Escuela Politécnica Superior

Universidad de Córdoba

Microsoft y Software Libre

Software Libre y Compromiso Social

Eduardo Arroyo Ramírez
i12arrae@uco.es
Curso 2019/2020

Índice

1. Introducción

Microsoft ha sido históricamente enemiga acérrima del software libre, pero desde que Satya Nadella fuera nombrado CEO de Microsoft en 2014, la compañía ha dado un giro de 180º a su postura al respecto liberando software, cediendo patentes y entrando en el negocio del software libre con Azure.

Este trabajo repasa el origen y la motivación del cambio de paradigma de la compañía y las repercusiones que dicho cambio ha tenido. También describe algunos de los productos que Microsoft ha liberado, como .NET Framework, Visual Studio Code o Xamarin.Forms y cómo ahora están soportados por plataformas como Linux o MacOS. Por último presenta un pequeño programa que ha sido desarrollado con tecnologías libres de Microsoft y con SQL Server utilizando GNU/Linux.

2. El estilo Microsoft

Microsoft es una multinacional fundada el 4 de abril de 1975 por Bill Gates y Paul Allen en Albuquerque, Nuevo México. Desde sus inicios, la compañía se ha caracterizado por su cuestionable ética y su agresiva estrategia empresarial, así como por el uso de su posición dominante para «exterminar» a la competencia e imponer sus productos y sus estándares. Prueba de ello son los numerosos juicios en los que la compañía se ha visto envuelta[1]. Con el fin de formar una imagen de la compañía, se resumen a continuación algunos hechos que tuvieron a Microsoft como protagonista.

2.1. IBM y Microsoft

A principios de los 80 IBM dominaba cómodamente el mercado de la computación para empresas. Con la llegada de los ordenadores personales, la compañía decidió contratar los servicios de Microsoft para el desarrollo de su sistema operativo para sus IBM-PC. Para ello, Microsoft compró los derechos no-exclusivos del sistema operativo QDOS, en el cuál basaría el desarrollo del nuevo sistema, a la empresa SCP. Microsoft desarrolló MS-DOS y lo licenció a IBM como PC-DOS. El único componente exclusivo para IBM era la BIOS.[2][3]

Supuestamente, el apretado calendario del nuevo sistema operativo llevó a Microsoft a utilizar malas prácticas en el desarrollo. Esto tuvo como consecuencia un bajo rendimiento de las llamadas al sistema a través de las funciones estándar de la BIOS en comparación con los accesos al sistema saltándose la BIOS. Este hecho llevó a muchos desarrolladores de videojuegos (por ejemplo, Microsoft Flight Simulator 1.0, 1982) a ignorar la BIOS, lo que revelaba la arquitectura del sistema en la programación, permitiendo a otros fabricantes de hardware obtener pistas sobre la arquitectura del IBM-PC rápidamente con el fin de clonarla. Ya en 1982 Compaq lanzó el Compaq Portable, el primer ordenador PC-Compatible. La rápida expansión del mercado de PCs IBM-Compatibes llevó a Microsoft, que inteligentemente había incluído en su contrato con IBM una cláusula que les permitía vender su sistema operativo MS-DOS a clientes diferentes de IBM, a convertirse en la compañía líder en software de ordenadores personales, desplazando a IBM de su hegemonía total.[4]

2.2. La guerra de los navegadores

Durante los años 90 tuvo lugar la conocida como «Guerra de Navegadores»[5] en la que Microsoft logró imponer su producto Internet Explorer distribuyéndolo gratuitamente junto con Windows 95, lo que asfixió comercialmente a su principal competidor, Netscape Communicator, que contaba con un producto mejor. Aunque Microsoft fue condenado en un juicio por monopolio[6], tras las apelaciones y el acuerdo final, las medidas adoptadas finalmente no obligaban a la compañía a realizar cambios efectivos en su estructura o su política comercial.[7]

