- Hardware: se refiere a todos los componentes físicos de una computadora, como la placa base, el microprocesador, la memoria, la pantalla o monitor, los periféricos, etc.
- Software: es el conjunto de código informático que utiliza el hardware para representar, almacenar y ejecutar lo que el usuario desee.
- Sistema operativo: es el software básico y mínimo que ha de estar instalado y configurado en un ordenador para que todos los demás programas puedan ejecutarse y funcionar. Es el que se encarga de manejar directamente el hardware al más bajo nivel.
- Software para el desarrollo de programación: es el conjunto de herramientas que nos permiten crear programas informáticos.
- Aplicaciones informáticas: son programas que tienen una utilidad funcional para el usuario final más o menos específica.
- Arquitectura de Von Neumann: es la arquitectura básica de los ordenadores modernos, que se compone de dispositivos de entrada, dispositivos de salida, CPU, memoria central y memoria secundaria.
- Dispositivos de entrada: son los dispositivos que permiten al usuario introducir datos en la computadora, como el teclado, el ratón, la pantalla táctil, los sensores de señales analógicas, etc.
- Dispositivos de salida: son los dispositivos que permiten a la computadora mostrar información al usuario, como la pantalla, la impresora, las señales analógicas, etc.
- CPU: es el procesador de la computadora, formado a su vez por la Unidad de Control y la Unidad Aritmético Lógica.
- Memoria central: es la memoria de acceso aleatorio (RAM) que se utiliza para almacenar temporalmente los datos y programas que se están utilizando en ese momento.
- Memoria secundaria: es la memoria que se utiliza para almacenar información de forma estable, como los pen drives, discos duros, discos sólidos o discos ópticos.
- Lenguajes de programación: son notaciones o conjuntos de símbolos y caracteres combinados entre sí de acuerdo con una sintaxis definida que se utilizan para programar una aplicación.
- Lenguajes de bajo nivel: son lenguajes que dependen absolutamente de la arquitectura de cada máquina, como los lenguajes máquina y los lenguajes ensamblador.
- Lenguajes de alto nivel: son lenguajes que tienen una gramática y una sintaxis similar a las palabras en una oración, como Fortran, Cobol, Basic, Algol, Pascal, C y ADA.
- Código fuente: es el conjunto de instrucciones ordenadas, entendibles y ejecutables por un ordenador, siguiendo una sintaxis no ambigua, que implementan un cierto algoritmo.
- Intérpretes: son programas que traducen y ejecutan el código fuente de un lenguaje de programación de forma directa, sin necesidad de compilarlo previamente.

- Compiladores: son programas que traducen el código fuente de un lenguaje de programación a un código en lenguaje máquina.
- Bytecode: es un código intermedio que se utiliza en algunos lenguajes de programación, como Java, para que el código sea independiente de la plataforma.
- Ciclo de vida del software: es el marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema, desde la definición de los requisitos, hasta la finalización de su uso.
- Metodología de desarrollo software: son colecciones de procedimientos y técnicas encaminadas a organizar el desarrollo de software mediante una serie de buenas prácticas, utilizando una serie de herramientas determinadas, articuladas según un modelo determinado de ciclo de vida, que genera una documentación de acuerdo a un formato estandarizado.
- Licencia de software: es un contrato que se establece entre el desarrollador de un software sometido a propiedad intelectual y a determinados derechos de autor, y el usuario, en el que se definen con precisión los derechos y deberes de ambas partes.
- Software libre: es aquel en el cual el autor cede una serie de derechos (denominados libertades) en el marco de una licencia.
- Software propietario: es aquel que se distribuye habitualmente en formato binario (o bytecode), sin posibilidad de acceso al código fuente y según una licencia en la cual el propietario prohíbe todas o algunas de las siguientes posibilidades: redistribución, modificación, copia, uso en varias máquinas simultáneamente, transferencia de la titularidad, difusión de fallos y errores que se pudieran descubrir en el programa, etc.
- Dominio público: es aquel software que no está sujeto a derechos de autor y puede ser utilizado, copiado, modificado y distribuido libremente por cualquier persona.
- Aplicaciones de escritorio: son los programas tradicionales que se instalan y configuran en un equipo y (en principio, aunque no siempre) no necesitan conexión a internet para funcionar.
- Aplicaciones web: son aplicaciones que no se instalan ni se ejecutan en el ordenador particular del usuario, sino en un servidor de internet, y el usuario accede a ella a través de un navegador.
- Aplicaciones móviles: son aplicaciones que se ejecutan en dispositivos móviles, como smartphones y tablets.
- Nativas: son aplicaciones móviles desarrolladas utilizando el lenguaje propio de cada plataforma.
- WebApp: son páginas web adaptadas para que se vean correctamente en dispositivos móviles y tratan de emular el funcionamiento de una App nativa.
- Híbridas: son una mezcla entre WebApp y App nativa y en función de la tecnología usada, pueden tener características más o menos avanzadas.

