Sesión 01

Introducción y evolución a Net 8

8 NET FULL-STACK DEVELOPER

Instructor:

ERICK ARÓSTEGUI

earostegui@galaxy.edu.pe

GALAXY TRAINING

ÍNDICE

GALAXY
TRAINING

01

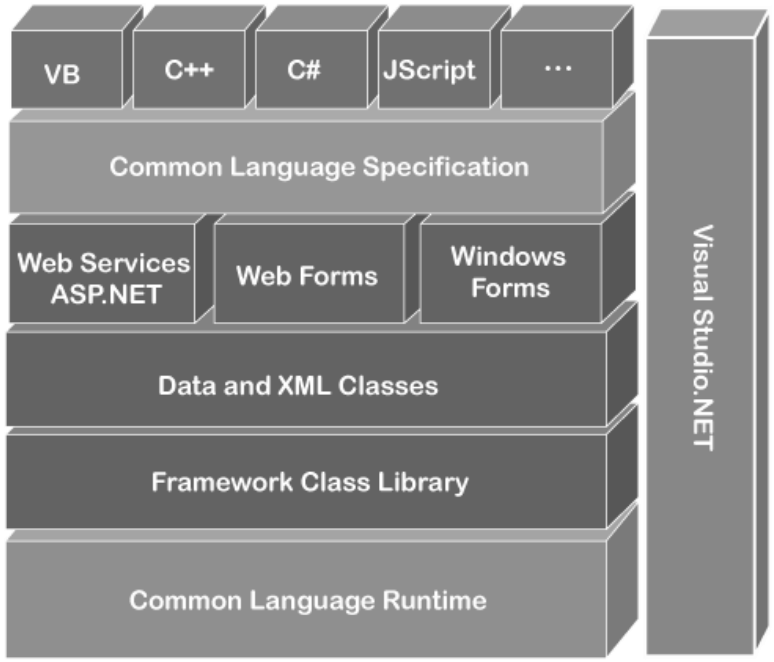
Historia y evolución de NET

La historia detrás de .NET

Microsoft comenzó a trabajar en el **framework .NET a finales de los 90**. La idea era crear una **plataforma basada en el llamado código administrado**, código que se puede ejecutar bajo un entorno de ejecución.

Esto era necesario para mejorar la experiencia de desarrollo y aliviar a los ingenieros del manejo de las operaciones de seguridad, la administración activa de la memoria y otros esfuerzos de bajo nivel con los que los desarrolladores de C / C ++ tenían que molestarse.

.NET Framework

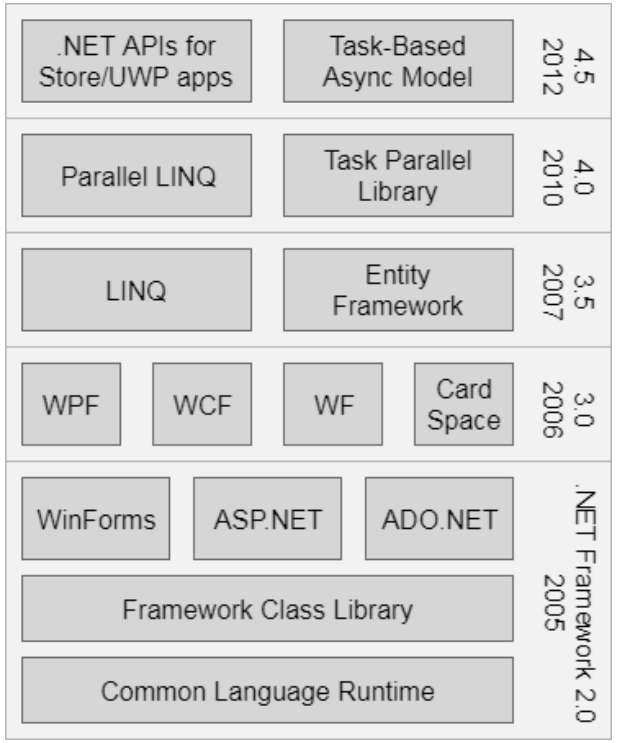


La primera versión de .NET **Framework en 2002** introdujo C#, un lenguaje para escribir código administrado que tenía un diseño similar a C++.

El marco en sí estaba dirigido a computadoras y servidores basados en Windows. **Tenía WinForms, una biblioteca GUI para aplicaciones de escritorio; ASP.NET, un framework para Web; y ADO.NET para el acceso a los datos. Common Language Runtime (CLR)** controló todos estos elementos para compilar y ejecutar código administrado.

GALAXY TRAINING

.NET Framework



Para unir varias funciones, **.NET ofrecía un Framework Class Library (FCL)** que incluía la **Base Class Library (BCL)**, la biblioteca de red, una biblioteca numérica y otras.

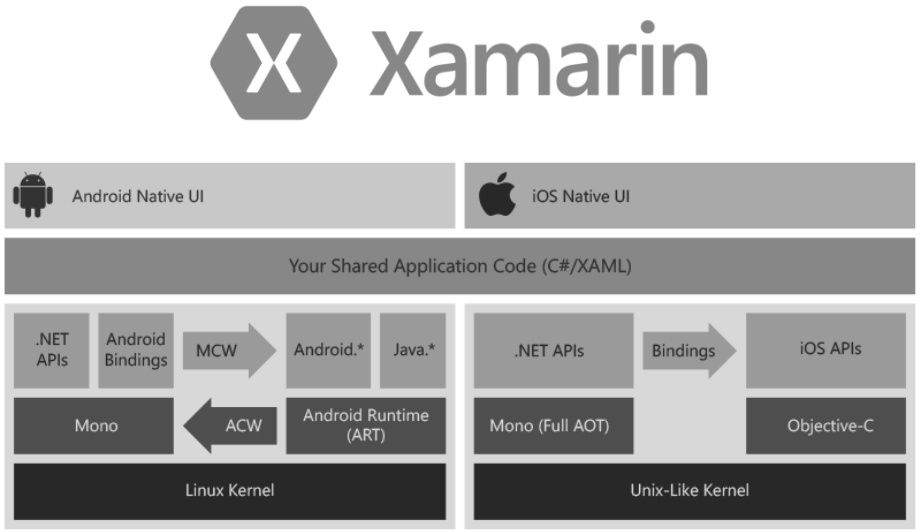
Desde entonces, el framework ha sufrido múltiples iteraciones que abarcan actualizaciones en tiempo de ejecución, nuevos **sistemas gráficos de escritorio (WPF),** API para aplicaciones orientadas a **servicios (WCF)** y más.

.NET CORE

En 2014, Microsoft anunció un cambio dramático en la forma en que existe .NET al presentar **.NET Core**, una nueva versión multiplataforma, compatible con la nube y de código abierto

.NET Core llegó a su **lanzamiento en 2016**, convirtiéndose en la principal tecnología a considerar para los nuevos proyectos. **Microsoft comenzó a portar los servicios existentes** para trabajar con Core. Algunos no recibieron portabilidad oficial, como **Windows Communication Foundation (WCF)** y fueron sustituidos por alternativas procedentes de la comunidad.

.NET CORE



En 2016, **Microsoft adquirió Xamarin**, anteriormente una tecnología patentada para el desarrollo móvil multiplataforma, lo que también la convierte en código abierto.

Microsoft continuó avanzando hacia la **"transparencia entre el equipo del producto y la comunidad"** y los Frameworks Windows Presentation Foundation (WPF), Windows Forms y WinUI en diciembre de 2018 se convirtieron en proyectos de código abierto .

NET

En mayo de 2019, Microsoft anunció el lanzamiento que uniría el ecosistema: se suponía que todos los elementos de .NET se incluirían en la plataforma de desarrollo .NET 5. Si bien se realizaron cambios en el cronograma debido a COVID-19, la plataforma de desarrollo unificado .NET 5 finalmente se introdujo en noviembre de 2020.

El sucesor de .NET Core 3.1 y .NET Framework 4.8, .NET 5 pone orden en la fragmentación del mundo .NET y proporciona muchas características para crear aplicaciones en Windows, Linux, macOS, iOS, watchOS, Android, tvOS o mediante WebAssembly. La plataforma viene con nuevas API, características de lenguaje y capacidades de tiempo de ejecución. Además, .NET 5 incluye ASP.NET Core, Xamarin, Entity Framework Core, WPF, WinForms y ML.NET.

