridad y permisos a una pieza de código, permitiendo así correr de manera segura aplicaciones de procedencia dudosa.

Gracias a su portabilidad los asemblies facilitan el mantenimiento y desarrollo de aplicaciones distribuidas.

5.2 Desventajas:

Al correr sobre una máquina virtual las aplicaciones.Net provocan un gasto de recursos mucho mayor al de una aplicación que corra de manera nativa,

El manejo de la memoria a través del recolector de basura consume gran cantidad de recursos y además le resta control al usuario sobre los datos de la aplicación.

6. Conclusiones

Gracias a la gran portabilidad y solides que ofrece la plataforma .Net, Microsoft a conseguido el apoyo de muchas empresas e instituciones que están ayudando al fortalecimiento de la misma, tal es el caso de Ximian/Novell que está trabajando para implementar la plataforma .Net para otros sistemas operativos como Linux, MacOS X, BDS y Solaris (Proyecto Mono), otros se dedican a crear lenguajes de programación para ampliar la plataforma que ya cuenta con más de una veintena, o a la creación de componentes, controles y bibliotecas de clases siendo algunos de estos de software libre.

7. Bibliografía

http://decsai.ugr.es/~cb/CSharp/dotnet/index.xml

http://jorgesaavedra.wordpress.com/2007/05/09/%C2%BFque-es-microsoftnet/

http://geneura.ugr.es/~jmerelo/ws/

http://es.wikipedia.org/wiki/.NET