La infraestructura de lenguaje común (en inglés *Common Language Infrastructure* o CLI) es una especificación estandarizada que describe un entorno virtual para la ejecución de aplicaciones, cuya principal característica es la de permitir que aplicaciones escritas en distintos lenguajes de alto nivel puedan luego ejecutarse en múltiples plataformas tanto de hardware como de software sin necesidad de reescribir o recompilar su código fuente.

Si bien el CLI tuvo sus orígenes en Microsoft (en principio se pensaba desarrollar un entorno de ejecución compartido para COM con el nombre de Common Object Runtime, que luego se extendió y generalizó para dar lugar a CLI), sus especificaciones fueron llevadas ante ECMA (European Computer Manufacturers Association), una importante organización europea de estándares, para su estandarización en el año 2000. Luego de un año de trabajo conjunto entre ECMA, Microsoft y otras empresas que co-patrocinaron el proceso (Intel, HP, IBM yFujitsu entre otras), el estándar ECMA-335 que define el entorno CLI finalmente vio la luz en diciembre de 2001. En abril del año 2003 ISO ratificó este estándar con el denominación ISO/IEC 23271:2003 .

Para comprender mejor la inclusión de cada una de las partes principales de la arquitectura de CLI es interesante analizar los objetivos de diseño que se plantearon desde su concepción. Según su especificación, la arquitectura de CLI debe:

- Permitir escribir componentes ínteroperables independientemente de la plataforma subyacente y del lenguaje de programación utilizado.
- Exponer todas las entidades programáticas a través de un único sistema unificado de tipos (en la especificación, este sistema es conocido como CTS, o Common Type System).
- Empaquetar todos los tipos en unidades completamente auto descriptivas y portables.
- Cargar los tipos de forma tal que se encuentren aislados unos de otros en tiempo de ejecución, pero que puedan a su vez compartir recursos.
- Resolver dependencias entre tipos en tiempo de ejecución usando una política flexible que pueda tener en cuenta la versión, atributos de localización y políticas administrativas.
- Ejecutar aplicaciones bajo la supervisión de un entorno privilegiado que permita

controlar y hacer cumplir políticas en tiempo de ejecución.
- Diseñar toda la infraestructura y servicios basándose en metadatos extensibles, de manera tal que toda la arquitectura pueda acomodarse con poco impacto a nuevas incorporaciones y cambios.
- Poder realizar tareas de bajo nivel, como carga de tipos en memoria, enlace con librerías y compilación a código nativo sólo cuando sea necesario (este enfoque se conoce típicamente como “on demand”, o “just in time”).
- Proveer una serie de funcionalidades comunes mediante un grupo de librerías de programación que los desarrolladores puedan utilizar para construir sus aplicaciones.

Microsoft .NET de hecho es un súper conjunto de esta especificación, es decir, provee todo lo necesario para cumplir con la misma y además agrega una serie de herramientas, librerías y funcionalidades no contempladas por ella originalmente y que proveen una enorme utilidad y flexibilidad a los desarrolladores (por ejemplo, librerías para la creación de aplicaciones y servicios web, acceso a motores de bases de datos, controles gráficos, herramientas para desensamblar assemblies, debuggers, etc.). Si bien es gratuito, su código fuente no es abierto, y es distribuido por Microsoft en versiones para sistemas operativos Windows 98 y sus sucesores únicamente.

La especificación del CLI está formada por cuatro partes:

- Sistema común de tipos, en inglés *Common Type System* (CTS).
- Metadatos.
- Especificaciones de lenguaje común, en inglés *Common Language Specification* (CLS).
- Sistema de ejecución virtual, del inglés *Virtual Execution System* (VES).