Años después tuvo lugar un juicio similar en la Unión Europea[8] por abuso de posición dominante en mercado a raíz de protestas de Novell y SUN Microsystems. Este juicio acabó obligando a Microsoft a que divulgara información sobre algunos de sus productos y liberara una versión de Windows sin Windows Media Player

2.3. Embrace, extend and extinguish

El departamento de justicia de los estados unidos descubrió que Microsoft usaba la expresión «embrace, extend and extinguish» internamente para describir su estrategia por la cuál Microsoft decide en un primer momento «adoptar» un estándar público, después lo «mejora» añadiendo extensiones incompatibles con el estándar original, y termina «extinguiendo» el estándar público al imponer sus propias extensiones propietarias por medio de su dominio del mercado.[9]

Otra estrategia de dudosa ética utilizada por Microsoft fue la llamada «FUD» por sus siglas en inglés «Fear, Uncertainty and Doubt»[10]. En el primero de los Halloween Documents[11], se revela que Microsoft utilizaba normalmente dicho método para desacreditar públicamente los productos de la competencia.

3. Relación de Microsoft con el Software Libre

Durante los 90 Microsoft estuvo en la cima del mercado del software para ordenadores personales gracias a las estrategias de marketing de la compañía pero al final de la década, la compañía comenzó a percibir el movimiento de Software Libre y Open Source como una amenaza a sus beneficios. La década siguiente estuvo marcada por los ataques constantes de Microsoft al Software Libre.[12]

3.1. El juego sucio

3.1.1. The Halloween Documents

Se conoce como «The Halloween Documents» una serie de documentos confidenciales de Microsoft sobre potenciales estrategias relacionadas con el software libre/open-source y sobre Linux en particular, y una serie de respuestas de diversos medios a estos documentos.

Los primeros fueron publicados por Eric S. Raymond el 30 de octubre de 1998 y Microsoft ha reconocido su autenticidad. Los dos primeros documentos reconocen que los programas libres/abiertos constituyen una amenaza significativa al dominio de Microsoft, que dichos programas son más competitivos que los suyos y sugieren medios para interferir en su desarrollo. Se trata de una información muy importante ya que contradice varias declaraciones publicas de la compañía. [11]

3.1.2. Las declaraciones de Steve Ballmer

Después de Bill Gates, Steve Ballmer fue CEO de Microsoft de 2000 a 2014. Entre sus declaraciones públicas, Ballmer dejó auténticas joyas, algunas de ellas referentes al software libre, en las que se puede ver un claro ejemplo de la estrategia «FUD» mencionada anteriormente (ver 2.3).

En 2000, Ballmer afirmó que «Linux tiene las características del comunismo que la gente tanto ama. Es decir, que es gratis» [13]. Más tarde, en 2001 diría «Linux es un cáncer que se pega, en sentido de la propiedad intelectual, a todo lo que toca». [14]

En 2016, habiendo tomado ya Nadella las riendas de la compañía, Ballmer rectificó su opinión sobre GNU/Linux y reconoció que es un enemigo real de Windows. También escribió un email a Satya Nadella para felicitarlo por las recientes decisiones de la compañía respecto a las nuevas políticas de la compañía respecto al Software Libre.[15]

3.1.3. 2014: Balmer contra LiMux

En 2003, la ciudad de Munich estaba apunto de aprobar una migración de los sistemas del ayuntamiento (aproximadamente 15000 equipos) a Linux cuando Steve Ballmer interrumpió sus vacaciones y voló hasta la ciudad para reunirse con su alcalde, Christian Ude, con el fin de persuadirlo para detener la migración. Ballmer trató de convencer al alcande de que sería una mala decisión cambiar a sistemas de código abierto porque no era «algo en lo que la administración pudiera confiar». La migración se completó antes de 2008, ahorrando a la ciudad 11 millones de euros y reduciendo su dependencia de estándares propietarios y fabricantes.[16]

3.2. La etapa de Satya Nadella

En 2014 Satya Nadella fue nombrado CEO de Microsoft. Desde entonces, la compañía comenzó a adoptar el Software Libre/Open Source en sus principales áreas de negocio. En contraste con la etapa de Ballmer, Nadella presentó una imagen que rezaba “Microsoft ♡ Linux”. Inicio 4/2/2014 Cambio de paradigma: cloud computing, software como servicio: el software de servidor es territorio libre.

