10

Fecha de presentación: diciembre, 2021
Fecha de aceptación: enero, 2022
Fecha de publicación: marzo, 2022

DESARROLLO
DE SOFTWARE CON NET CORE
SOFTWARE DEVELOPMENT WITH NET CORE

Luis Antonio Llerena Ocaña[1]
E-mail: ua.luisllerena@uniandes.edu.ec
ORCID: https://orcid.org/0000-0001-6440-0167
Fausto Alberto Viscaino Naranjo[1]
E-mail: ua.faustoviscaino@uniandes.edu.ec
ORCID: https://orcid.org/0000-0003-1760-6992
Walter Vinicio Culque Toapanta[1]
E-mail: ua.walterculque@uniandes.edu.ec
ORCID: https://orcid.org/0000-0002-8086-4209
[1] Universidad Regional Autónoma de Los Andes. Ecuador.

Cita sugerida (APA, séptima edición)

Llerena Ocaña, L. A., Viscaino Naranjo, F. A., & Culque Toapanta, W. V. (2022). Desarrollo de software con Net Core. *Revista Universidad y Sociedad*, 14(2), 85-89.

RESUMEN

Desde el inicio de la informática, se puede considerar que existen dos bandos, el software libre y el software propietario, este artículo se enfoca en el mundo que Microsoft está ingresando y de una manera muy fuerte. El framework de desarrollo Net Core es una de las mejores armas que actualmente posee para el desarrollo de software y este se puede adaptar a cualquier tipo de sistema operativo haciéndolo multiplataforma, lo que facilita el desarrollo por su propuesta que, contiene 3 métodos para desarrollo Model First, Code First o Database First. En todo sentido, las metricas aplicadas para el desarrollo de aplicaciones web o API's genera mucha confianza gracias a su fácil uso y sobre todo a que usa Linq como medio para consultas de datos estructuradas, lo que reduce los ataques por inyecciones de código. Mediante un experimento realizado con base en la aceptación de los estudiantes de octavo semestre de la carrera de Ingeniería de Software, se midió tiempo de desarrollo, complejidad para el desarrollo, tiempo de respuesta en consumo de datos, todo esto trabajando con una metodología basada en problemas para obtener los mejores resultados.

Palabras clave: Net Core, Desarrollo Web, frameworks.

ABSTRACT

Since the very beginning of computing, there has been two well-defined sides, free software and proprietary software; this article focuses on the whole world of impact caused by Microsoft, and in a very efficient way. The Net Core development framework is one of the best tools currently available for software development, and this can be adapted to any type of operating system making it multiplatform, which facilitates development by its proposal that contains 3 methods for development: Model First, Code First or Database First. In any sense, the metrics applied for the development of web applications or API's generates a lot of confidence thanks to its ease of use and especially because it uses Linq as a means for structured data queries, which reduces attacks by code injections. In an experiment carried out based on the acceptance of eighth semester students of Software Engineering, the following parameters were measured: development time, complexity for development, and response time in data consumption; all of these were analyzed using a problem-based methodology in order to obtain the best results.

Keywords: Net Core, Web Development, frameworks.

INTRODUCCIÓN

La plataforma .NET de Microsoft y el lenguaje de programación C# fueron aceptados formalmente alrededor de 2002 y se han convertido rápidamente en un pilar del desarrollo de software moderno. La plataforma .NET permite un gran número de lenguajes de programación (incluidos C #, VB.NET y F #) para interactuar entre sí. Un programa escrito en C# puede ser referenciado por otro programa escrito en VB.NET (Meyer, 2001).