Historia y evolución de NET

GALAXY TRAINING

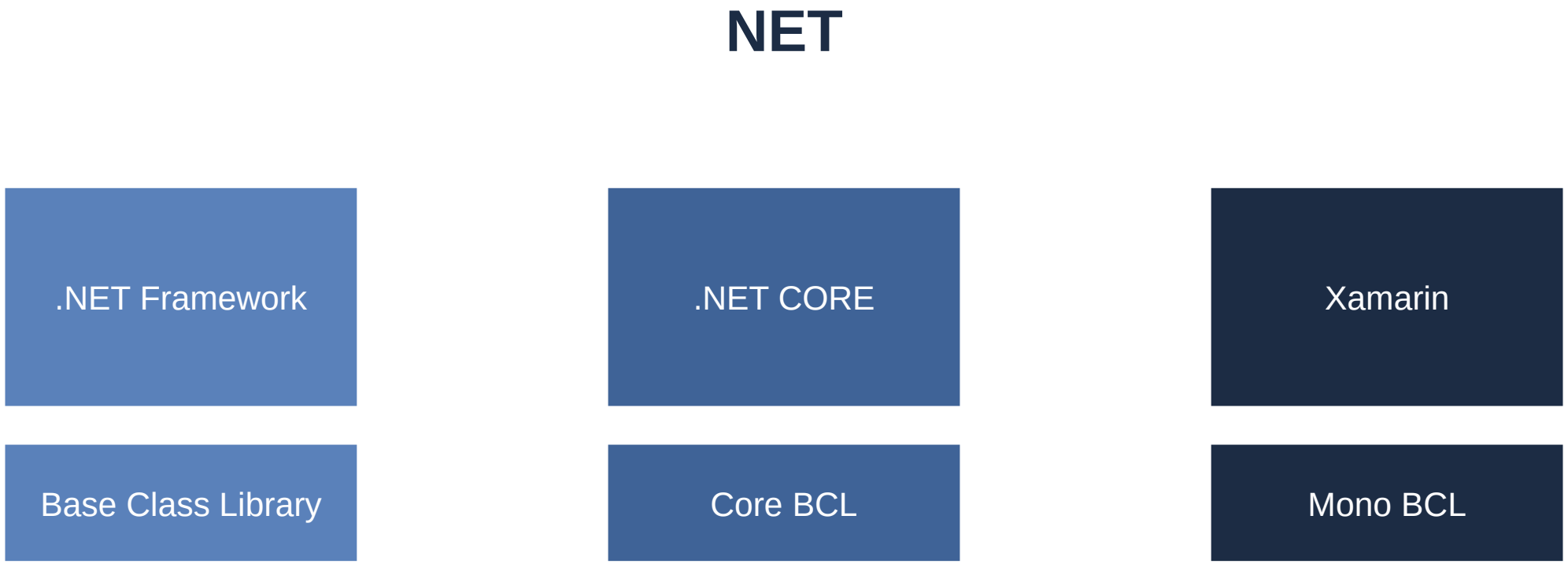
NET

.NET Framework

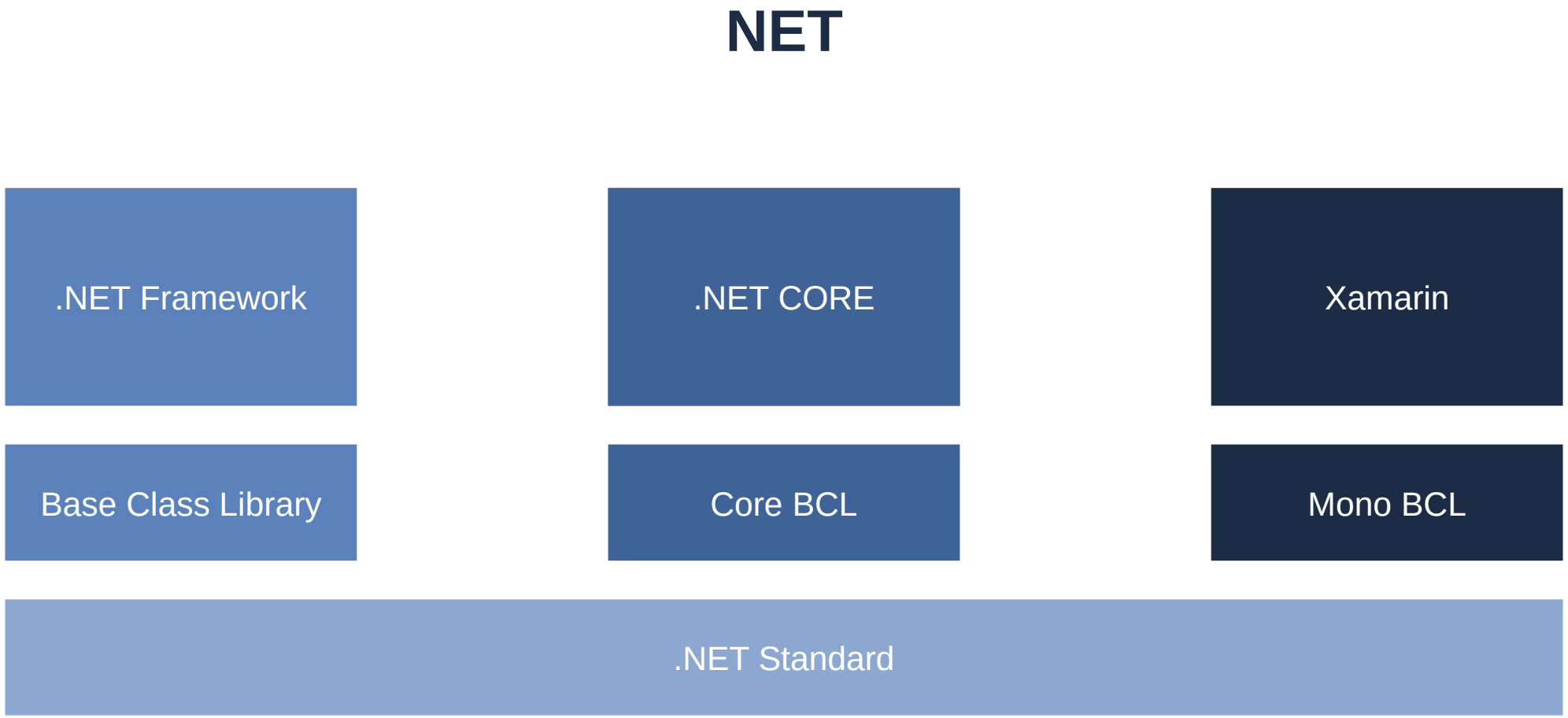
.NET CORE

Xamarin

Historia y evolución de NET



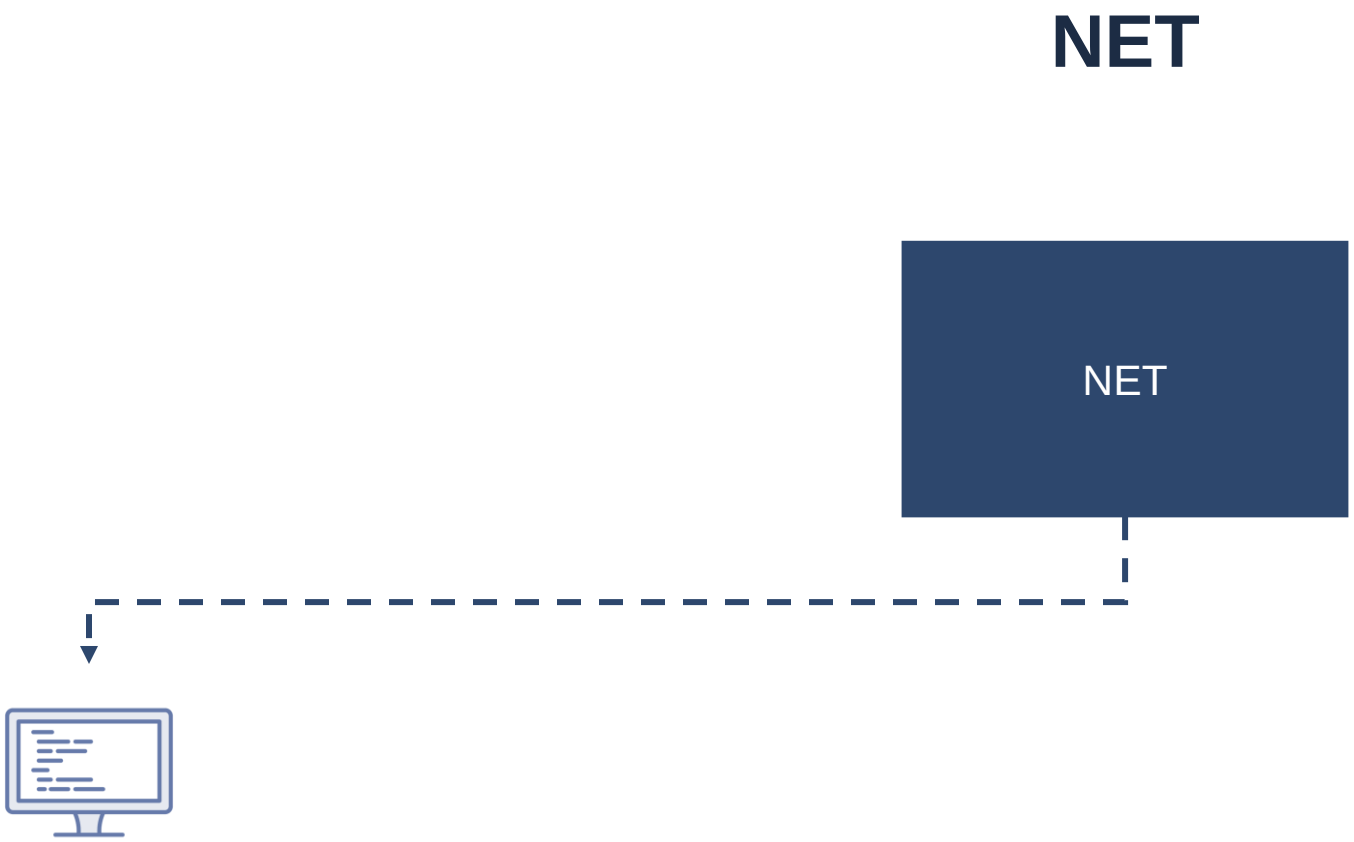
Historia y evolución de NET



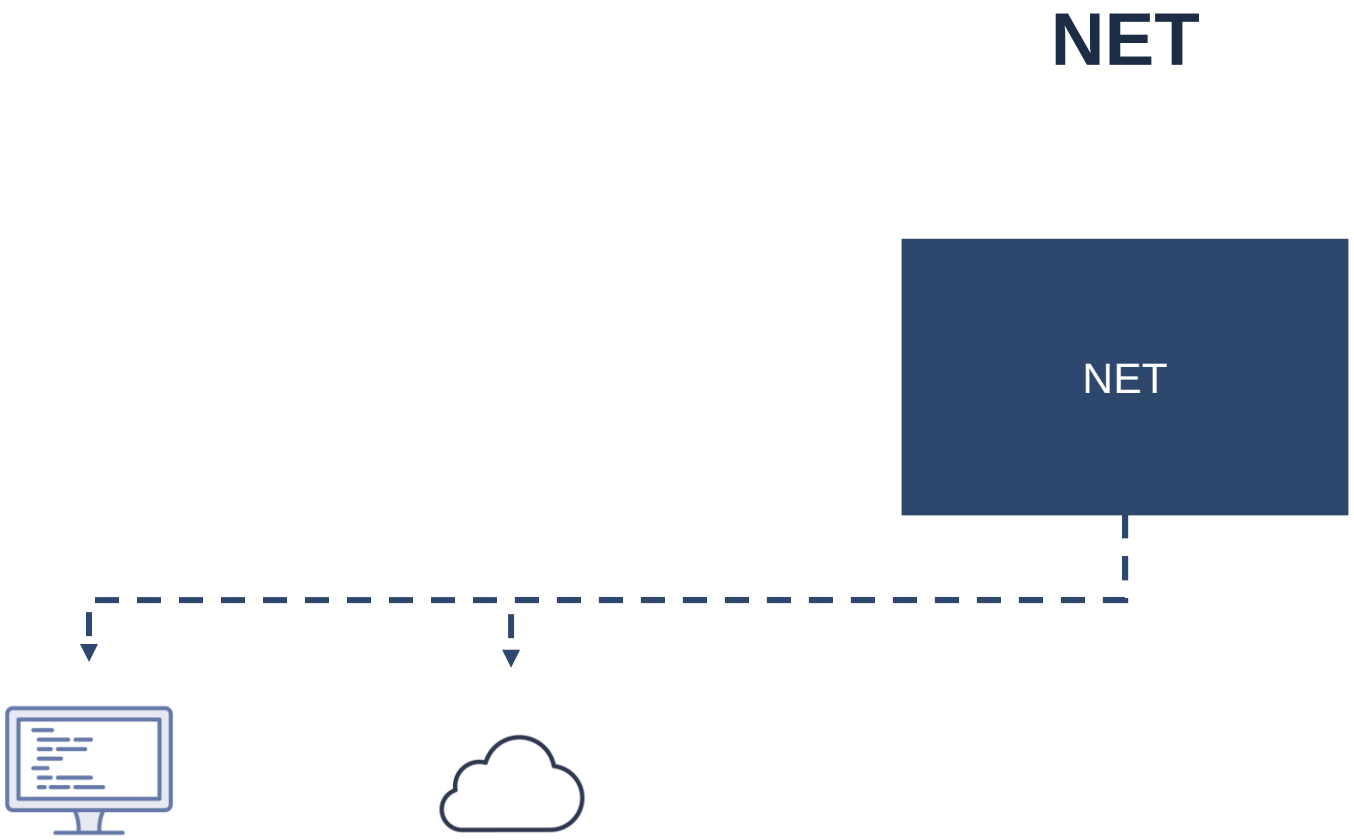
Historia y evolución de NET

NET

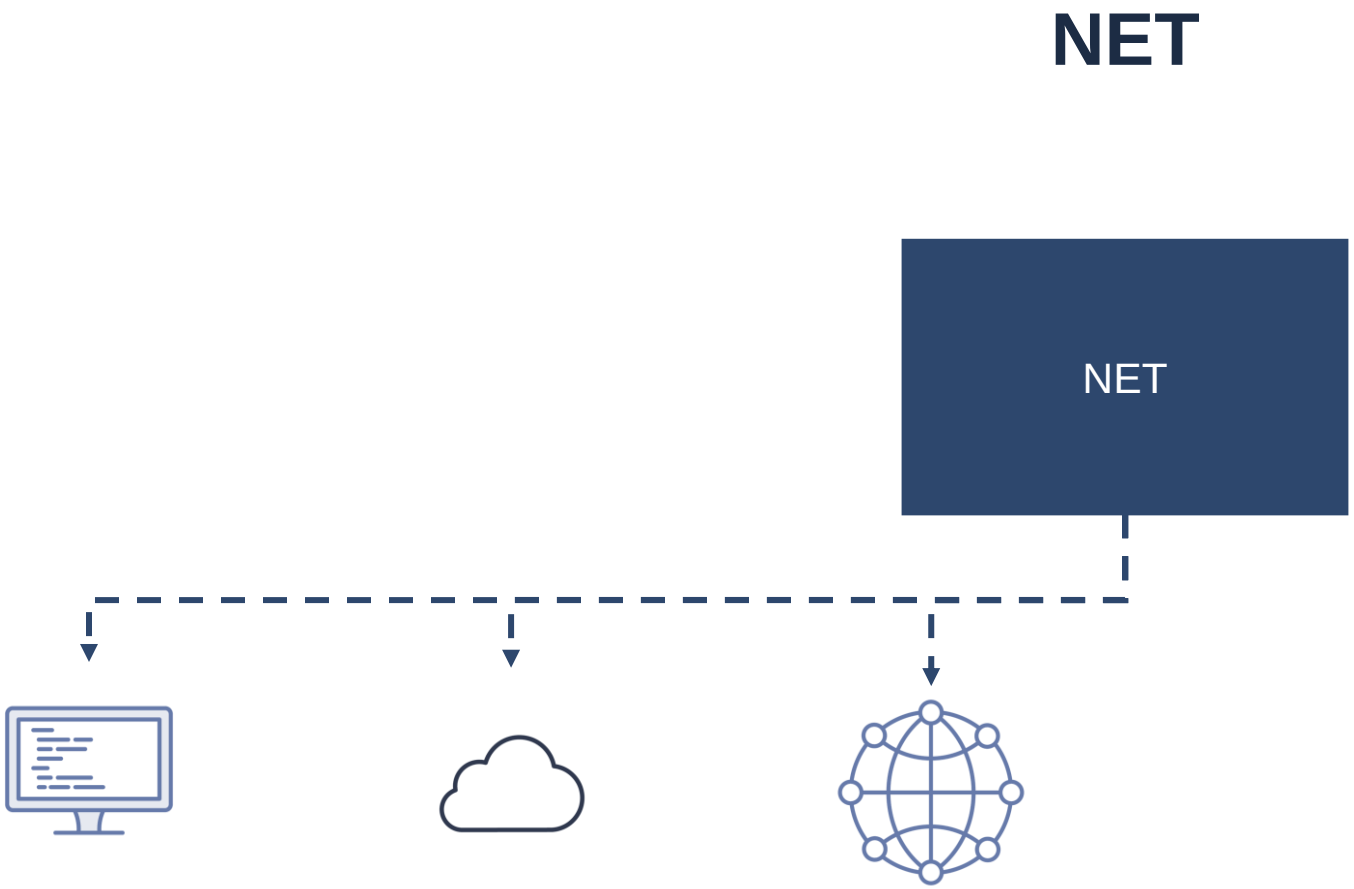
Historia y evolución de NET



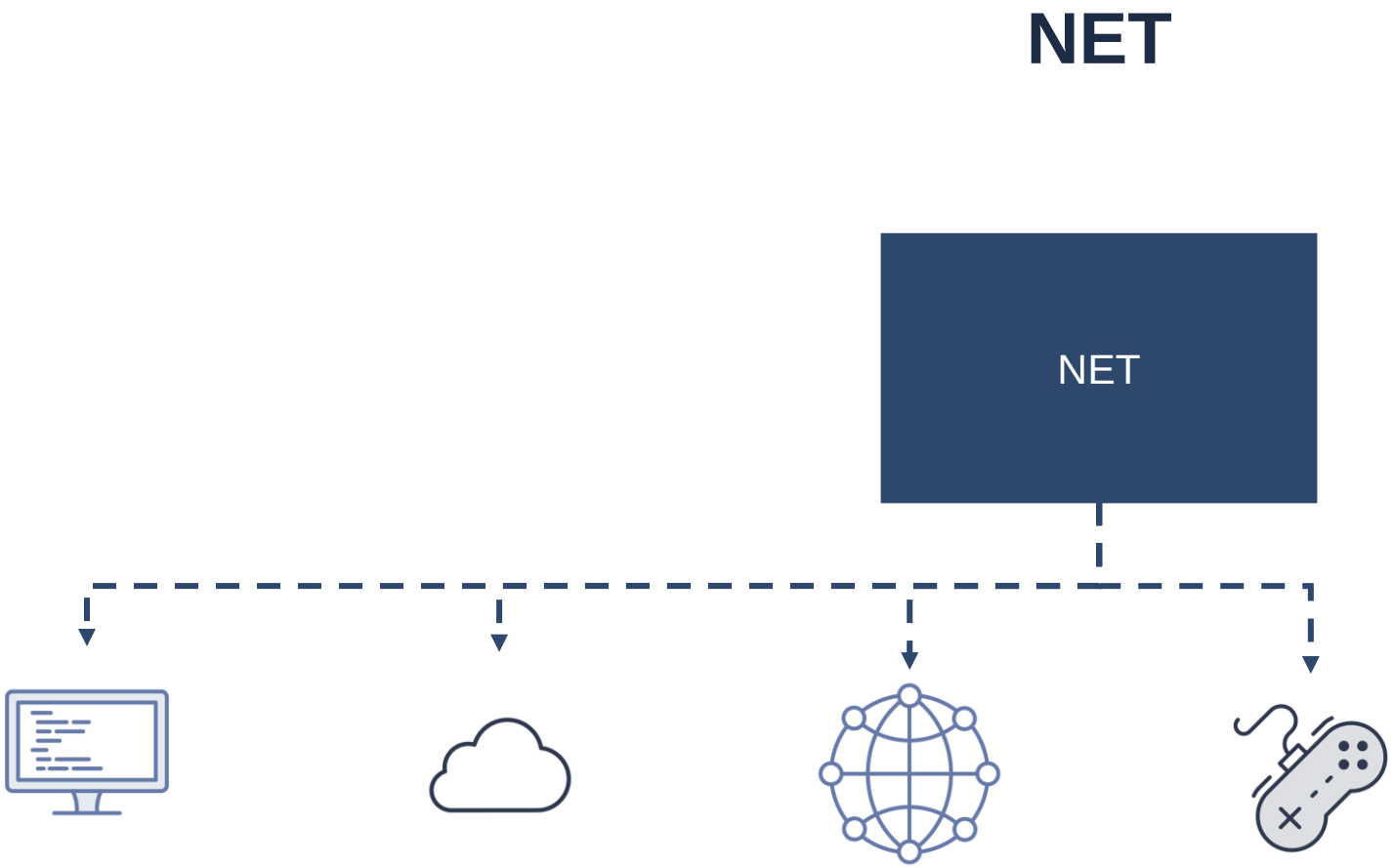
Historia y evolución de NET



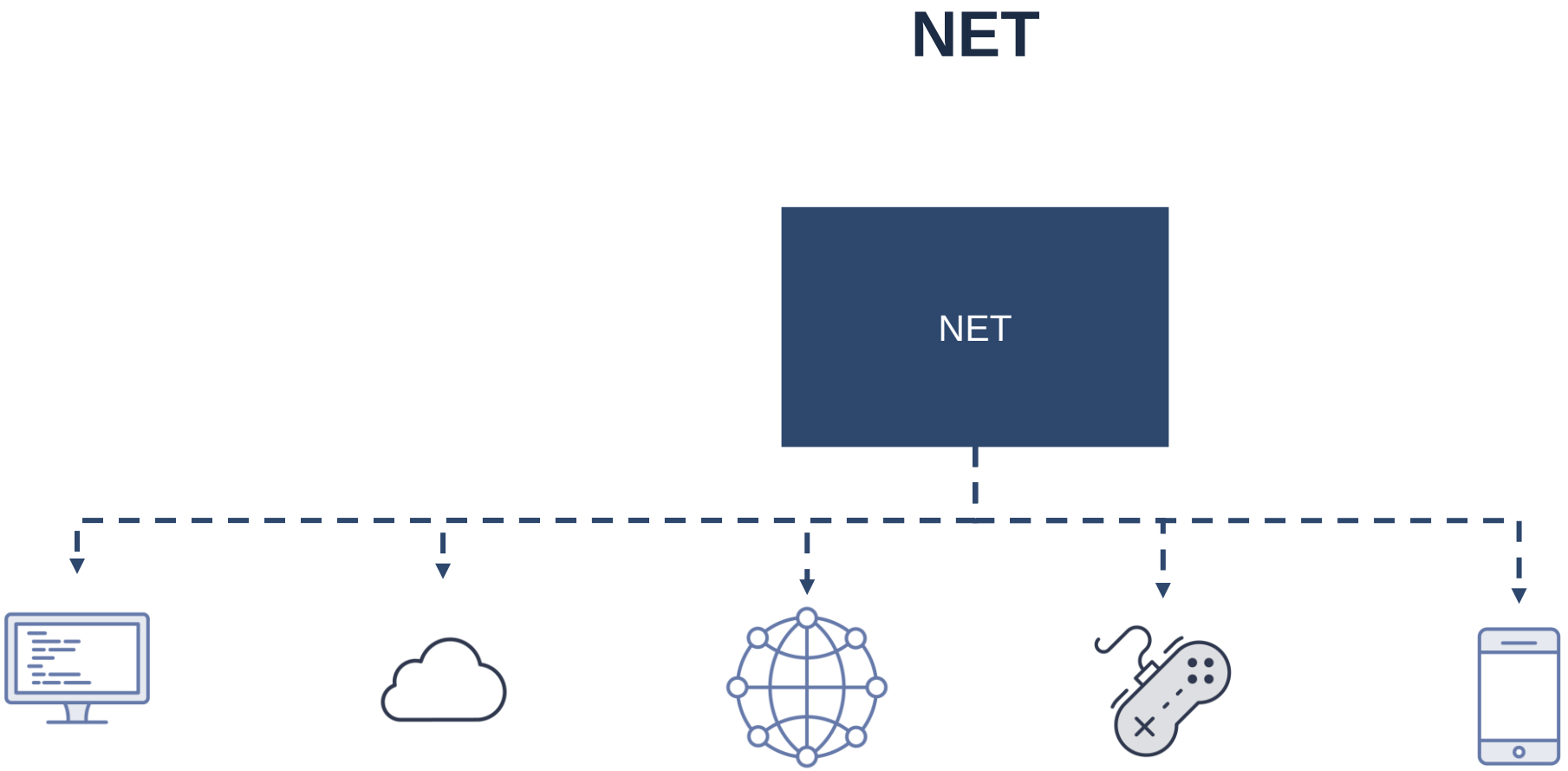
Historia y evolución de NET