3.2.1. Azure

3.2.2. 2014: Microsoft abre .NET y lo lleva a Linux y OS

Microsoft abre .NET y lo lleva a Linux y OS

3.2.3. 2015: Microsoft se une a la Linux Foundation

Microsoft —yes, Microsoft— joins the Linux Foundation

3.2.4. 2016: Windows Subsystem for Linux

3.2.5. 2016: Xamarin

Compra de Xamarin
Hacen abierto el SDK de Xamarin

3.2.6. 2018: Compra de Github

Compra de GitHub 1
Compra de GitHub 2
Compra de GitHub 3

3.2.7. 2018: Open Invention Network

Hacer hincapié en las patentes que aporta, relación con la guerra de patentes en el pasado. Microsoft se une a la Open Invention Network 1
Microsoft se une a la Open Invention Network 2
Microsoft se une a la Open Invention Network 3

3.2.8. 2019: Charla de Richard Stallman en Microsoft

Charla de Stallman en Microsoft

3.2.9. 2019: Petición de liberar Windows 7

De llegar a liberar windows 7, quizás les interesaría utilizar una licencia copyleft para que nadie les hiciera competencia vendiendo su producto como no-libre.

3.3. Motivos para el cambio

En la década de los 2000, Microsoft se encontraba cómodamente en la cima del mercado del software para ordenadores personales, pero la expansión del Smartphone en la década de 2010 junto con la aparición del cloud computing, forzaron a la compañía de Redmond a replantear su estrategia.

Microsoft tenía copado el mercado de escritorio con Windows, pero al descender las ventas de PC y aumentar el uso de software como servicio tuvieron que cambiar su estrategia.

4. Licencias de Microsoft

Microsoft cuenta con varias licencias libres reconocidas por la FLS y por la OSS. A pesar de ello, la mayoría de los proyectos OSS de Microsoft utilizan licencias Apache 2.0 o MIT.

4.1. Microsoft Public License (MS-PL)

La Licencia Púbica de Microsoft (MS-PL) es la menos restrictiva de las licencias de Microsoft y permite la distribución de código compilado ya sea para fines comerciales como no comerciales bajo cualquier licencia que cumpla con la MS-PL. La redistribución del código fuente en sí únicamente se autoriza bajo la MS-PL. Inicialmente titulada Microsoft Permissive License, fue renombrada a Microsoft Public License, mientras que se estaba revisando para su aprobación por la Open Source Initiative (OSI). La licencia fue aprobada en 2007 junto con el MS-RL. De acuerdo con la Free Software Foundation, es una licencia de software libre. Sin embargo, no es compatible con la GNU GPL.[17]

4.2. Microsoft Reciprocal License (MS-RL)

La Licencia Recíproca de Microsoft permite la distribución de código derivado siempre y cuando los archivos fuentes estén incluidos y mantengan la licencia MS-RL. La MS-RL permite que aquellos archivos en la distribución que no contengan código originalmente licenciado bajo la MS-RL sean licenciados de acuerdo a la elección del titular de los derechos de autor. Esto es equivalente a la CDDL, la EPL o la LGPL. En un principio conocida como la Licencia Comunitaria de Microsoft, fue renombrada en el proceso de aprobación de OSI. En 2007 se anunció que Microsoft había presentado oficialmente la Ms-PL y la Ms-RL a OSI para su aprobación. De acuerdo con la Free Software Foundation, es una licencia de software libre. Sin embargo, no es compatible con la GNU GPL.[18]