El 27 de junio de 2016, Microsoft lanzó oficialmente .NET Core. Como .NET, .NET Core permite que los lenguajes interoperabilidad entre sí (aunque se admite un número limitado de idiomas). Más importante aún, esto el nuevo framework ya no se limita a ejecutarse en el sistema operativo Windows, sino que también puede ejecutarse (y ser desarrollado) en iOS y Linux. El 23 de septiembre de 2019, Microsoft lanzó .NET Core 3.0 y C# 8 todas las versiones anteriores de C# se ejecutan en el marco .NET original (depende de la versión, por supuesto), C# 8 solo se ejecutará en .NET Core 3.0 y superior. Esta versión de C# introduce muchas características nuevas que crearían rupturas cambios en versiones anteriores de .NET Framework (así como .NET Core). Junto con el hecho de que muchos de las versiones anteriores de .NET Framework están integradas en varios sistemas operativos de Windows, no era posible que el framework .NET se actualizara para admitir C# 8 (Troelsen, 2001; Griffin, 2017).

Algunos beneficios clave de la plataforma .NET Core es que es una plataforma de software para crear aplicaciones web y sistemas de servicios en los sistemas operativos Windows, iOS y Linux, así como las aplicaciones WinForms y WPF en Windows.

Interoperabilidad con código existente: esto es (por supuesto) algo bueno .NET existente el framework puede operar con el software .NET Core más nuevo y viceversa a través de .NET Standard.

Soporte para numerosos lenguajes de programación: las aplicaciones .NET Core se pueden creado con los lenguajes de programación C #, F # y VB.NET (siendo C # el enfoque principal para ASP.NET Core).

Un motor de tiempo de ejecución común compartido por todos los lenguajes .NET Core: un aspecto de esto es un conjunto bien definido de tipos que comprende cada lenguaje .NET Core. Integración de idiomas: .NET Core admite la herencia entre idiomas, el manejo de excepciones entre idiomas y la depuración de código entre idiomas. Por ejemplo, puede definir una clase base en C # y extender este tipo en Visual Basic.

Una biblioteca de clases base completa: esta biblioteca proporciona miles de tipos que le permiten crear bibliotecas de código, aplicaciones de terminal simples, gráficos aplicaciones de escritorio y sitios web de nivel empresarial (Hummel, et al., 1993; Kirk, et al., 1998; Freeman, 2016; Brown, et al., 2019).

Un modelo de implementación simplificado: las bibliotecas .NET Core no están registradas en el registro del sistema. Además, la plataforma .NET Core permite múltiples versiones del framework, así como aplicaciones para existir en armonía en una sola máquina.

Amplia compatibilidad con la línea de comandos: .NET Core Command Line Interface (CLI) es una Cadena de herramientas multiplataforma para desarrollar y empaquetar aplicaciones .NET Core (Heino, et al., 2009; Price, 2019).

Se pueden instalar herramientas adicionales (global o localmente) más allá de las herramientas estándar que enviar con .NET Core SDK.

Verá cada uno de estos temas (y muchos más) examinados en los capítulos siguientes. Pero primero, necesito explique el nuevo ciclo de vida de soporte para .NET Core.

La información recopilada dentro del experimento realizado oferta una comparativa entre el desarrollo web enfocados al uso de .Net Framework y .Net Core para tener una comparativa clara entre las tecnologías de Microsoft (Brown, 1995).

Se debe aclara que Microsoft ha creado varios lenguajes de programación entre ellos TypeScript, el cual es usado en muchos frameworks relacionados con JavaScript para la web (Llerena Ocaña, et al., 2021).

MATERIALES Y MÉTODOS

Dentro del experimento realizado, se buscó tratar la naturaleza abstracta propia sobre el desarrollo debido a que, el aprendizaje para desarrollar un sistema o aplicación web es muy basto, ya que tiende a llevar un determinado proceso que provee una metodología de desarrollo de software (Llerena Ocaña & González Hernández, 2020).

El diseño de una aplicación puede generar varios conflictos con bases de datos o el mismo lenguaje de programación, al generar API's para consumo de aplicaciones externas, si hablamos por el uso de Angular, ReactJs, Vue, Ionic o si se lo realiza por el consumo de aplicaciones de escritorio o por la misma web con el uso de html, css y javascript (Llerena Ocaña & González Hernández, 2017).