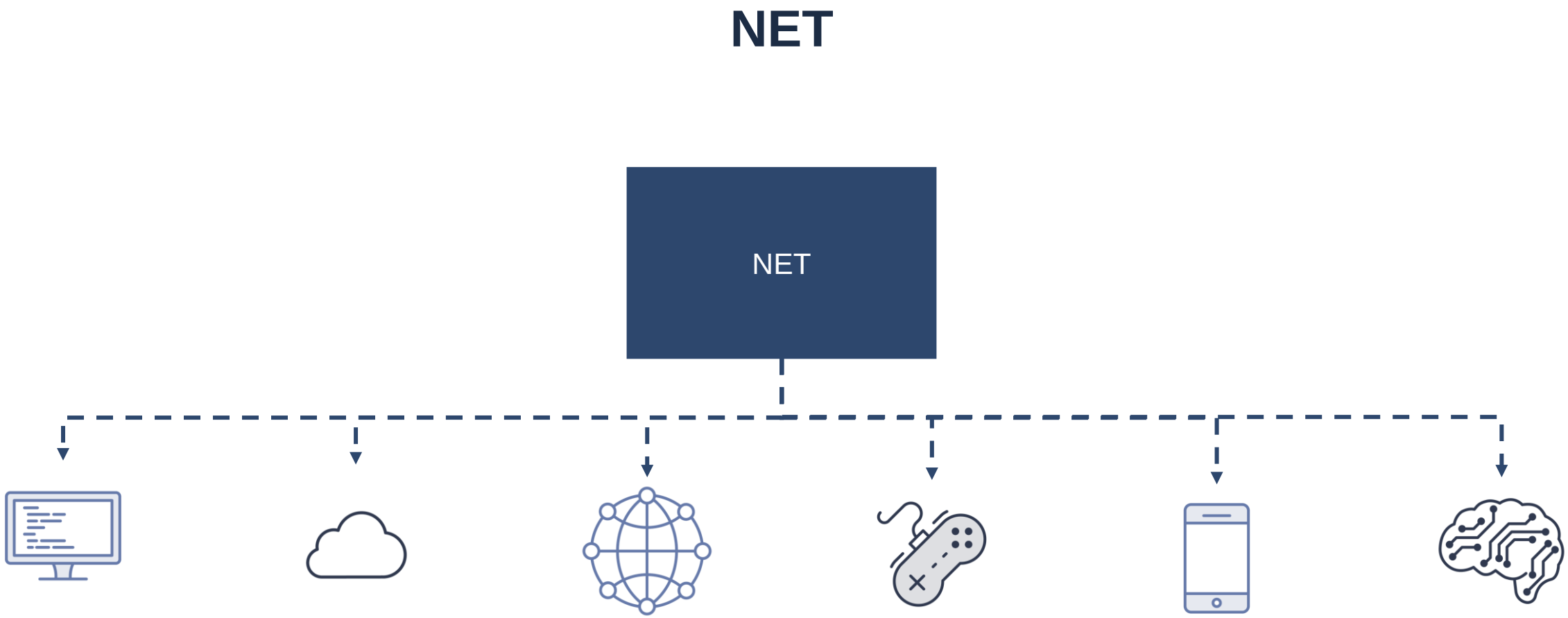
Historia y evolución de NET



Historia y evolución de NET

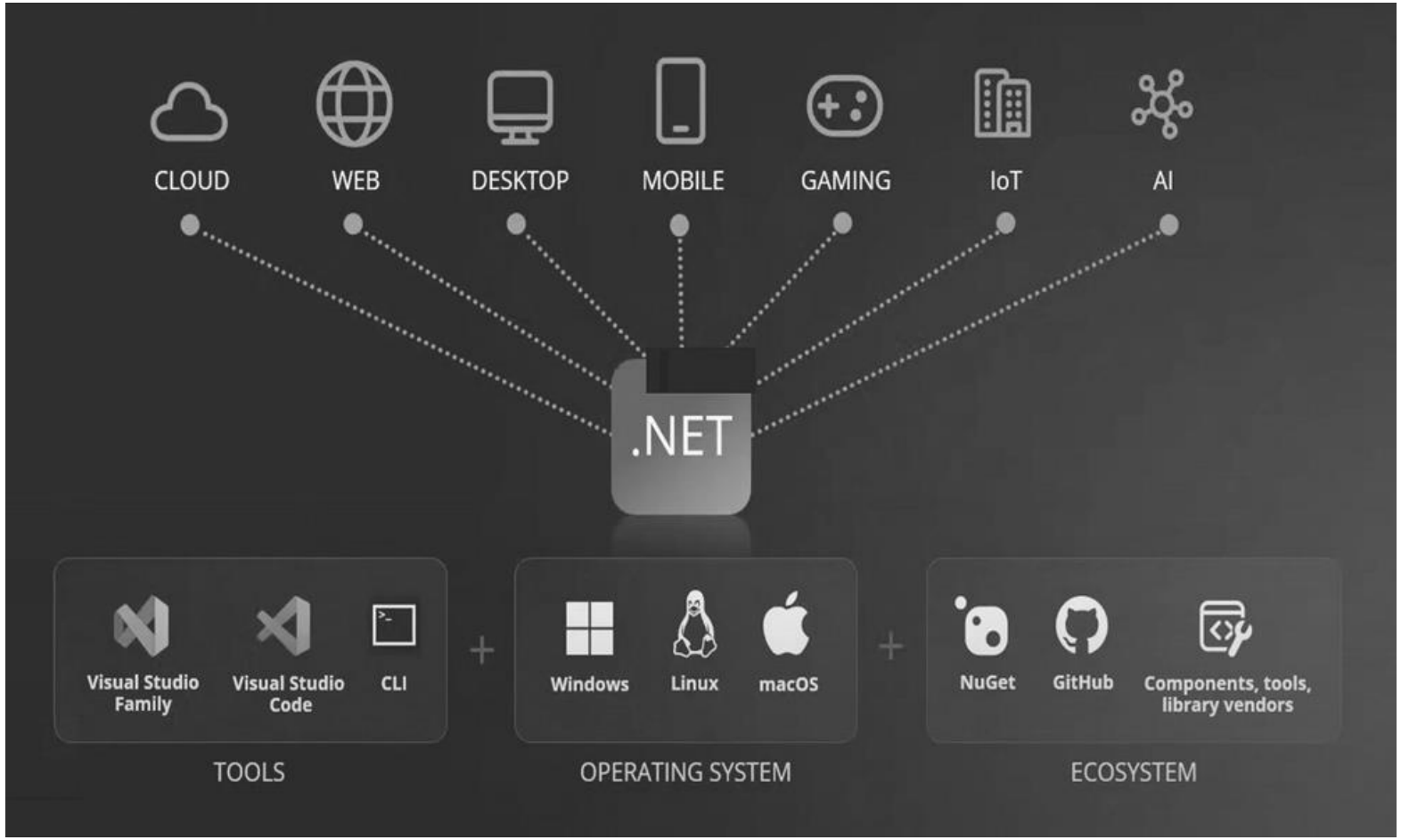


Historia y evolución de NET



Historia y evolución de NET

NET



.NET 5 estableció las bases de unificación, la versión de .NET 6 se entregó las partes finales en noviembre de 2021, y Visual Studio 2022 se lanzó el mismo día. **NET 7 es una plataforma unificada** para crear proyectos en entornos de nube, explorador, IoT, móviles y de escritorio, lo que permite a todos usar las mismas bibliotecas .NET, SDK y tiempo de ejecución.