4.3. Licencias privativas, cerradas y comerciales

4.3.1. Microsoft Reference source License (MS-RSL)

Esta es la más restrictiva de las licencias de código compartido de Microsoft. El código fuente está disponible sólo para verse con fines de referencia, principalmente para poder ver las clases de código fuente de Microsoft durante la depuración. Los desarrolladores no pueden distribuir o modificar el código para fines comerciales o no comerciales. La licencia ha sido anteriormente abreviada Ms-RL, pero Ms-RL ahora se refiere a la Licencia Recíproca de Microsoft.[19]

4.3.2. Microsoft Limited Public License (MS-LPL)

Esta es una versión de la licencia pública de Microsoft en la que los derechos sólo se conceden a los desarrolladores de software basado en Microsoft Windows. Esta licencia no es de código abierto, tal como se define por la OSI, ya que viola la condición de que las licencias de código abierto debe ser tecnológicamente neutrales.[20]

4.3.3. Microsoft Limited Reciprocal License (MS-LRL)

Esta es la versión de la Licencia Recíproca de Microsoft en la que los derechos sólo se conceden cuando se desarrolla software para una plataforma Microsoft Windows. Al igual que la Ms-LPL, esta licencia no es de código abierto porque no es tecnológicamente neutral.[21]

5. Productos de Software Libre de Microsoft

Microsoft ha desarrollado numerosos proyectos Open Source que se pueden consultar en Microsoft Open Source. A continuación se exponen algunos de los proyectos OSS de Microsoft más significativos:

5.1. .NET Core

.NET Core es una plataforma de desarrollo de código abierto para uso general. Se pueden crear aplicaciones de .NET Core para Windows, macOS y Linux para procesadores x64, x86, ARM32 y ARM64 mediante los lenguajes C#, Visual Basic y F#. Se proporcionan marcos y API para la nube, IoT, la Interfaz de usuario de cliente y el aprendizaje automático.

La primera versión 1.0 apareció en junio de 2016, .NET Core 2.0 en agosto de 2017 y la última versión mayor, .NET Core 3.0, en mayo de 2019. Esta versión incluye soporte para aplicaciones de escritorio (sólo en entorno Windows), inteligencia artificial, aprendizaje automático e IoT.

.NET Core es un proyecto Open Source de la .NET Foundation. Se puede participar registrando incidencias y preguntas en la comunidad de desarrolladores y también aportando código en los repositorios en GitHub.

5.1.1. Características

- Multiplataforma: se ejecuta en los sistemas operativos Windows, macOS y Linux.
- Código abierto: el marco .NET Core es de código abierto, con licencias de MIT y Apache 2. .NET Core es un proyecto de .NET Foundation.
- Moderno: implementa paradigmas modernos como programación asincrónica, patrones que no son de copia que usan estructuras y gobernanza de recursos para contenedores.
- Rendimiento: proporciona alto rendimiento con características como intrínsecos de hardware, compilación en niveles e Intervalo<T>.
- Coherente entre entornos: el código se ejecuta con el mismo comportamiento en varios sistemas operativos y varias arquitecturas, como x64, x86 y ARM.
- Herramientas de línea de comandos: incluye herramientas de línea de comandos sencillas que se pueden usar para el desarrollo local y la integración continua.
- Implementación flexible: se puede incluir .NET Core en la aplicación o de forma paralela (instalaciones a nivel de usuario o de sistema). Se puede usar con contenedores de Docker.

5.1.2. API

.NET Core expone marcos para compilar cualquier tipo de aplicación:

- Aplicaciones en la nube con ASP.NET Core
- Aplicaciones móviles con Xamarin
- Aplicaciones de IoT con System.Device.GPIO
- Aplicaciones cliente de Windows con WPF y Windows Forms
- Aprendizaje automático ML.NET

.NET Core proporciona compatibilidad con las API .NET Framework y Mono API implementando la especificación de .NET Standard.

.NET Core consta de las siguientes partes:

- El runtime de .NET Core, que proporciona un sistema de tipos, la carga de ensamblados, un colector de elementos no usados, interoperabilidad nativa y otros servicios básicos. Las bibliotecas de .NET Core Framework proporcionan tipos de datos primitivos, tipos de composición de aplicaciones y utilidades fundamentales.

