Apéndice. Estructura *.NET*

El lanzamiento de la **plataforma *.NET*** fue anunciado oficialmente por primera vez por Microsoft en el año 2000. Pero, dos años después, el ***framework*** se lanzó como parte de ***Visual Studio .NET*.**

La importancia de cada tecnología que incluye la plataforma ha cambiado significativamente con eltiempo. En este artículo conocerás toda la evolución de *.NET* y sus características principales.

Qué es *.NET*

.NET es una plataforma de aplicaciones que permite la **creación y ejecución de servicios web y aplicaciones de Internet**. En la plataforma de desarrollo se pueden utilizar una serie de lenguajes,implementaciones, herramientas y bibliotecas para el desarrollo de las aplicaciones.

En definitiva, es hoy en día la plataforma de desarrollo de software más usada para nuevosproyectos de desarrollo de software además de Java.

Microsoft .NET es una colección de diferentes plataformas de software de Microsoft. El *framework* original fue desarrollado como una competencia directa a la plataforma Java. Así pues,los entornos de aplicación pueden ser desarrollados y ejecutados en base a *.NET*.

Hasta aproximadamente 2003, el término *.NET* sirvió a Microsoft como término de marketing ycomo palabra de moda para productos nuevos, pero muy diferentes, como sistemas operativos, servidores y software de oficina. Más tarde, el término se concentró en el desarrollo de software.

Inicialmente no tuvo mucho éxito, pero el entorno *.NET* ha cambiado significativamente a lo largo de los años y ha ganado en importancia. Hoy en día, el ***framework .NET*** se ha vuelto indispensableen la práctica diaria.

Componentes de la Arquitectura *.NET*

Los **componentes de la arquitectura *.NET*** juegan un papel importante en el desarrollo de aplicaciones. Los podemos clasificar en: el clásico ***.NET Framework***, que es un framework monolítico, el más actual **.NET Core *framework***, que es modular, la plataforma ***Xamarin*** y la específica de Windows ***UWP***. A continuación, los explico con mayor detenimiento para una mayor comprensión.

1. Implementaciones

.NET Framework está dividido en diferentes subcategorías y categorías de programas y, por lo tanto, contiene diferentes modelos de ejecución entre los que el usuario debe elegir al desarrollar el *software*. La base del desarrollo es la biblioteca de clases, que ha estado disponible en general comofuente compartida desde 2014. La llamada biblioteca de clases base permite el desarrollo de aplicaciones no sólo para entornos *Windows*, sino también para plataformas como *Android* o *MacOS*.

Esta plataforma de desarrollo se utiliza normalmente para crear aplicaciones de *Windows*, *Windowsmobile*, *Windows* S*erver*, etc con *Asp.net*, *WPF* y *Windows Forms.*

Por otro lado, ***.NET Core*** es una nueva alternativa que se separó por primera vez del *.NET Framework* en 2015. Debido a la mejora de la modularidad y a la portabilidad aún más sencilla del*software* a plataformas que no sean de *Microsoft*, *.NET Core* es particularmente apreciado por muchos desarrolladores.

Generalmente, desarrolla aplicaciones para la plataforma universal de ***Windows UWP*** y ***Asp.NetCore***.

Además, se utiliza para desarrollar aplicaciones en la nube. La biblioteca de clases***Core Ex*** es compatible con *Windows*, *MacOs* y *Linux*.

Otra implementación que destacar es la **plataforma *Xamarin*** en la que se pueden desarrollaraplicaciones para ***Android, iOS, tvOS, watchOS, macOS y Windows***. Ésta, disponía de herramientas y bibliotecas específicas.

Por último, incluir ***UWP,*** la plataforma universal de *Windows*. Aunque algunos desarrolladores lasitúan dentro de la plataforma *.NET Core* al compartir algunas bibliotecas de este, a *Microsoft* le gusta que se la considere una implementación más.

2. .NET Standard Library

Otro de los componentes que formaban parte de la arquitectura era la **biblioteca de clases portablePCL.** Con ella se podía compartir el código entre varios proyectos específicos de la plataforma, tanto en *IOS*, *Android*, *Windows* y *Windows* Phone.

Pero, las PCL presentaban muchas desventajas de compatibilidad entre implementaciones. Paraello, los desarrolladores crearon la ***API .NET Standard Library*.** Se trata de una fusión de las bibliotecas base y PCL compatible con todas las implementaciones.

En la actualidad, con la última versión de ***Visual Studio*** en 2017, las PCL quedaron obsoletas yborradas del sistema, así como las bibliotecas base de cada implementación. En su lugar fueronreemplazadas por ***.NET Standard Library***.

También, existen otras *API* suplementarias que son específicas de los sistemas operativos en los quese ejecuta.

3. Entorno en Tiempo de Ejecución

Una de las partes relevantes de la arquitectura es el **Entorno en Tiempo de Ejecución,** que comosu nombre indica es donde se ejecuta el programa administrado o el intervalo de tiempo en el queun software se ejecuta en un sistema operativo. Según la implementación utilizada:

- *.NET framework:* ***CLR (Common Language Runtime).***
- *.NET Core:* ***Core CLR (CoreCommon Language Runtime****).*
- *Xamarin*: entorno de implementación **Mono.**
- *UWP:* ***.NET Native.***

4. Infraestructura Común

Siguiendo con los componentes de la arquitectura, encontramos la **Infraestructura Común** dondeestán los **lenguajes de programación**: *C#, F#, VB* y el **motor de compilación *Ms Build*** para compilar los proyectos.