NET

Historia y evolución de NET

GALAXY TRAINING

NET

2016 – 2018

.NET Core
1.0 - 1.1 – 2.0 – 2.1

Historia y evolución de NET

GALAXY TRAINING

NET

2016 – 2018

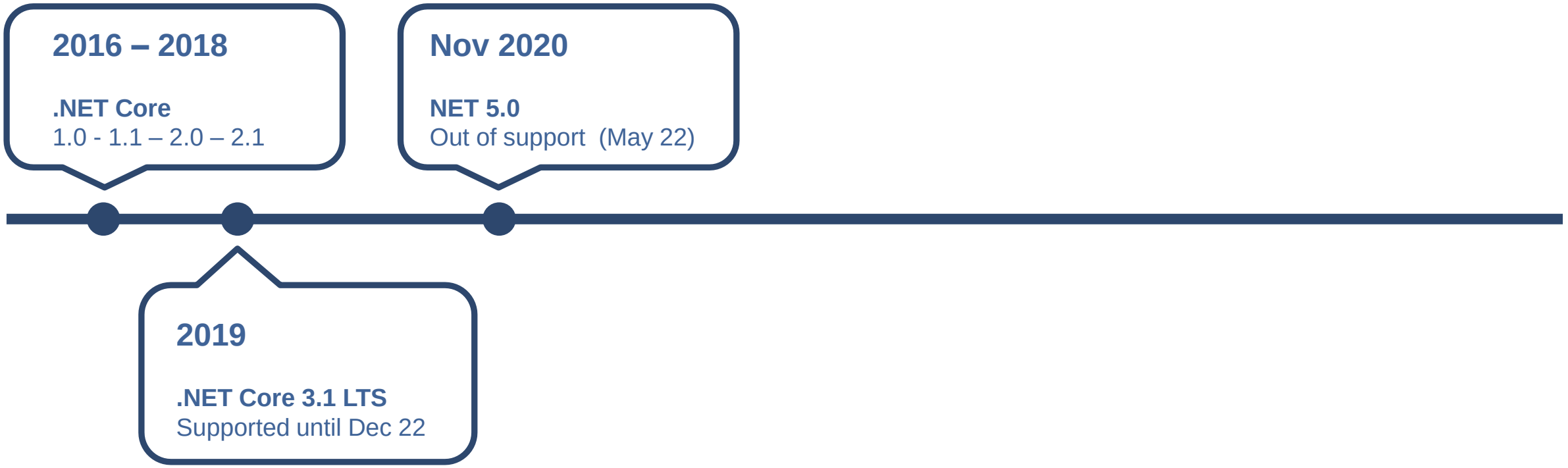
.NET Core
1.0 - 1.1 – 2.0 – 2.1

2019

.NET Core 3.1 LTS
Supported until Dec 22

Historia y evolución de NET

NET

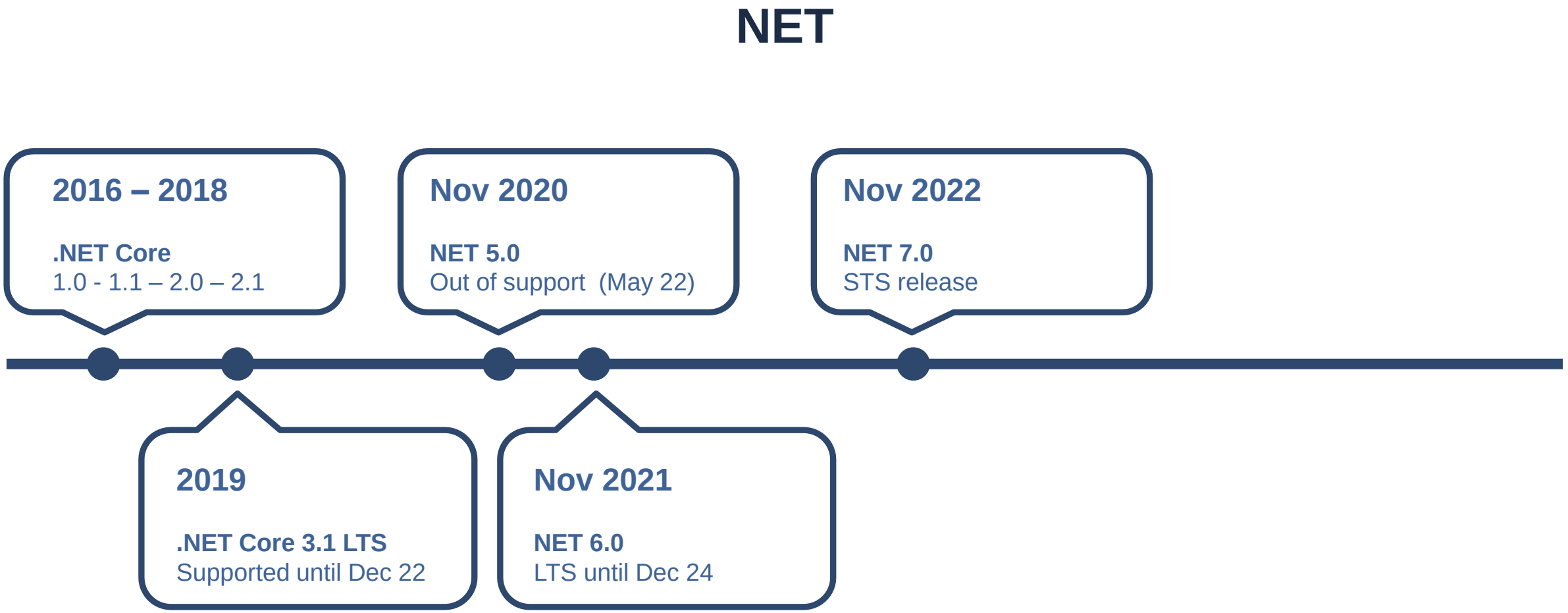


Historia y evolución de NET

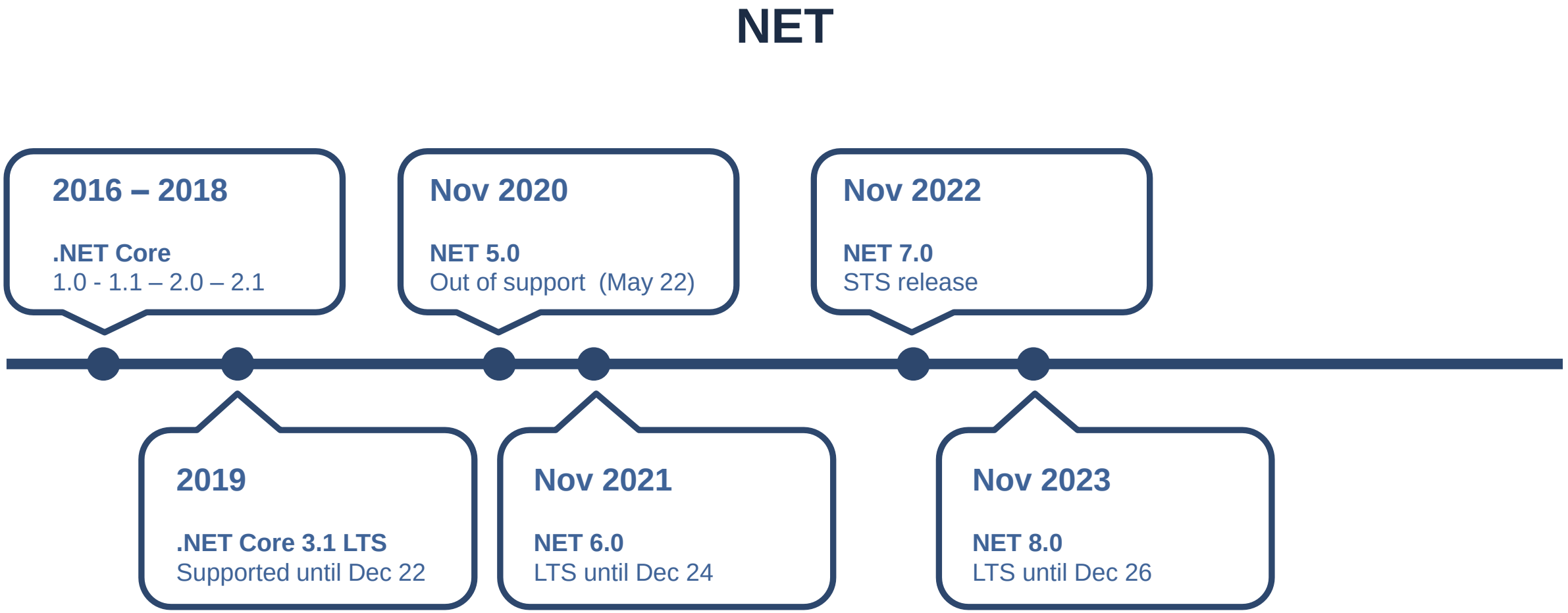
GALAXY TRAINING

NET

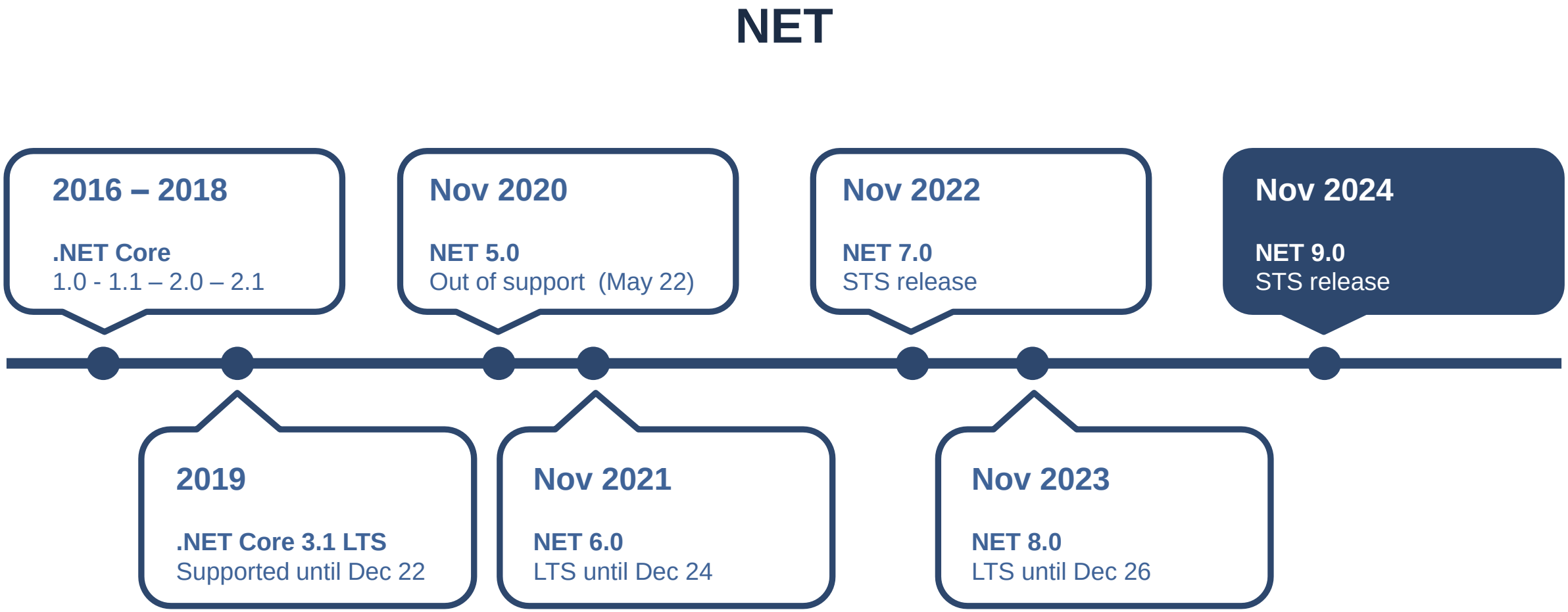
Historia y evolución de NET



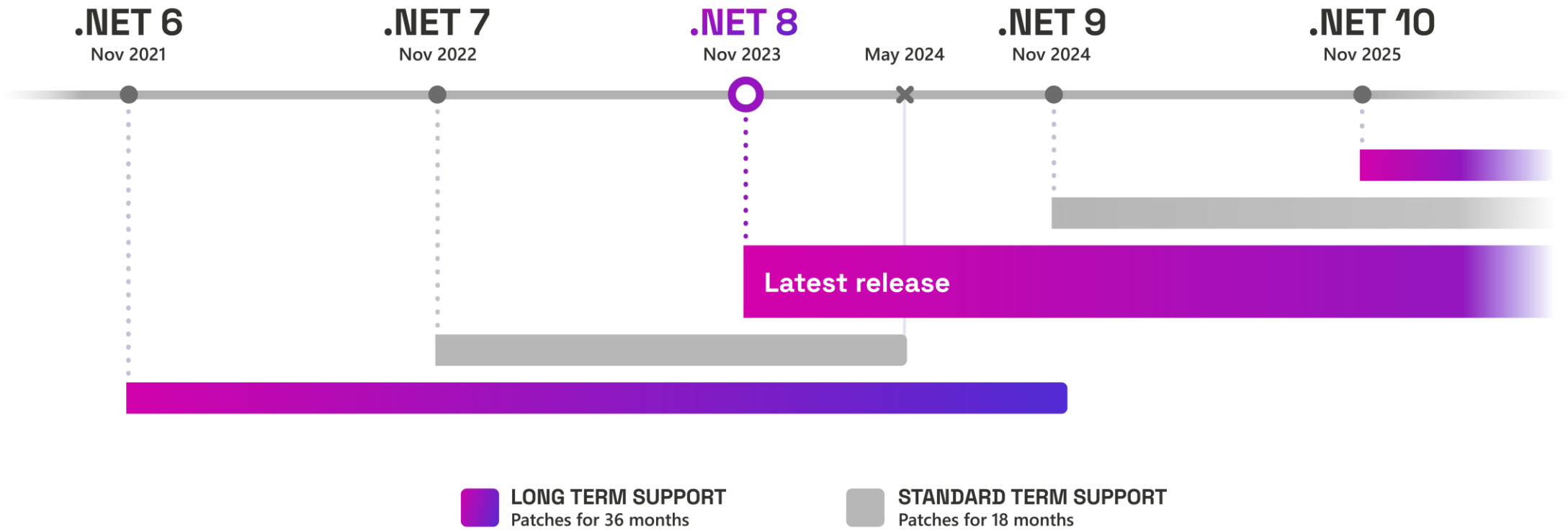
Historia y evolución de NET



Historia y evolución de NET



Historia y evolución de NET



Long Term Support (LTS)

Las versiones de LTS tendrán soporte durante tres años después de la versión inicial.

Standard Term Support (STS)

Las versiones STS tendrán soporte durante seis meses después de una versión STS o LTS posterior. Las versiones se producen cada 12 meses, por lo que el período de soporte técnico para STS es de 18 meses.

https://dotnet.microsoft.com/en-us/platform/support/policy/dotnet-core

02

GALAXY
TRAINING

¿Qué es NET 8?,
Ventajas que ofrece NET 8

¿Qué es NET 8?, Ventajas que ofrece NET 8

NET Es una plataforma gratuita popular que se utiliza actualmente para muchos tipos diferentes de aplicaciones, ya que proporciona el entorno de programación para la mayoría de las fases de desarrollo de software.

.**NET se adapta mejor a las empresas que buscan una amplia gama de características**, como servicios basados en web, software de escritorio y compatibilidad con infraestructura en la nube.

¿Qué es NET 8?, Ventajas que ofrece NET 8

GALAXY TRAINING

.NET foundation

Openness

Community

Rapid innovation

.NET Compiler Platform ("Roslyn")
LLILC
MVVM Light Toolkit
MSBuild
ASP.NET MVC
MEF
IdentityManager
ASP.NET Core
.NET SDK for Hadoop
Kudu
.NET Core
.NET Micro Framework
Mono
Mailkit
Cecil
Xamarin.Auth
Mimekit
Umbraco
ASP.NET AJAX Control Toolkit
Open Live Writer
NuGet
Cake
WorldWide Telescope
Couchbase Lite for .NET
ASP.NET SignalR
ASP.NET Web Pages
Microsoft Azure SDK for .NET
WCF
Xamarin SDK
Entity Framework
Open XML SDK
IdentityServer
Microsoft Azure WebJobs SDK
OWIN Authentication Middleware
Microsoft Web Protection Library
ASP.NET Web API
Prism
System.Drawing
Orchard CMS
ProtoBuild
Xamarin.Mobile
Salesforce Toolkits for .NET
Orleans

Features

Protection
- Licenses
- Copyrights
- Trademarks
- Patents

Practices
- Mentorship
- Governance
- Feedback
- Co-ordination

Visibility
- Media
- Branding
- Events

Support
- Hosting
- Code signing
- CLA Management
- Swag

¿Qué es NET 8?, Ventajas que ofrece NET 8

Ventajas

Modelo de desarrollo de orientado a objetos

Monitorización automática en ASP.NET

IDE Visual Studio

Popularidad y comunidad de .NET

Potentes compiladores Roslyn y RyuJIT

Soporte técnico activo de Microsoft

Implementación flexible y de fácil mantenimiento

Diseño multiplataforma

¿Qué es NET 8?, Ventajas que ofrece NET 8

Ventajas Net 8

Mejoras en el rendimiento

Soporte para C# 12

.NET MAUI

ASP.NET Core y EF Core mejoras

Windows Forms y Windows Presentation Foundation (WPF) actualizaciones

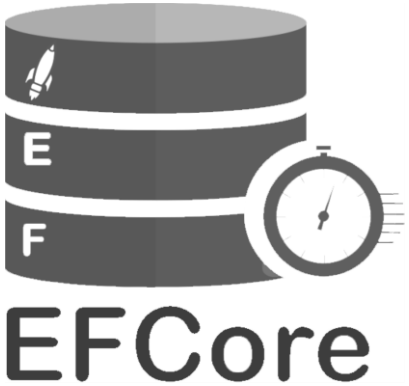
Mejor manejo de recursos, lo que lleva a una mayor velocidad y estabilidad.

.NET Aspire

Una nueva pila preparada para la nube que facilita la construcción de aplicaciones observables, distribuidas y listas para producción, con paquetes NuGet que abordan problemas específicos nativos de la nube.

¿Qué es NET 8?, Ventajas que ofrece NET 8

Desafíos



Soporte relacional de objetos limitado

Costo de licencia

La brecha entre la liberación y la estabilidad

Transición de .Net Framework a NET

03

GALAXY
TRAINING

Principales características de NET 8

Principales características de NET 8

Características

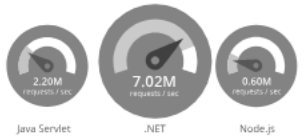
Multi-lenguaje

Any app, any platform

Maduro(Consolida lo mejor de Net Framework)

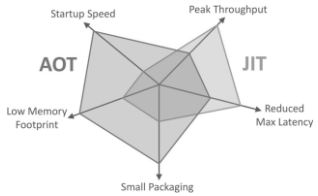
Diseñado para desarrollo de Aplicaciones Cloud Nativas (Cloud)

Permite el desarrollo de aplicaciones diversas(web, móvil, microservicios, IoT, IA, ML)



On-stack replacement (OSR), Permite que el tiempo de ejecución cambie el código ejecutado

Generación de código mejorada para Arm64



AOT nativo genera un ejecutable independiente en el formato de archivo de la plataforma

04

Características de C# 12

GALAXY
TRAINING

Características de C# 11

- **Literales de cadena sin formato**. Un literal de cadena sin formato comienza con al menos tres caracteres de comillas dobles (""")
- **Literales de cadena de UTF-8**. Puede especificar el sufijo u8 en un literal de cadena para especificar la codificación de caracteres UTF-8
- **Miembros requeridos**. Puede agregar el modificador **required** a propiedades y campos para aplicar constructores y llamadores para inicializar esos valores
- **Estructuras predeterminadas automáticas.** Este cambio significa que el compilador inicializa automáticamente cualquier campo o propiedad automática no inicializados por un constructor.
- **Tipos locales de archivo.** El modificador de acceso **file** se puede usar para crear un tipo cuya visibilidad esté limitada al archivo de origen en el que se declara
- **Patrones de lista.** Amplía la coincidencia de patrones para buscar coincidencias con secuencias de elementos de una lista o una matriz

La lista completa en https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-11

Características de C# 12

- **Constructores principales**. Los parámetros del constructor principal están en el ámbito en toda la definición de clase. Es importante ver los parámetros del constructor principal como parámetros. Todos los demás constructores de una clase deben llamar al constructor principal, directa o indirectamente, a través de una invocación de constructor **this()**.
- **Alias de cualquier tipo**. Permitirle apuntar a cualquier tipo de tipo, no solo a los tipos con nombre (namespace). Esto admitiría tipos que no se permiten hoy en día, como: tipos de tupla, tipos de puntero, tipos de matriz, etc.
- **Expresiones de colección**. Puede usar una expresión de colección para crear valores de colección comunes. Una expresión de colección es una sintaxis tersa que, cuando se evalúa, se puede asignar a muchos tipos de colección diferentes.
- **Parámetros lambda predeterminados.** Ahora puede definir valores predeterminados para parámetros en expresiones lambda. La sintaxis y las reglas son las mismas que agregar valores predeterminados para los argumentos a cualquier método o función local.
- **Interceptores.** Los interceptores proporcionan una instalación limitada para cambiar la semántica del código existente agregando código nuevo a una compilación, por ejemplo, en un generador de origen.

La lista completa en https://learn.microsoft.com/es-es/dotnet/csharp/whats-new/csharp-12

05

Análisis comparativo Net Framework, NET core/5/6/7/8.