- El runtime de ASP.NET, el cual proporciona un marco para crear aplicaciones modernas conectadas a Internet y basadas en la nube, como aplicaciones web, aplicaciones de IoT y back-ends móviles.
- El SDK de .NET Core y los compiladores de lenguaje (Roslyn y F#) que habilitan la experiencia de desarrollador de .NET Core.
- El comando dotnet, que se usa para iniciar aplicaciones .NET Core y comandos de CLI. Se selecciona y lo hospeda, proporciona una directiva de carga de ensamblados e inicia aplicaciones y herramientas.

5.1.3. Casos de éxito

.NET Core es utilizado por numerosas compañías en productos de primer orden como GoDaddy, Setpoint Medical o incluso Stack Overflow. Se pueden consultar más productos que utilizan la plataforma .NET Core en el .NET Customers Showcase.

5.2. Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows , Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto, aunque la descarga oficial está bajo software privativo e incluye características personalizadas por Microsoft.

Visual Studio Code se basa en Electron, un framework que se utiliza para implementar Chromium y Node.js como aplicaciones para escritorio, que se ejecuta en el motor de diseño Blink. Aunque utiliza el framework Electron, el software no usa Atom y en su lugar emplea el mismo componente editor (Monaco) utilizado en Visual Studio Team Services (anteriormente llamado Visual Studio Online).

5.2.1. Características

Visual Studio Code es compatible con varios lenguajes de programación. Muchas de las características de Visual Studio Code no están expuestas a través de los menús o la interfaz de usuario. Más bien, se accede a través de la paleta de comandos o a través de archivos .json (por ejemplo, preferencias del usuario). La paleta de comandos es una interfaz de línea de comandos.

Visual Studio Code se puede extender a través de complementos, disponible a través de un repositorio central. Esto incluye adiciones al editor y soporte de idiomas. Una característica notable es la capacidad de crear extensiones que analizan código, como linters y herramientas para análisis estático, utilizando el Protocolo de Servidor de Idioma.

5.3. Xamarin

Xamarin es una plataforma de código abierto bajo licencia MIT para el desarrollo de aplicaciones para iOS, Android y Windows con .NET. Xamarin es una capa de abstracción que administra la comunicación de código compartido con el código de plataforma subyacente. Xamarin se ejecuta en un entorno administrado que proporciona ventajas

como la asignación de memoria y la recolección de elementos no utilizados. El proyecto está disponible bajo licencia MIT en su repositorio de Github.

Xamarin permite a los desarrolladores compartir un promedio del 90 % de la aplicación entre plataformas. Este patrón permite a los desarrolladores escribir toda la lógica de negocios en un solo lenguaje (o reutilizar el código de aplicación existente), pero conseguir un rendimiento y una apariencia nativos en cada plataforma.

Las aplicaciones de Xamarin se pueden escribir en PC o Mac, y compilar en paquetes de aplicación nativos, como un archivo .apk en Android o .ipa en iOS.

5.4. Entity Framework Core

Entity Framework 6 (EF6) es un asignador relacional de objetos (O/RM) probado para .NET con más de diez años de desarrollo de características y estabilización, aunque ya no se desarrolla activamente.

EF Core es una versión más moderna, ligera y extensible de Entity Framework que tiene capacidades y ventajas muy similares a EF6. EF Core es producto de una reescritura completa y contiene muchas características nuevas que no están disponibles en EF6, aunque todavía carece de algunas de las funcionalidades más avanzadas de asignación de EF6. Microsoft recomienda el uso de Entity Framework Core siempre y cuando las características del desarrollo se ajusten a los requisitos.

Ambos funcionan con SQL Server o SQL Azure, SQLite, Azure Cosmos DB, MySQL, PostgreSQL y muchas otras bases de datos a través de un modelo de complemento de proveedor de bases de datos.