- Programa: es un conjunto de instrucciones que indican a una computadora qué hacer. Estas instrucciones están escritas en un lenguaje de programación y se traducen a código máquina para que la computadora pueda ejecutarlas.
- Algoritmo: es un conjunto de instrucciones que describen cómo resolver un problema o realizar una tarea. Los algoritmos se utilizan en programación para diseñar programas que resuelvan problemas específicos.
- Lenguajes de programación: son notaciones o conjuntos de símbolos y caracteres combinados entre sí de acuerdo con una sintaxis definida. Están diseñados para posibilitar la transmisión clara de instrucciones al hardware y suelen ser bastante exigentes y precisos. Los lenguajes de programación se utilizan para desarrollar o construir aplicaciones o programas.
- Lenguajes de bajo nivel: son lenguajes de programación que dependen absolutamente de la arquitectura de cada máquina. Los lenguajes de bajo nivel incluyen los lenguajes máquina y los lenguajes ensamblador.
- Lenguajes máquina: son lenguajes de programación que consisten enteramente en una secuencia de 0s y 1s que los controles de la computadora interpretan como instrucciones. Estos lenguajes son prácticamente ya no se utilizan, ni siquiera en electrónica.
- Lenguajes ensamblador: son lenguajes de programación que sustituyen cadenas binarias de operandos y códigos de operaciones por nombres simbólicos, que necesitan ser posteriormente traducidas a 0's y 1's por un software traductor específico, denominado también ensamblador.
- Lenguajes de alto nivel: son lenguajes de programación que tienen una gramática y una sintaxis similar a las palabras en una oración. Estos lenguajes son más fáciles de leer y escribir para los programadores que los lenguajes de bajo nivel. Ejemplos de lenguajes de alto nivel son Fortran, Cobol, Basic, Algol, Pascal, C, ADA, entre otros.
- Código fuente: es el conjunto de instrucciones escritas en un lenguaje de programación que componen un programa. El código fuente es el archivo que se utiliza para crear un programa ejecutable.
- Intérpretes: son programas que traducen un código fuente escrito en un lenguaje de alto nivel a lenguaje máquina, instrucción por instrucción, y ejecutan cada orden una vez que se traduce, sin almacenar el resultado de esa traducción. Ejemplos de lenguajes interpretados son JavaScript, PHP, Ruby, Python y Smalltalk.
- Compiladores: son programas que traducen un programa escrito en un lenguaje de alto nivel a un código en lenguaje máquina. El programa obtenido no es directamente ejecutable, pues es simplemente la traducción línea a línea del fuente y aún necesita de algún enlace y adecuación adicional. Ejemplos de lenguajes compilados son C, C++, Java, entre otros.
- Enlazador: es un programa que se encarga de unir los diferentes módulos de un programa y generar un archivo ejecutable. El enlazador se utiliza después de la compilación.