GALAXY
TRAINING

Análisis comparativo Net Framework, NET core/5/6/7/8

Característica	NET Core/5/6/7/8	.Net Framework
Plataforma o Framework	Cuando hablamos de .NET se define como la plataforma en la que se basan frameworks como ASP.NET y la Plataforma Universal de Windows (UWP) y amplían las funcionalidades de la plataforma .NET.	.Net Framework es un framework de desarrollo completo. El framework proporciona todos los requisitos básicos para el desarrollo de aplicaciones como UI, conectividad DB, servicios, APIs, etc.
Código abierto	.NET es una plataforma de código abierto.	.Net Framework incluye ciertos componentes de código abierto.
Multiplataforma	Se basa en el concepto de "create once, run anywhere.r". Debido a que es multiplataforma, es compatible con una variedad de sistemas operativos, incluidos Windows, Linux y Mac OS.	.NET Framework solo es compatible con el sistema operativo Windows
Modelos de aplicación	El modelo de aplicación de .Net incluye aplicaciones universales, ASP.NET y Windows.	El modelo de aplicación de .NET Framework incluye WinForms, ASP.NET y WPF.
Instalación	.Net es multiplataforma, por lo que debe instalarse de forma independiente.	.NET Framework tiene un único paquete de instalación y entorno de ejecución para Windows.
Soporte de microservicios	**.** NET es compatible con microservicios. , NET permite una combinación de tecnologías que se pueden minimizar para cada microservicio.	.NET Framework no permite la construcción y despliegue (**es más compleja**) de microservicios en múltiples lenguajes.
Soporte de servicios REST	.NET no admite servicios WCF (Windows Communication Foundation). Siempre necesitaría crear una API de REST (pero tienes alternativas de la comunidad).	Cuando se trata de servicios WCF (Windows Communication Foundation), the.NET Framework es una opción fantástica. También funciona con servicios RESTful.
Rendimiento y escalabilidad	.NET proporciona una alta escalabilidad y rendimiento en comparación con .NET Framework debido a su arquitectura.	.NET Framework es menos escalable y proporciona un rendimiento bajo en comparación con .NET**.**

Análisis comparativo Net Framework, NET core/5/6/7/8

Característica	NET Core/5/6/7/8	.Net Framework
Seguridad	Características como la seguridad de acceso del código no están presentes en .NET, por lo que .NET Framework tiene el perímetro en ese caso.	**.** NET Framework tiene esta característica llamada seguridad de acceso a código.
Enfoque en dispositivos	**.** NET se enfoca en desarrollar aplicaciones en una variedad de dominios como juegos, dispositivos móviles, IoT, IA, etc.	**.** NET Framework está limitado al sistema operativo Windows.
Compatibilidad	.NET es compatible con varios sistemas operativos: Windows, Linux y Mac OS.	Por otro lado, .NET Framework solo es compatible con el sistema operativo Windows.
Desarrollo móvil	Las aplicaciones móviles son cada vez más importantes para las empresas. NET tiene cierta compatibilidad con aplicaciones móviles. Es compatible con Xamarin y otras plataformas de código abierto para aplicaciones móviles.	Por otro lado, .NET Framework no admite su desarrollo en absoluto, y eso es un problema.
Herramientas de CLI	Para todas las plataformas, .NET Core proporciona una CLI muy ligera (interfaz de línea de comandos). Siempre existe la opción de cambiar a un IDE.	.NET Framework es demasiado pesado para la CLI. algunos desarrolladores prefieren trabajar en CLI en lugar de en IDE.
Modelo de despliegue	Cuando se instala una nueva versión de NET, se actualiza en una computadora a la vez, lo que resulta en nuevos directorios / carpetas que se crean en el programa existente sin afectarlo. Como resultado, .NET proporciona un modelo de despliegue sólido y adaptable.	IDE En el caso de .NET Framework, cuando se publica la versión actualizada, primero se implementa solo en el IIS.
Empaquetado y entrega	NET se incluye como una colección de paquetes Nuggets.	Todas las bibliotecas de .NET Framework se empaquetan y se distribuyen juntas.

NET vs .NET Framework – ¿Cuál es mejor?

La respuesta a esta pregunta depende del requisito del proyecto y de lo que exige nuestro proyecto, así que aquí hay algunos puntos que debemos considerar para elegir lo mejor para nuestro proyecto fuera de .NET Framework y .NET Core.

Preferir o elegir NET si

- El proyecto exige una integración multiplataforma.
- El proyecto requiere el desarrollo de microservicios.
- El proyecto se basa en gran medida en CLI (interfaz de línea de comandos) ya que NET Core es adecuado para CLI.

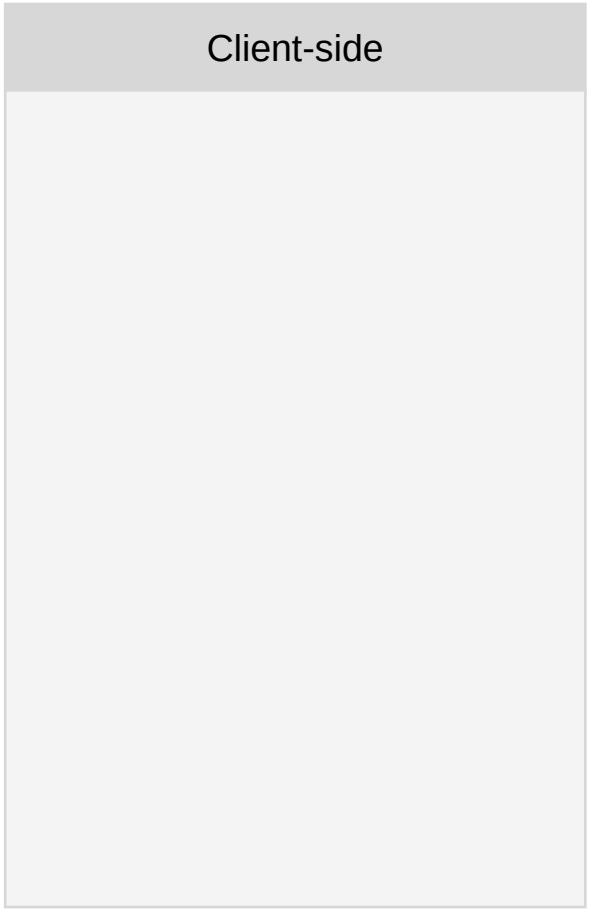
Preferir o elegir .Net Framework si

- Las aplicaciones ya se están ejecutando en .Net Framework.
- Las aplicaciones requieren tecnologías como flujo de trabajo, formularios web o WCF que no están presentes en NET
- Las aplicaciones están diseñadas para ejecutarse solo en Windows.