Tanto EF6 como EF Core se distribuyen bajo licencia Apache 2.0 y sus fuentes están disponibles en sus respectivos repositorios en Github.

5.5. .NET Core SignalR

.NET Core SignalR es una biblioteca de Open Source distribuida junto con .NET Core bajo licencia Apache 2.0 que simplifica la incorporación de funcionalidades Web en tiempo real a las aplicaciones. La funcionalidad web en tiempo real permite que el código del lado servidor inserte contenido en los clientes al momento. SignalR proporciona una API para crear llamadas a procedimiento remoto (RPC)de servidor a cliente. Las RPC llaman a funciones de JavaScript en los clientes desde el código de .NET Core del lado servidor. El código fuente de esta biblioteca puede encontrarse en su repositorio de Github.

Estas son algunas características de SignalR para ASP.NET Core:

- Controla la administración de conexiones automáticamente.
- Envía mensajes a todos los clientes conectados simultáneamente. Por ejemplo, un salón de chat.
- Envía mensajes a clientes o grupos de clientes específicos.
- Escala para controlar el aumento del tráfico.

Existe una versión de SignalR para .NET Framework Standard que también se distribuye bajo Apache 2.0 y sus fuentes está disponibles en Github.

6. Aplicación .NET Core desarrollada en entorno Linux

A modo de demostración de uso de las tecnologías libres de Microsoft se ha desarrollado una pequeña aplicación con .NET Core sobre Linux. Esta aplicación está disponible, junto con este documento y las transparencias en un repositorio de Github[1].

Se trata de una aplicación web MVC que utiliza la biblioteca de comunicaciones en tiempo real SignalR y el ORM Entity Framework Core para persistir algunos datos en una base de datos administrada por un servidor SQL-Server en el mismo sistema Linux. La aplicación tiene las siguientes funcionalidades:

- Lista de tareas colaborativa con persistencia.
- Chat en tiempo real con persistencia de los mensajes.
- Pizarra colaborativa en tiempo real.

Para el desarrollo de la aplicación se han utilizado las siguientes herramientas:

- sqlcmd: Herramienta de comandos para configuración de SQL-Server.
- Editor Visual Studio Code para la edición del código, la elaboración de este documento, de las transparencias y de las preguntas de test en formato GIFT.
- Plugin de Visual Studio Core para consultas contra la base de datos.
- Herramienta cliente de .NET Core (dotnet) para la gestión de paquetes del proyecto y la generación del modelo OO a partir del esquema relacional de la base de datos.
- Servidor web Kestrel.

La instalación de .net Core SDK se ha realizado siguiendo las instrucciones del Microsoft[2]. La creación del proyecto ha partido de la plantilla de aplicación web MVC de .NET Core que se obtiene mediante la herramienta `dotnet` desde línea de comandos:

```
$ dotnet new mvc -o tasklist
```

Utilizando la misma herramienta se han instalado la herramienta de scaffolding de Entity Framework Core y las dependencias del proyecto: SignalR y Entity Framework Core:

```
$ dotnet tool install --global dotnet-ef
$ dotnet add package Microsoft.AspNetCore.SignalR.Client
$ dotnet add package Microsoft.EntityFrameworkCore.Design
$ dotnet add package Microsoft.EntityFrameworkCore.SqlServer
```

El modelado de la base de datos se ha realizado mediante el plugin mssql de Visual Studio Code, que permite visualizar la estructura y realizar consultas de todo tipo contra una base de datos de SQL-Server. La clase de contexto de la base de datos, así como los modelos correspondientes a las tablas se generan automáticamente utilizando la herramienta de línea de comandos:

```
$ dotnet ef dbcontext scaffold "<connection-string>" _
    Microsoft.EntityFrameworkCore.SqlServer -o Models
```

[1] https://github.com/eduarroyo/microsoft_y_software_libre
[2] https://docs.microsoft.com/es-es/dotnet/core/install/linux-package-manager-ubuntu-1904

7. Conclusiones

Referencias

[1] "Microsoft litigation." https://en.wikipedia.org/wiki/Microsoft_litigation, abril 2020. Accedido: 26/4/2020.