5. Herramientas de Desarrollo

Finalmente, otro componente son las **Herramientas de Desarrollo** para la creación de aplicacionesweb o móviles en los diferentes sistemas operativos mencionados:

- Administrador de paquetes para microsoft: Nuget
- Entorno de desarrollo integrado (*IDE*): *Visual Studio*, *Xamarin Studio*, *Visual Studio* para *Mac*, *JetBrains Rider*.
- Editores de Código: *Visual Studio Code* y *Plugin OmniSharp*.

Ilustración 1. Componentes de la Arquitectura .NET

Cómo funciona .*NET*

Para desarrollar un programa basado en el *.NET Framework* no se necesita mucho. Para escribir elprograma, en casos simples es suficiente un **editor de texto** y un **compilador**, que se incluye en el*framework*. Con estos dos componentes ya se pueden crear aplicaciones sencillas de *Windows*.

Para proyectos más grandes, *Microsoft* ofrece el entorno de desarrollo ***Visual Studio*** en diferentes ediciones. Además de un editor de código, contiene muchas herramientas útiles. Por ejemplo, para elanálisis, la solución de problemas o las pruebas. El **compilador**, por otro lado, es el responsable de traducir el código escrito del programa a un idioma que pueda ser leído por la computadora.

En las primeras versiones de *.NET*, este lenguaje intermedio generado se llamaba ***MSIL (Microsoft Intermediate Language)***. Debido a la estandarización, desde entonces ha sido renombrado como ***CIL (Common Intermediate Language)***.

El *CIL* permite a los programadores trabajar en el mismo proyecto en diferentes lenguajes de programación, ya que el código escrito del programa siempre se traduce al mismo lenguaje intermedio. Además, el *framework* permite a los programadores desarrollar partes de programas en*C++*, *C#* y *VB.NET* y luego combinarlos y usarlos juntos en un proyecto.

Mientras tanto, *Microsoft* también ofrece la posibilidad de desarrollar programas con ***HTML5*** y***JavaScript*** basados en el *.NET Framework*. Esta tecnología se puede encontrar, por ejemplo, en aplicaciones para *Windows 8.1* o *Windows Phone 8.1*.

Lenguajes de programación en .*NET*

Los dos principales lenguajes de programación en *.NET* son ***C#*** y ***Visual Basic .NET***. El lenguaje de programación ***F#*** también es muy utilizado. Desde 2013, ***JavaScript*** también está totalmente adherida y aceptada.

El ***C++*** no se ha usado durante mucho tiempo, pero también puede ser utilizado como una extensióndel lenguaje ***C++/CLI***. La extensión del lenguaje *C++/CLI* ofrece la posibilidad de transferir objetos directamente con otros lenguajes de objetos. El lenguaje de programación ***Python*** también se usa de forma regular.

Además, existen otros innumerables lenguajes de programación, que a menudo tienen un carácter más bien experimental y no son adecuados para productos comerciales.

Para que sirve .*NET*

El **objetivo de *.NET*** es crear una plataforma de desarrollo de software moderna, flexible y neutralpara el desarrollo de todo tipo de software. De esta forma, proporciona a los consumidores y a los programadores de las empresas una interfaz de usuario perfectamente interoperable. Esta circunstancia ocasiona que la informática esté cada vez más orientada a los navegadores.

Con la excepción de la programación de controladores de hardware, el *.NET* abarca todo tipo deaplicaciones, desde aplicaciones de escritorio hasta aplicaciones web; desde servicios de sistemahasta servicios web; y desde rutinas de bases de datos hasta programación de aplicaciones *IoT* y*Machine Learning*.

Microsoft diseñó esta plataforma para ser un sistema neutral desde el principio, pero no hizo ningúnesfuerzo para implementarlo en *Mac* y *Unix/Linux*. Debido a la iniciativa de otras compañías (especialmente *Novell*), grandes partes de *.NET* están ahora disponibles para otros sistemas operativos.

Y esto es incluso apoyado por *Microsoft* en vista del creciente número de sistemas operativos quecompiten entre sí, especialmente en el mercado de los dispositivos móviles. El propio *Microsoft* ofrece una variante para *Mac OS* con *Silverlight*.

Ventajas de *.NET*

Básicamente, las diferentes variantes de los entornos de desarrollo *.NET* ofrecen a losprogramadores toda una gama de ventajas. Estas son, entre otras:

- **Interoperabilidad**: los elementos de *software* y programas desarrollados pueden utilizar las funcionalidades de los programas desarrollados fuera de *.NET*.
- ***Common Language Runtime (CLR)*:** un entorno de tiempo en ejecución uniforme de todos los lenguajes de programación *.NET* disponibles. Esto asegura un comportamientoconsistente en las áreas de uso de memoria y seguridad.
- **Independencia del lenguaje utilizado:** La base es una arquitectura de lenguaje común, que permite el intercambio de datos entre dos programas en diferentes lenguajes de programación.
- **Una biblioteca de clases comunes:** Una biblioteca de códigos para las funciones másutilizadas para evitar la duplicación y la programación innecesaria.
- **Seguridad:** Todas las soluciones de software desarrolladas se basan en un modelo deseguridad común y efectivo.