06

Introducción a Full-Stack Architecture con NET 8

GALAXY
TRAINING

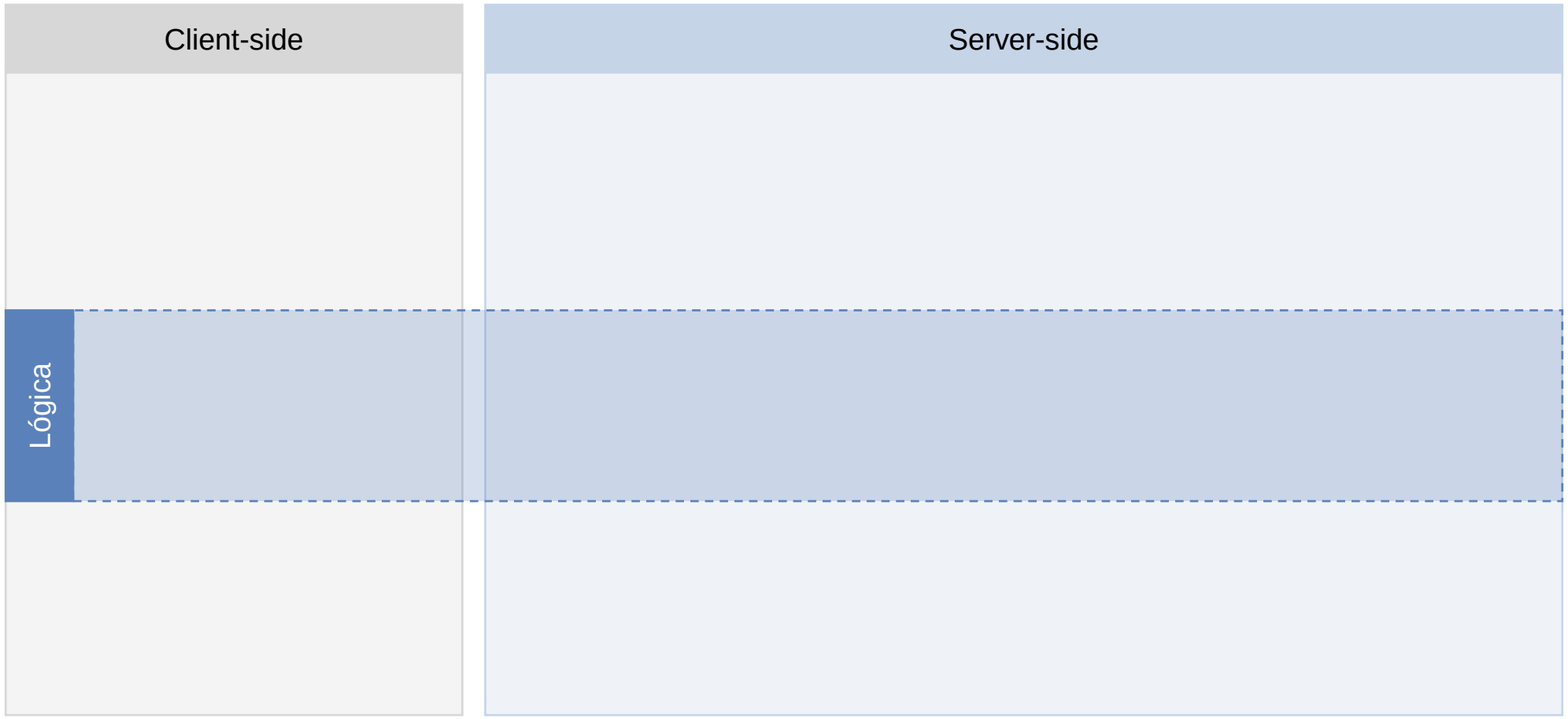

Client-side

Introducción a Full-Stack Architecture con NET 8

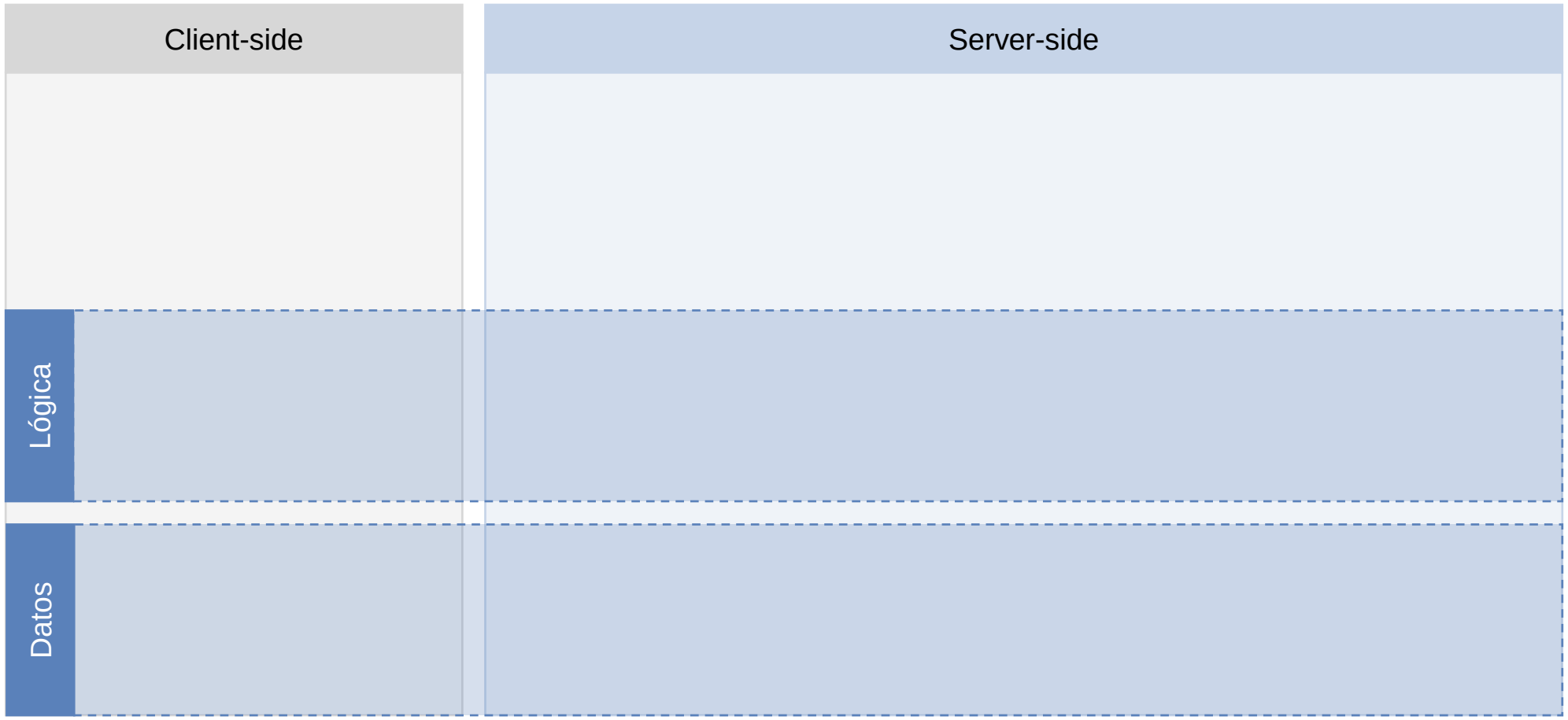
GALAXY TRAINING

Client-side	Server-side

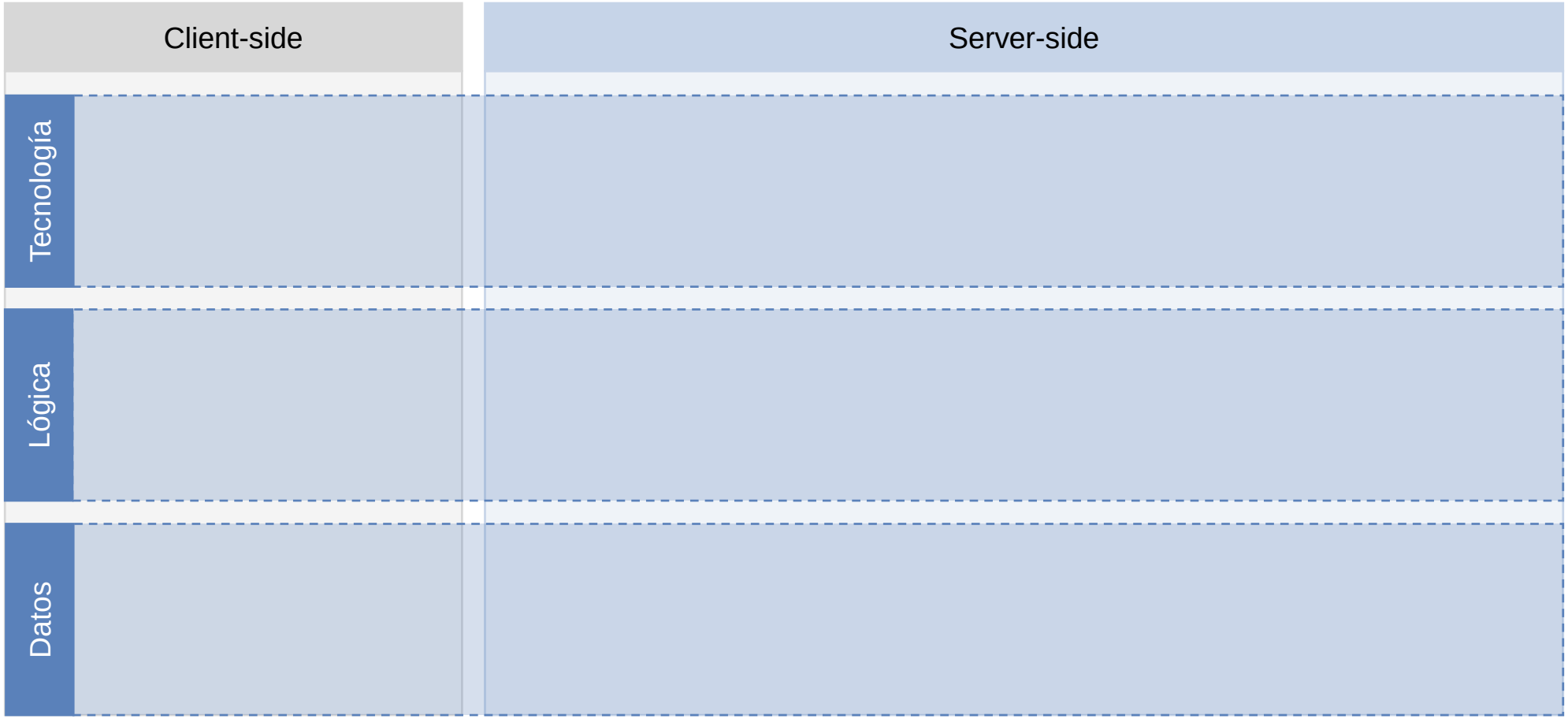