[2] M. Brown, "Ibm signs a deal with the devil." https://thisdayintechhistory.com/11/06/ibm-signs-a-deal-with-the-devil/, noviembre 2019. Accedido 26/4/2020.

[3] M. Brown, "Microsoft buys full rights to 86-dos." https://thisdayintechhistory.com/07/27/microsoft-buys-full-rights-to-86-dos/, julio 2019. Accedido 26/4/2020.

[4] Wikipedia, "Bm pc compatible." https://en.wikipedia.org/wiki/IBM_PC_compatible#Origins, marzo 2020. Accedido 26/4/2020.

[5] "Guerra de navegadores." https://es.wikipedia.org/wiki/Guerra_de_navegadores, Apr 2020. Accedido 26/4/2020.

[6] "Caso estados unidos contra microsoft." https://es.wikipedia.org/wiki/Caso_Estados_Unidos_contra_Microsoft, enero 2020. Accedido 26/4/2020.

[7] "The history of the browser wars: When netscape met microsoft." https://thehistoryoftheweb.com/browser-wars/, marzo 2019. Accedido 26/4/2020.

[8] "Microsoft corp. v. commission." https://en.wikipedia.org/wiki/Microsoft_Corp._v._Commission, marzo 2020. Accedido: 26/4/2020.

[9] "Embrace, extend, and extinguish." https://en.wikipedia.org/wiki/Embrace,_extend,_and_extinguish, marzo 2020. Accedido 26/4/2020.

[10] "Fear, uncertainty, and doubt." https://en.wikipedia.org/wiki/Fear,_uncertainty,_and_doubt, abril 2020. Accedido: 26/4/2020.

[11] "Halloween documents." https://en.wikipedia.org/wiki/Halloween_documents, marzo 2020. Accedido 26/4/2020.

[12] Wikipedia, "Microsoft and open source." https://en.wikipedia.org/wiki/Microsoft_and_open_source, 2020. Accedido 21/4/2020.

[13] G. Lea, "Ms'ballmer: Linux is communism." https://www.theregister.co.uk/2000/07/31/ms_ballmer_linux_is_communism/, julio 2000. accedido 26/4/2020.

[14] T. C. Greene, "Ballmer: 'linux is a cancer'." https://www.theregister.co.uk/2001/06/02/ballmer_linux_is_a_cancer/, octubre 2018. accedido 26/4/2020.

[15] L. Tung, "Ballmer: I may have called linux a cancer but now i love it," marzo 2016. accedido 27/4/2020.

[16] M. Saunders, M. Saunders, Mike, M. Saunders, Mike, P. Dann, D. May, S. May, P. May, T. May, and et al., "How munich switched 15,000 pcs from windows to linux." https://www.linuxvoice.com/the-big-switch/. accedido 27/4/2020.

[17] "Shared source." https://es.wikipedia.org/wiki/Shared_source#Microsoft_Public_License_(Ms-PL), noviembre 2019. Accedido: 27/4/2020.

[18] "Shared source." https://es.wikipedia.org/wiki/Shared_source#Microsoft_Reciprocal_License_(Ms-RL), noviembre 2019. Accedido: 27/4/2020.

[19] "Shared source." https://es.wikipedia.org/wiki/Shared_source#Microsoft_Reference_Source_License_(Ms-RSL), noviembre 2019. Accedido: 27/4/2020.

[20] "Shared source." https://es.wikipedia.org/wiki/Shared_source#Microsoft_Limited_Public_License_(Ms-LPL), noviembre 2019. Accedido: 27/4/2020.

[21] "Shared source." https://es.wikipedia.org/wiki/Shared_source#Microsoft_Limited_Reciprocal_License_.28Ms-LRL.29, noviembre 2019. Accedido: 27/4/2020.