Los participantes son del octavo nivel de la Carrera de Ingeniería de Software consta con 7 alumnos, se uso el total de los estudiantes y de este modo se tomo el "caso típico" que menciona (Cunningham, et al., 2020), para el desarrollo al estudiante se le dio a elegir el sistema operativo que desee, así como el editor de código. El rol del docente se toma el de Gold (1969), citado en Llerena Ocaña & González Hernández (2020): "observador-como-participante", ya que existieron conflictos a nivel de aprendizaje o desarrollo y se puede dar solución a cada uno de estos.

Este estudio se realizó dentro del periodo de ordinario de clases en el periodo académico desarrollo en el año 2021. En cuanto a la comparación entre varios componentes propios de los frameworks (.net y .net core) iniciamos con el ORM el cual es un modelo de programación que permite mapear las estructuras de una base de datos relacional (SQL Server, Oracle, MySQL, etc.), en adelante RDBMS (Relational Database Management System), sobre una estructura lógica de entidades con el objeto de simplificar y acelerar el desarrollo de nuestras aplicaciones (Reeves & Nass, 2016).

Otro de los datos evaluados es el modelo o patrón de desarrollo, el cual se enfocó propiamente al estudiante según su afinidad y gusto por el desarrollo, por ejemplo, la mayoría de los estudiantes usaron un patrón MVC (Modelo – Vista – Controlador) y el uso de formularios web; dos estudiantes usaron API's y para la interfaz de usuario se uso Angular.

RESULTADOS Y DISCUSIÓN

En las comparativas los ORM trabajan de manera similar con el motor de base de datos propio de Microsoft, en la tabla 1 se muestra su efectividad con varias bases de datos demostrando su efectividad, cabe tener en cuenta que, la empresa dueña de la tecnología da opción para que otras aporten para el desarrollo, una vez que se valida y se publica, estos aportes pueden acceder, muchos de estos son libres otros con mayor funcionalidad son de paga.

En todos los casos, se ajusto que las líneas de código a ejecutar para cada base de datos de prueba, se debe tener en cuenta que para Sql Server la librería es nativa, para Mysql y MariaDb se usó la librería de Pomelo y para Oracle las librerías propias de Oracle.

En una comparativa clara, Sql Server junto a MariaDb tienen los mismos tiempos de respuesta promedio, como consecuencia a esta investigación fue que no se tuvo en cuenta los componentes físicos de las maquinas de los sujetos de prueba.

Con 325 filas de código en donde se detalla la creación de 4 tablas relacionadas entre sí con sus respectivos registros, se muestra a detalle los tiempos de respuesta promedio (Tabla 1).

Tabla 1. Tiempo de respuesta con base de datos.

Base de Datos	Registros	Net Framework	Net Core
Sql Server	325	1.30 ms	1.27 ms
Mysql	325	1.50 ms	1.50 ms
Oracle	325	1.34 ms	1.30 ms
MariaDB	325	1.30 ms	1.27 ms

Con la base de datos lista, los sujetos de prueba escogieron desarrollar el problema según el patrón de desarrollo más acepta por ellos o en su defecto con el que estén más entrelazados, y el patrón MVC (Modelo – Vista - Controlador) fue es más aceptado, por las bondades que en cada uno de los frameworks ofrece (Tabla 2).

Tabla 2. Patrón de Desarrollo de Software.

Patrón de Desarrollo	Net Framework	Net Core
Web Forms	1	0
MVC	2	2
API's	0	2

Por razones de aprendizaje, se excluyeron muchas partes del proceso normal que rige una metodología de desarrollo de software, por tal motivo, con el conocimiento de la base de datos se procedió a resolver las operaciones básicas CRUD para medir los tiempos de desarrollo en cada una de las tecnologías que los sujetos de prueba eligieron.

Si hablamos con formularios web, en donde todos los procesos son escritos por el programador, el tiempo de desarrollo es muy lento; al realizar una comparativa en tiempo de desarrollo es imperativo mostrar a los estudiantes para que sirve cada línea de código, la ventaja de este proceso es que el conocimiento que adquiere este, podrá ocuparlo en cualquier lenguaje de programación, además en este tipo de desarrollo, se usa un lenguaje estructurado de consultas (SQL).

