1 Ciclo de vida del software

También conocido como (SDLC o Systems Development Life Cycle), es el proceso que se sigue para construir y hacer evolucionar un determinado software. El ciclo de vida permite iniciar una serie de fases mediante las cuales se procede a la validación y al desarrollo del software garantizando que se cumplan los requisitos para la aplicación y verificación de los procedimientos de desarrollo; para ello, se utilizan métodos del ciclo del software, que indican distintos pasos a seguir para el desarrollo de un producto.

Si bien existen diferentes ciclos de desarrollo de software, la normativa ISO/IEC/IEEE 12207:2017 establece:

Un marco común para los procesos del ciclo de vida de los programas informáticos, con una terminología bien definida, a la que pueda remitirse la industria del software. **Contiene procesos, actividades y tareas aplicables durante la adquisición, el suministro, el desarrollo, el funcionamiento, el mantenimiento o la eliminación de sistemas, productos y servicios informáticos**. Estos procesos del ciclo de vida se llevan a cabo mediante la participación de los interesados, con el objetivo final de lograr la satisfacción del cliente (s.p.).

A continuación, se indican cuáles son los elementos que integran un ciclo de vida:

Regresar Continuar

Fases:

Una fase es un conjunto de actividades relacionadas con un objetivo en el desarrollo del proyecto. Se construye agrupando tareas (actividades elementales) que pueden compartir un tramo determinado del tiempo de vida de un proyecto. La agrupación temporal de tareas impone requisitos temporales correspondientes a la asignación de recursos (humanos, financieros o materiales).

Entregables:

Son los productos intermedios que generan las fases. Pueden ser materiales o inmateriales (documentos, software). Los entregables permiten evaluar la marcha del proyecto mediante comprobaciones de su adecuación o no a los requisitos funcionales y de condiciones de realización previamente establecido.

1.1 Fases

Las fases del modelo de ciclo del software son: planificación, análisis, diseño, implementación, pruebas y mantenimiento, las cuales se describen en la siguiente figura.



FASE PLANIFICACIÓN:

Regresar Continuar

En esta primera fase se realiza el planteamiento del problema, se definen alcances y objetivos del software.

Objetivos: estudio de viabilidad, realizar planificación detallada.

1/5 ›

1.2. Paradigmas de los modelos de ciclo de vida del software

Con la finalidad de proporcionar una metodología común entre el cliente y la empresa de software, se utilizan los modelos de ciclos de vida o paradigmas de desarrollo de software para plasmar las etapas y la documentación necesaria, de manera que cada fase se valide antes de continuar con la siguiente.

Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software e intenta determinar el orden de las etapas involucradas y los criterios de transición asociados entre estas.

Según la norma 1074 IEEE se define al ciclo de vida del software como:

Una aproximación lógica a la adquisición, el suministro, el desarrollo, la explotación y el mantenimiento del software

La ISO/IEC 12207 Information Technology / Software Life Cycle Processes señala que es:

Un 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 (2008).

Existen modelos preestablecidos con los cuales se pude elaborar un proyecto; a continuación, se mencionan los diferentes paradigmas de modelos de ciclo de vida para desarrollar software.

Paradigma tradicional

‹ Regresar Continuar ›



Los paradigmas tradicionales se identifican, fundamentalmente, por ser lineales, es decir se trata de completar cada proceso de principio a fin hasta que quede listo para avanzar a la segunda fase del ciclo del software.

Si bien es verdad que las metodologías actuales están basadas en lo que fueron los paradigmas tradicionales, hoy en día se ha evolucionado, sin embargo, los paradigmas tradicionales se mantienen.



Desventaja:

Pérdida de tiempo si se encuentran errores en una fase avanzada porque al devolverse se debe pasar nuevamente por todas las fases y reestructurar de acuerdo con las modificaciones.



Paradigma orientado a objetos:

Las etapas de desarrollo de software en el paradigma orientado a objetos, se conforma principalmente por la creación de clases, análisis de requisitos y el diseño. Con este paradigma se pretende que el código fuente sea reutilizable para otros proyectos.

Paradigma de desarrollo ágil

El objetivo de este paradigma es el desarrollo de proyectos en poco tiempo, se simplifican procesos tediosos, se agilizan las fases del desarrollo y las interacciones se hacen en corto tiempo.



Una de las principales diferencias con los paradigmas anteriores es que el cliente se ve involucrado en el proyecto durante el desarrollo de este, así, el cliente sugiere mejoras, propone ideas y se mantiene al tanto del desarrollo del producto, a diferencia del

‹ Regresar Continuar ›

paradigma tradicional y el orientado objeto donde el cliente únicamente está al principio.

A continuación, se revisan los modelos del paradigma tradicional más utilizados.

− Modelo en cascada

Uno de los primeros modelos de ciclo de vida del desarrollo fue establecido por W. Royce en 1970 y es conocido como el “modelo de cascada” (waterfall model).

En su concepción básica, cada una de las actividades genera, como salidas, productos y modelos que son utilizados como entradas para el proceso subsiguiente; esto supone que una actividad debe terminarse (por lo menos, en algún grado) para empezar la siguiente.

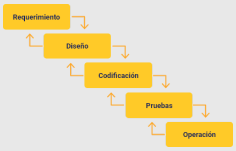


Figura 1. Modelo de cascada del ciclo de vida del desarrollo.

+ Modelo espiral

+ Modelo iterativo o por prototipos

Con respecto a los modelos del ciclo de vida del paradigma ágil, estos se caracterizan por estar basados en etapas del ciclo de vida del software tradicional, pero combinándolas con algunas técnicas, al respecto se pueden revisar los siguientes:

− Modelo Scrum

Este modelo se basa en el desarrollo incremental, es decir conforme pasen las fases y la iteración mayor será el tamaño del proyecto que se está desarrollando.

Los procesos que utiliza son:

1. Product Backlog.
2. Sprint Backlog.
3. Sprint Planning Meeting.
4. Daily Scrum.
5. Sprint Review.
6. Sprint Retrospective

El Scrum consiste en realizar un análisis de los requerimientos del sistema (Product Backlog), señalar cuáles serán los objetivos a corto o mediano plazo dentro de un sprint, o sea, la fase de desarrollo. Posteriormente, los desarrolladores harán lo suyo, se realizarán algunas pruebas y se retroalimentará de acuerdo con lo conseguido al terminar la última fase.

Ventajas

• **Gestión regular de las expectativas del usuario:** los usuarios participan y proponen soluciones.

• **Resultados anticipados:** no es necesario esperar hasta el final para ver resultados.

• **Flexibilidad y adaptación:** se adapta a cualquier contexto, área o sector.

• **Gestión sistemática de riesgos:** los problemas son gestionados en el mismo momento de su aparición.

+ Modelo Kanban

+ Modelo XP o programación extrema

‹ Regresar Continuar ›