Introducción a Full-Stack Architecture con NET 8



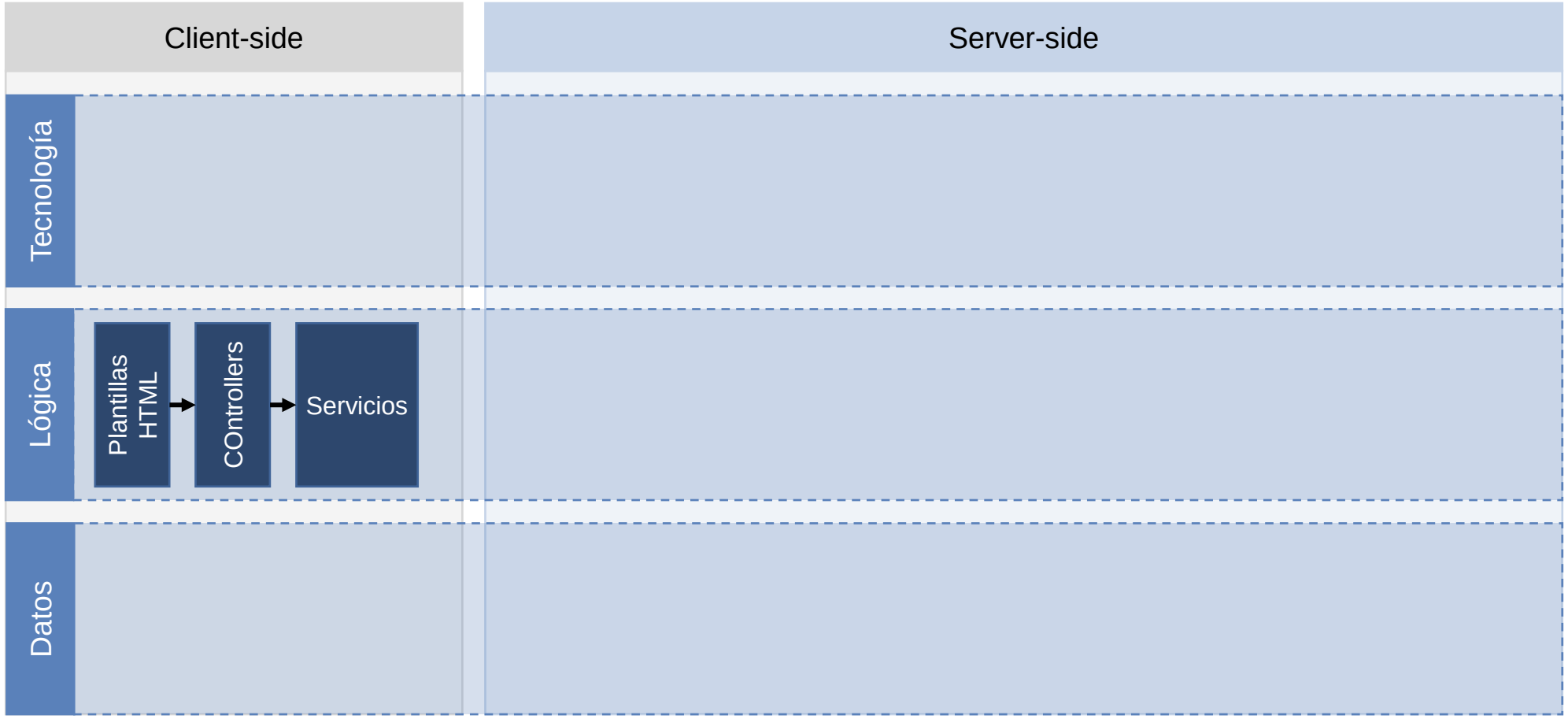
Introducción a Full-Stack Architecture con NET 8



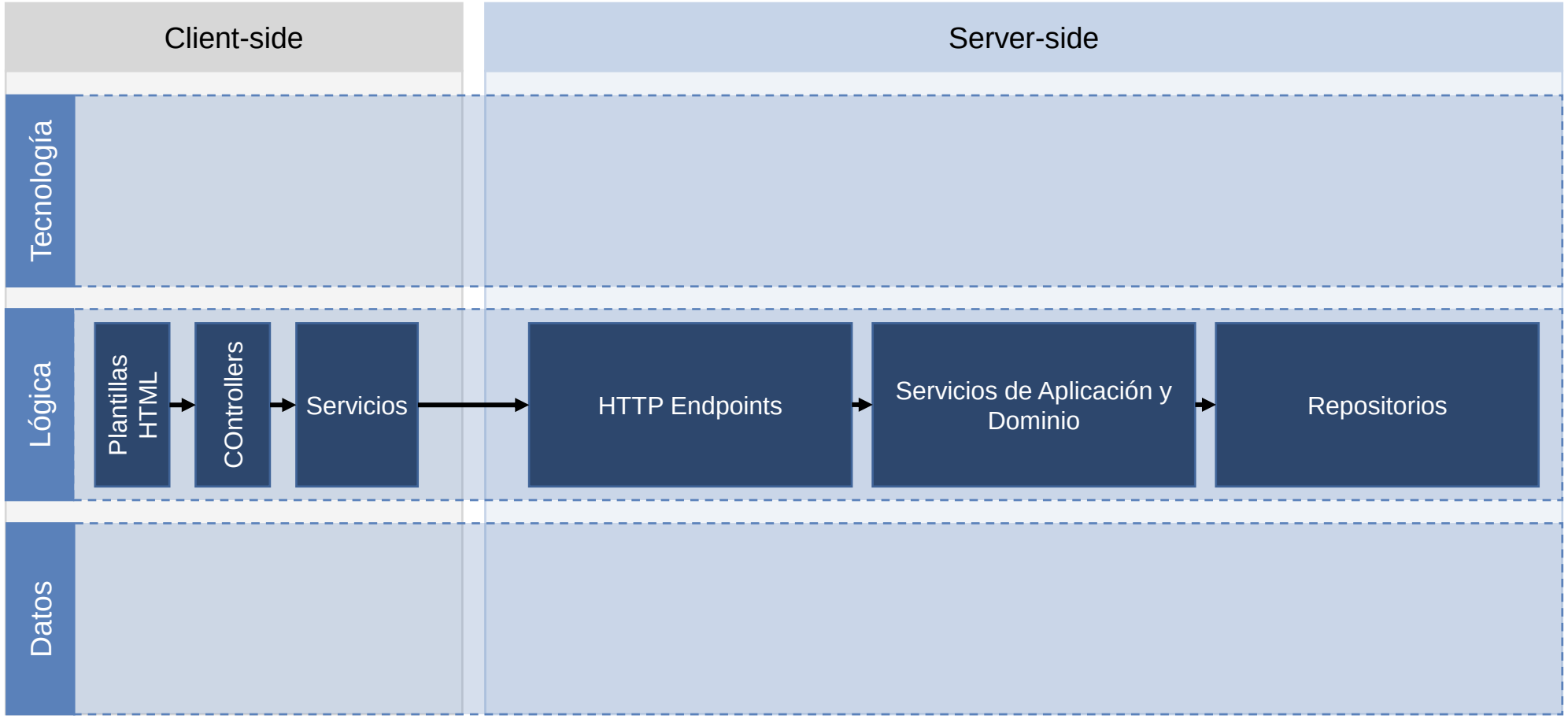
Introducción a Full-Stack Architecture con NET 8



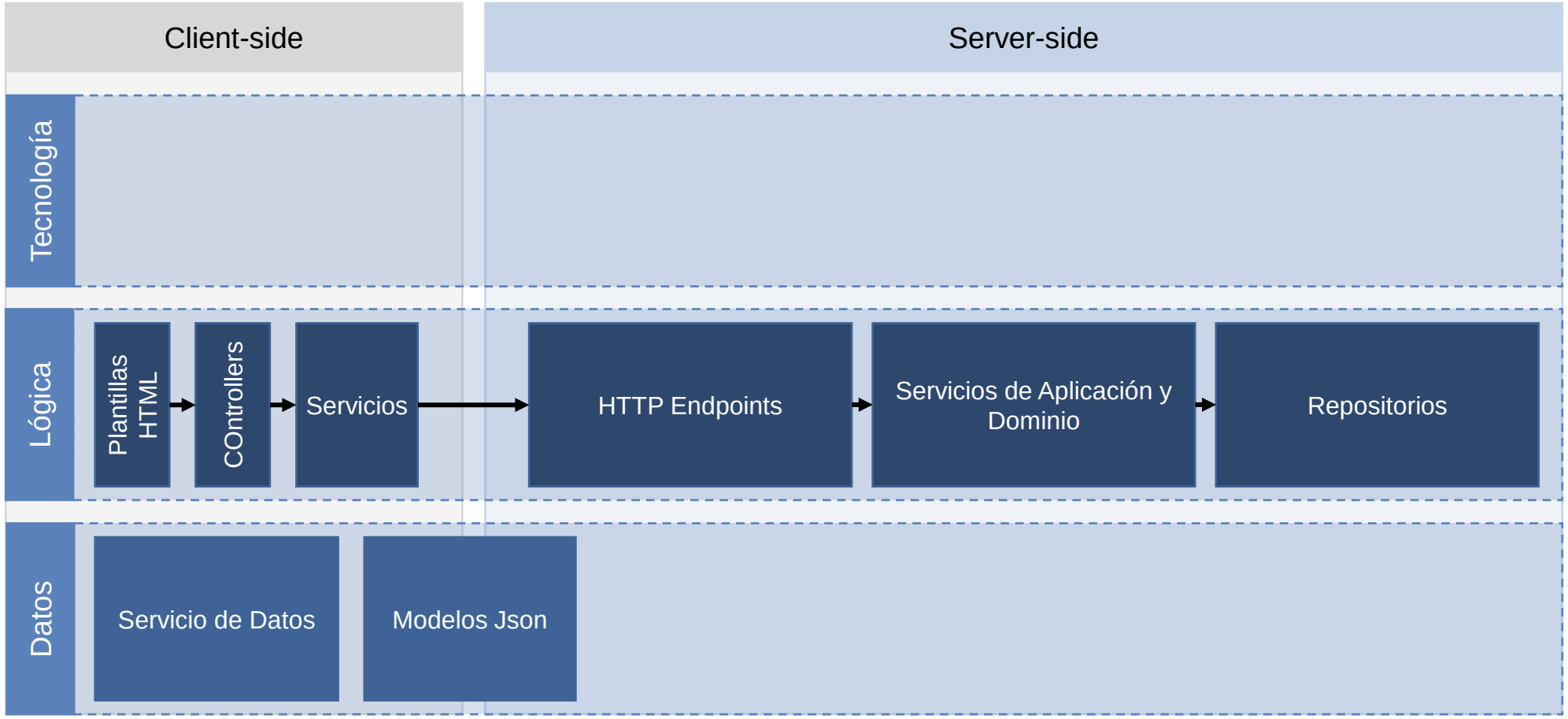
Introducción a Full-Stack Architecture con NET 8