- Bytecode: es un código intermedio que se utiliza en algunos lenguajes de programación, como Java. El bytecode se traduce a código máquina en tiempo de ejecución por una máquina virtual.
- Primera generación: se refiere a los lenguajes de programación de los primeros ordenadores, conocidos como lenguajes máquina. Estos lenguajes consistían enteramente en una secuencia de 0s y 1s que los controles de la computadora interpretan como instrucciones.
- Segunda generación: se refiere a los primeros lenguajes de alto nivel que aparecieron en el mercado. Estos lenguajes tenían una gramática y una sintaxis similar a las palabras en una oración. Los primeros lenguajes de esta generación fueron Fortran, Cobol y Basic.
- Tercera generación: se refiere a los lenguajes de alto nivel estructurados que surgieron después de la llamada Primera crisis del software. Ejemplos de lenguajes estructurados de esta época son Algol, Pascal, C y ADA.
- Quinta generación: se refiere a los lenguajes para la implementación de algoritmos que intentan imitar el resultado del razonamiento humano. Son lenguajes orientados hacia la inteligencia artificial (AI – Artificial Intelligence) y el manejo de redes neuronales (Neural Networks).
- Análisis: es la primera etapa del ciclo de vida del software, en la que se identifican y definen los requisitos del usuario y se establecen las especificaciones del software.
- Diseño: es la segunda etapa del ciclo de vida del software, en la que se establece la arquitectura del software, la interfaz de usuario y los procedimientos.
- Codificación: es la tercera etapa del ciclo de vida del software, en la que se escribe el código fuente del software.
- Pruebas: es la cuarta etapa del ciclo de vida del software, en la que se verifica que el software cumpla con los requisitos del usuario y se corrigen los errores encontrados.
- Mantenimiento: es la última etapa del ciclo de vida del software, en la que se realizan las correcciones y mejoras necesarias para mantener el software en funcionamiento.
- Modelo en cascada: es un modelo de ciclo de vida del software en el que las etapas se realizan secuencialmente, de forma que cada etapa debe completarse antes de pasar a la siguiente.
- Modelo en cascada con realimentación: es una variante del modelo en cascada en la que se permite la retroalimentación entre las etapas, de forma que se pueden realizar correcciones y mejoras en etapas anteriores.
- Modelo iterativo e incremental: es un modelo de ciclo de vida del software en el que las etapas se realizan de forma iterativa e incremental, de forma que se van realizando mejoras y correcciones en cada iteración.
- Modelo en espiral: es un modelo de ciclo de vida del software en el que las etapas se realizan de forma iterativa y se van ajustando en función de los resultados obtenidos en cada iteración.

- Modelo Agile: es un modelo de ciclo de vida del software que se basa en la entrega continua de software funcional y en la colaboración entre el equipo de desarrollo y el cliente.

- Ciclo de vida del software: es el marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema, desde la definición de los requisitos, hasta la finalización de su uso. Comprende, por tanto, el periodo que transcurre desde que el producto es concebido hasta que deja de estar disponible o es retirado. Las etapas que suelen considerarse siempre son: análisis de los requerimientos de los usuarios, diseño de las estructuras de datos, la arquitectura del software, la interfaz de usuario y los procedimientos, codificación, pruebas y mantenimiento.
- Metodología de desarrollo software: es una colección de procedimientos y técnicas encaminadas a organizar el desarrollo de software mediante una serie de buenas prácticas, utilizando una serie de herramientas determinadas, articuladas según un modelo determinado de ciclo de vida, que genera una documentación de acuerdo a un formato estandarizado. Algunas de las principales metodologías de desarrollo de software son: cascada o lineal, iterativo e incremental, espiral y Agile.
- Scrum: es una metodología ágil de gestión de proyectos de software que se centra en la entrega temprana de un producto de calidad. Se basa en la colaboración, el trabajo en equipo y la comunicación constante entre los miembros del equipo. El proceso de Scrum se divide en sprints, que son ciclos de trabajo de una duración fija en los que se desarrolla una parte del producto final.
- Extreme Programming: es una metodología ágil de desarrollo de software que se centra en la calidad del código y la satisfacción del cliente. Se basa en la comunicación constante entre los miembros del equipo, la simplicidad en el diseño y la programación, y la retroalimentación continua. Extreme Programming se enfoca en la entrega temprana y frecuente de software funcional.
- Métrica v3: es una metodología de gestión de proyectos de software desarrollada por el Ministerio de Administraciones Públicas de España. Se basa en un conjunto de buenas prácticas y técnicas para la gestión de proyectos de software, que incluyen la planificación, el seguimiento y control, la gestión de riesgos, la calidad y la gestión de la configuración.
- UML: es un lenguaje de modelado visual utilizado para representar gráficamente sistemas de software. UML se utiliza para describir, especificar, diseñar y documentar sistemas de software. UML consta de varios tipos de diagramas, como diagramas de casos de uso, diagramas de clases, diagramas de secuencia, entre otros.

