Tipos de mantenimiento

Mantenimiento preventivo. Consiste en la revisión constante del *software* para detectar posibles focos de problemas que puedan surgir en el futuro.

Mantenimiento predictivo. Evalúa el flujo de ejecución del programa para predecir con certeza el momento en el que se producirá la falla, y así determinar cuándo es adecuado realizar los ajustes correspondientes.

Mantenimiento correctivo. Corrige los defectos encontrados en el *software*, y que originan un comportamiento distinto al deseado. Estas fallas pueden ser de procesamiento, rendimiento (por ejemplo, uso ineficiente de los recursos de *hardware*), programación (inconsistencias en la ejecución), seguridad o estabilidad, entre otras.

Mantenimiento adaptativo. Si se requiere cambiar el entorno de uso de la aplicación (que incluye al sistema operativo, a la plataforma de *hardware* o, en el caso de las aplicaciones web, al navegador), puede ser indispensable modificarla para mantener su plena funcionalidad en estas nuevas condiciones.

Mantenimiento evolutivo. Es un caso especial donde la adaptación resulta prácticamente obligatoria, ya que de lo contrario el programa quedaría obsoleto con el paso del tiempo. Por ejemplo, el cambio de versión en un navegador (muchas veces impuesto sin el consentimiento del usuario) suele obligar a realizar ajustes en *plugins* y aplicaciones web.

Mantenimiento perfectivo. Por distintas razones, el usuario puede solicitar el agregado de nuevas funcionalidades o características no contempladas al momento de la implementación del *software*. El mantenimiento perfectivo adapta la aplicación a este requerimiento.

Kanban

Kanban es originario de Japón. Se trata de un sistema desarrollado por Toyota en el año 1947. Esto explica su nombre, que está compuesto por las dos palabras japonesas kan y ban y cuya traducción aproximada es «tarjeta visual».

El objetivo de la implantación de Kanban consiste en mejorar el flujo de trabajo de un equipo, aumentando al mismo tiempo la productividad y la calidad del producto final. Kanban se engloba dentro de la denominada «metodología ágil» y, como tal, otorga una gran flexibilidad a los procesos de trabajo. Las tareas se dividen en pequeñas fases que se realizan de forma consecutiva y en base al siguiente lema: «**Stop starting – start finishing**».

Case

Case es un conjunto de herramientas que contiene programas y aplicaciones informáticas diseñados con la finalidad de generar mayor productividad, brindar facilidades de uso que ahorran tiempo y dinero en el desarrollo de softwares o nuevas aplicaciones.

La palabra CASE es el resultado de las siglas en inglés *Computer Aided Software Engineering*, que en español quiere decir ingeniería de software asistida por ordenador.

Herramientas Case

Las herramientas CASE fueron diseñadas tanto para desarrollar softwares con bajos costos de producción y que impliquen menos tiempo de trabajo, como con el propósito de extender el ciclo de utilidad del software creado a través de este medio, por el cual se puede diseñar un proyecto con un costo de producción determinado y a su vez agilizar el proceso de programación. las herramientas CASE se pueden elaborar, en un mismo proceso y con una misma herramienta, el código fuente (lenguaje de alto nivel, que interpreta el usuario), la compilación de datos (análisis, detección y documentación de errores), y posteriormente un código objeto (lenguaje de bajo nivel, que interpreta el equipo).

UML

UML son las siglas de “Unified Modeling Language” o “Lenguaje Unificado de Modelado”. Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software (programas informáticos).

UML es un lenguaje para hacer modelos y es independiente de los métodos de análisis y diseño. Existen diferencias importantes entre un método y un lenguaje de modelado. Un método es una manera explícita de estructurar el pensamiento y las acciones de cada individuo. Además, el método le dice al usuario qué hacer, cómo hacerlo, cuándo hacerlo y por qué hacerlo; mientras que el lenguaje de modelado carece de estas instrucciones. Los métodos contienen modelos y esos modelos son utilizados para describir algo y comunicar los resultados del uso del método.