Cundo se utiliza un framework, los tiempos se reducen notablemente, al usar el modelo MVC con Entity Framework de .Net, el trabajo baja a 10 minutos, esto debido a que el proceso se complica cuando debemos usar las restricciones para mostrar en la vista.

Entity Framework Core reduce el trabajo aún más, las anotaciones de datos que se utilizan para controlar las

validaciones son relativamente más fáciles al igual que el código que se debe escribir (Tabla 3).

Tabla 3. Tiempo de Desarrollo por tabla de Base de datos.

Patrón de Desarrollo	Net Framework	Net Core
Web Forms	30 minutos	0
MVC	10 minutos	3 minutos
API's	0	3 minutos

En la tabla #4 se muestran los mismos valores de la tabla anterior, esto hace relación debido a que los frameworks con el patrón MVC generan la vista al igual que los controladores, sucede lo mismo con la API, pero esto difiere con un desarrollo mediante formulario web ya que, como se explicó anteriormente, el proceso se desarrolló basándonos en el proceso que se necesita.

Aquí se tomó en cuenta el desarrollo de la interfaz para el consumo de la API mediante Angular, si tomamos el tiempo que se menciona en la tabla 3 por el desarrollo con. Net Core suman 18 minutos por tabla de base de datos, el uso de componentes y servicios para el consumo de datos mediante Json resulta aún más productivo para el desarrollo (Tabla 4).

Tabla 4. Tiempo de Desarrollo en Interfaz Gráfica de Usuario.

Patrón de Desarrollo	Net Framework	Net Core
Web Forms	30 minutos	0
MVC	10 minutos	3 minutos
Angular	0	18 minutos

Los medios de validación por parte de Angular y de los formularios web deben ser escritos lo que conlleva un tiempo adicional para el desarrollo, a diferencia del marco de desarrollo o framework, estos son definidos directamente en el modelo, este se traduce a que no hay necesidad de preocuparse por las restricciones o validaciones de datos (Tabla 5).

Tabla 5. Tiempo por validación.

Patrón de Desarrollo	Net Framework	Net Core
Web Forms	10 minutos	0
MVC	Se define en el modelo	Se define en el modelo
Angular	0	15 minutos

En los primeros días de .NET, al comienzo del nuevo milenio, las aplicaciones de escritorio eran el estándar en la industria. Fue un tiempo antes de que las aplicaciones de navegador dieran forma a la industria del desarrollo de software.

En aquel entonces, las aplicaciones web eran escasas y en su mayoría estaban compuestas por contenido textual. Sin embargo, lo que los distinguió de las aplicaciones de escritorio fue la disponibilidad. Allí no era necesario realizar instalaciones complejas y podían funcionar en cualquier dispositivo que tuviera un navegador web debido a su disponibilidad, la complejidad y la prevalencia de las aplicaciones web aumentan constantemente. Desde su primera versión, los muchos autores han creado soporte para el desarrollo de aplicaciones web en .NET como parte integral de la biblioteca de clases del framework.

La herramienta de desarrollo de aplicaciones web se llamó ASP.NET Web Forms, en línea con la herramienta de desarrollo de aplicaciones de escritorio. Los desarrolladores podrían rápidamente, utilizando controles similares a los de un escritorio ya definidos, implemente sitios web. Sin embargo, la naturaleza de los formularios web, que fue la base del rápido desarrollo, ha sido cada vez más el objetivo de crítica. Es decir, el desarrollo funcionó bien siempre que no haya necesidad de más ajuste de los elementos de la interfaz de usuario existentes. Al mismo tiempo, los frameworks de otros lenguajes de programación dieron a los desarrolladores un control total sobre el código del cliente y los lenguajes del cliente se volvieron más predominante. Junto con Web Forms, Microsoft ha desarrollado otro framework orientado a la web llamado Windows Communication Foundation (WCF), diseñado utilizando una arquitectura orientada a servicios principios para apoyar la computación distribuida.