- Licencia de software: es un contrato que se establece entre el desarrollador de un software sometido a propiedad intelectual y a determinados derechos de autor, y el usuario, en el que se definen con precisión los derechos y deberes de ambas partes. Las licencias de software pueden ser libres o propietarias.
- Software libre: es aquel en el cual el autor cede una serie de derechos (denominados libertades) en el marco de una licencia. Estas libertades incluyen la libertad de utilizar el programa con cualquier fin en cuantos ordenadores se desee, la libertad de estudiar cómo funciona el programa y de adaptar su código a necesidades específicas, la libertad de distribuir copias a otros usuarios (con o sin modificaciones) y la libertad de mejorar el programa (ampliarlo, añadir funciones) y de hacer públicas y distribuir al público las modificaciones. Para ello, como condición previa, es necesario poder acceder al código fuente.
- Software propietario: es aquel que se distribuye habitualmente en formato binario (o bytecode), sin posibilidad de acceso al código fuente y según una licencia en la cual el propietario prohíbe todas o algunas de las siguientes posibilidades: redistribución, modificación, copia, uso en varias máquinas simultáneamente, transferencia de la titularidad, difusión de fallos y errores que se pudieran descubrir en el programa, etc.
- GPL: es una licencia de software libre que garantiza a los usuarios la libertad de utilizar, estudiar, modificar y distribuir el software. La GPL se aplica a muchos programas de software libre, incluyendo el sistema operativo Linux.
- Dominio público: es aquel que no está sujeto a derechos de autor y puede ser utilizado, copiado, modificado y distribuido libremente por cualquier persona.
- Creative Commons: es una organización sin fines de lucro que ofrece licencias de derechos de autor flexibles para obras creativas. Las licencias Creative Commons permiten a los autores mantener sus derechos de autor mientras permiten a otros copiar, distribuir y utilizar sus obras de manera gratuita, siempre y cuando se cumplan ciertas condiciones.
- Máquina virtual: es un software que simula una computadora dentro de otra computadora. La máquina virtual permite ejecutar múltiples sistemas operativos en una misma computadora física.
- Hipervisor: es un software que permite la creación y gestión de máquinas virtuales. El hipervisor se encarga de asignar los recursos de hardware de la computadora física a las máquinas virtuales.
- Entorno de ejecución: es un software que proporciona un entorno en el que se puede ejecutar un programa de software. El entorno de ejecución incluye el sistema operativo, las bibliotecas de software y otros componentes necesarios para que el programa se ejecute correctamente.

- Framework: es un conjunto de herramientas, bibliotecas y componentes de software que se utilizan para desarrollar aplicaciones de software. El framework proporciona una estructura y una base para el desarrollo de software, lo que permite a los desarrolladores centrarse en la lógica de la aplicación en lugar de en los detalles de implementación.
- Computación en la nube: es un modelo de entrega de servicios de computación a través de Internet. La computación en la nube permite a los usuarios acceder a recursos informáticos, como servidores, almacenamiento y aplicaciones, según sea necesario y sin tener que invertir en infraestructura de TI propia.
- SaaS: es un modelo de entrega de software en el que el software se proporciona como un servicio a través de Internet. Los usuarios acceden al software a través de un navegador web y pagan por el uso del software en función del número de usuarios o de la cantidad de tiempo que lo utilizan.
- PaaS (Platform as a Service): es un modelo de servicio en la nube que proporciona una plataforma de desarrollo y despliegue de aplicaciones. PaaS permite a los desarrolladores crear aplicaciones sin tener que preocuparse por la infraestructura subyacente, ya que la plataforma proporciona todo lo necesario para ejecutar y escalar la aplicación. Ejemplos de proveedores de PaaS son Google App Engine, Microsoft Azure y Heroku.
- IaaS (Infrastructure as a Service): es un modelo de servicio en la nube que proporciona infraestructura de TI, como servidores, almacenamiento y redes, a través de Internet. IaaS permite a las empresas alquilar recursos de TI en lugar de comprar y mantener su propia infraestructura. Ejemplos de proveedores de IaaS son Amazon Web Services, Microsoft Azure y Google Cloud Platform.
- DevOps (Development-Operation): es una filosofía de trabajo que busca la colaboración y coordinación continua entre los equipos de desarrollo y operaciones de TI, con el objetivo de acelerar el ciclo de vida del software y mejorar la calidad del producto. DevOps se centra en la automatización de procesos, la integración continua y la entrega continua, lo que permite a los equipos de desarrollo y operaciones trabajar juntos de manera más eficiente y efectiva.
-