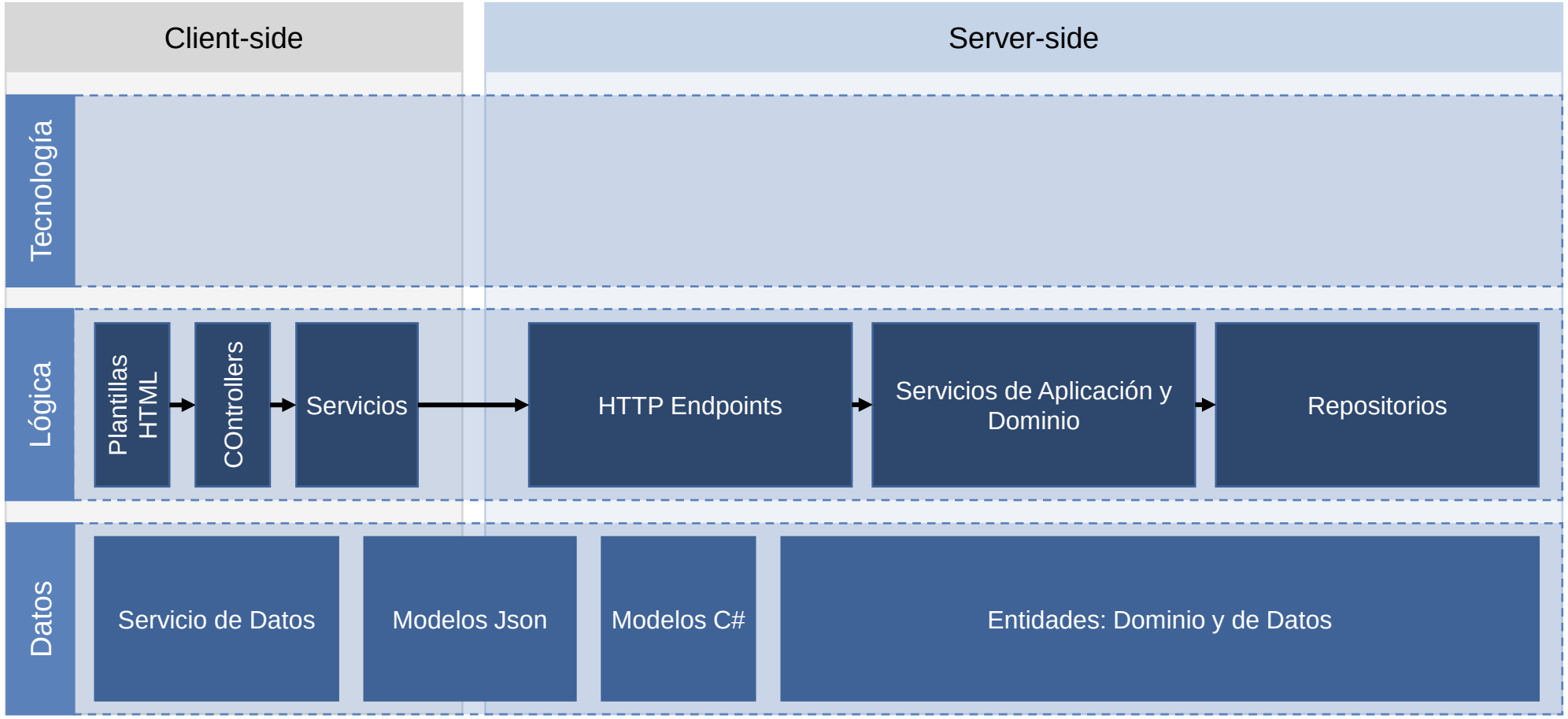
Introducción a Full-Stack Architecture con NET 8



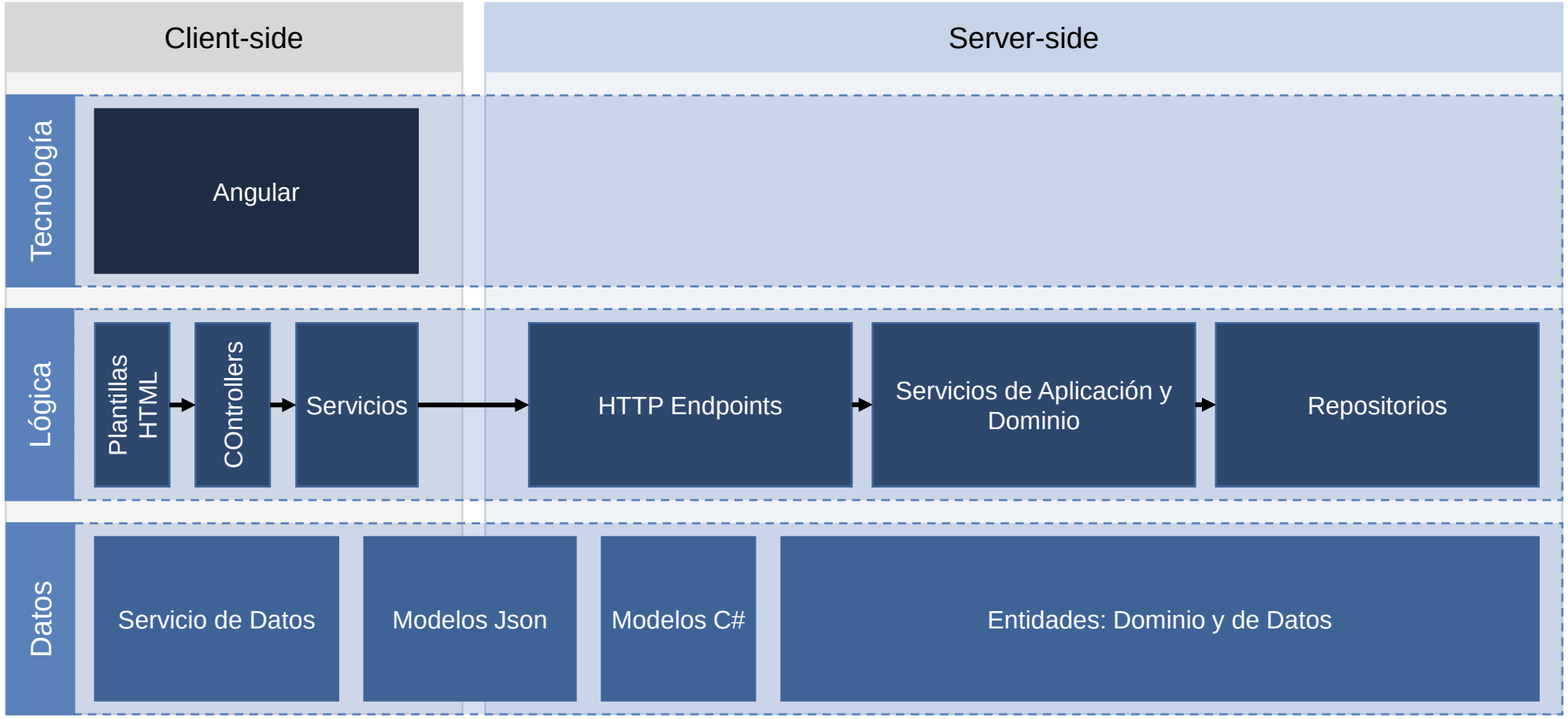
Introducción a Full-Stack Architecture con NET 8



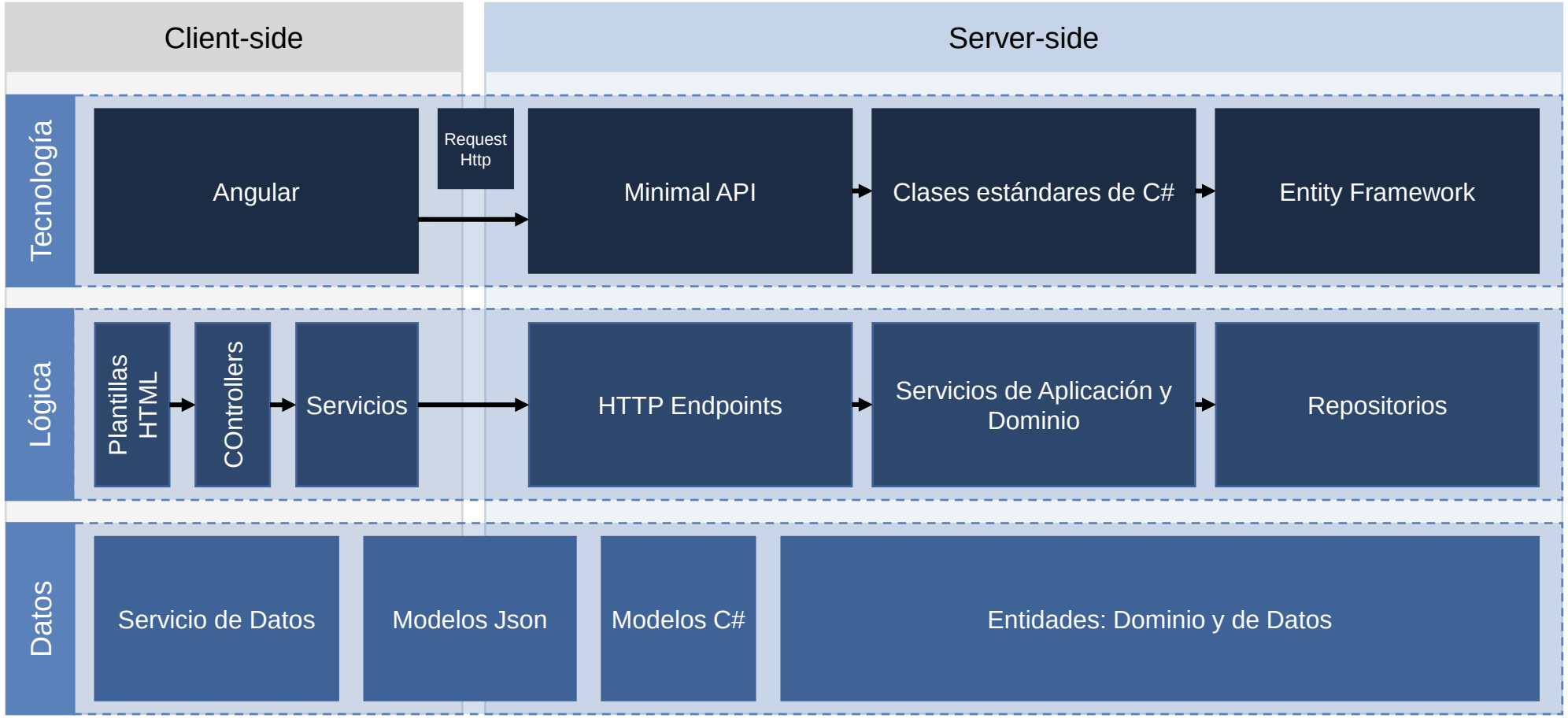
Introducción a Full-Stack Architecture con NET 8



Introducción a Full-Stack Architecture con NET 8



Introducción a Full-Stack Architecture con NET 8



GRACIAS

POR SU PREFERENCIA

GALAXY
TRAINING