Los formularios web fueron la herramienta de Microsoft para el desarrollo de aplicaciones web hasta 2009, cuando se lanzó la primera versión del framework ASP.NET MVC. Fue lanzado como una solución al problema de formularios web cerrados, diseñados utilizando principios de software estándar para aplicaciones web desarrollo en ese momento.

Con patrones de desarrollo fuertemente estructurados, aparecen renovadas maneras de programar, MVC marca tendencia dentro del desarrollo de software, así como los frameworks que trabajan sobre este tipo de desarrollo. Net Core presenta una salida fresca al código propietario orientado por Microsoft, dando una salida más cómoda para el desarrollo; Net Core sigue este patrón de desarrollo para acoplarse, teniendo una gran variedad de consumo de datos mediante aplicaciones web, API's o microservicios.

CONCLUSIONES

En la literatura investigada se detalla visiblemente los pasos a seguir sobre el proceso de desarrollo de software según la metodología que se elija, pero los tiempos se reducen cuando se usa un framework el cual agilita el manejo de la base de datos.

Al comparar el backend de net core con php existe una considerable reducción de tiempo como se puede verificar en los resultados del experimento.

El controlar el frontend con un patrón MVC que el framework lo resuelve, como el framework esta desarrollado en ingles, cuando se escribe el código se debe usar anotaciones de datos para evitar que el texto de error, aparezcan en inglés.

REFERENCIAS BIBLIOGRÁFICAS

Brown, J. S., Collins, A., & Duguid, P. (2019). Situated cognition and the culture of learning. Educational researcher, 18(1), 32-42.

Brown, W. H. (1995). Trends in patent renewals at the United States Patent and Trademark Office. ***World Patent Information***, ***17***(4), 225-234.

Cunningham, K., Bejarano, R. A., Guzdial, M., & Ericson, B. (2020). I'm Not a computer: How Identity Informs Value and Expectancy During a Programming Activity. International Society of the learning sciences.

Freeman, A. (2016). Pro Asp. net core MVC. Apress.

Griffin, M. L. (2017). Using critical incidents to promote and assess reflective thinking in preservice teachers. Reflective practice, 4(2), 207-220.

Heino, N., Dietzold, S., Martin, M., & Auer, S. (2009). Developing semantic web applications with the ontowiki framework. En, T., Pellegrini, S., Auer, K., Tochtermann, y S. Schaffert, ***Networked Knowledge-Networked Media***. (pp. 61-77). Heidelberg.

Hummel, R. E., Morrone, A., Ludwig, M., & Chang, S. S. (1993). On the origin of photoluminescence in spark-eroded (porous) silicon. Applied physics letters, 63(20), 2771-2773.

Kirk, D., & Macdonald, D. (1998). Situated learning in physical education. Journal of Teaching in Physical education, 17(3), 376-387.

Llerena Ocaña, L. A., & González Hernández, W. (2017). La competencia desarrollar sistemas web en la formación de los profesionales informáticos: una aproximación a su estudio. Reidocrea. 19(1), 229-245.

Llerena Ocaña, L. A., & González Hernández, W. (2020). Formación de la competencia «desarrollar sistemas web en los espacios virtuales de aprendizaje. Revista Cubana de Educación Superior, 39(1), 1-16.

Llerena Ocaña, L. A., Fernández Villacres, G. E., Viscaino Naranjo, F. A., & Baño Naranjo, F. P. (2021). Typescript-based frameworks for the development of interactive web applications. Dilemas contemporáneos: educación, política y valores, 8(3), 1-15.

Meyer, B. (2001) .NET is coming [Microsoft Web services platform]. ***Computer***, ***34***(8), 92-97.

Price, M. J. (2019). ***C# 8.0 and. NET Core 3.0–Modern Cross-Platform Development: Build applications with C#,. NET Core, Entity Framework Core, ASP. NET Core, and ML. NET using Visual Studio Code***. Packt Publishing Ltd.

Reeves, B., & Nass, C. I. (2016). The media equation: How people treat computers, television, and new media like real people and places. Cambridge University Press.

Troelsen, A. (2001). ***C# and the. NET Platform***. Apress.