1. TEMA 086. EL CICLO DE VIDA DE LOS SISTEMAS DE INFORMACIÓN. MODELOS DEL CICLO DE VIDA

1.1.EL CICLO DE VIDA DE LOS SISTEMAS DE INFORMACIÓN

Ciclo de vida: etapas por las que pasa el software desde que un proyecto es concebido hasta que se deja de usar.

Ciclo de desarrollo: tiempo desde la concepción del proyecto hasta su instalación una vez desarrollado.

CICLO DE VIDA = CICLO DE DESARROLLO + MANTENIMIENTO.

El ciclo de vida indica QUÉ actividades se realizarán y en qué orden, pero es la **metodología** la que indica CÓMO llevar a cabo esas actividades.

1.2.MODELOS DE CICLO DE VIDA

1.2.1. CICLO DE VIDA DE CODIFICACIÓN DIRECTA (CODE AND FIX)

Consiste en no aplicar ningún modelo de ciclo de vida. Se comienza a codificar inmediatamente, se prueba para detectar errores y se van corrigiendo sobre la marcha según aparecen. No existe documentación y supone grandes costes de mantenimiento a largo plazo.

1.2.2. CICLO DE VIDA EN CASCADA (WATERFALL)

Concebido en los años 70 por Winston Royce.

Consta de una serie de fases sucesivas (“fases en cascada”), cada una de las cuales se completa antes de avanzar a la siguiente

Ventajas:

- Marca pautas de trabajo muy claras.
- Facilita la estimación y el seguimiento del progreso.
- Proporciona productos entregables intermedios que forman el conjunto final del producto

Inconvenientes:

- Se comienza estableciendo TODOS los requisitos, algo difícil incluso para el propio usuario.
- Rigidez
- Los problemas se detectan tarde, incluso aunque se incluyan actividades de verificación y validación.

1.2.3. CICLO DE VIDA EN V

Es similar al ciclo de vida en cascada, pero se caracteriza por tener dos tiempos claramente marcados:

- Actividades de **desarrollo** (**rama descendente**).
- Actividades de **prueba** (**rama ascendente**).

Ventajas:

- Las del ciclo de vida en cascada.
- Favorece la consideración de las pruebas lo antes posible.

Inconvenientes:

- Rigidez, aunque menos que en el modelo en cascada.

1.2.4. MODELO DE CONSTRUCCIÓN DE PROTOTIPOS

3 fases:

- **Escuchar al usuario**: recolección de requisitos.
- **Construir y revisar el prototipo**
- **El usuario prueba el prototipo**

Se aplican varios ciclos de prototipado (cada uno con sus 3 fases)

Ventajas:

- Ideal para obtener requisitos cuando no están claros.

Inconvenientes:

- El usuario puede creer que el trabajo ya está hecho y que en breve dispondrá de un sistema funcional.
- Arrastrar al sistema real decisiones de implementación tomadas para agilizar el desarrollo del prototipo.

1.2.5. CICLO DE VIDA RAD (RAPID APPLICATION DEVELOPMENT) O DRA (DESARROLLO RÁPIDO DE APLICACIONES)

Desarrollado inicialmente por James Martin en 1980. Comprende desarrollo iterativo, construcción de prototipos y el uso de herramientas CASE.

DRA es una adaptación a "alta velocidad" del modelo en cascada. Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso DRA permite crear un "sistema completamente funcional" dentro de periodos cortos de tiempo (usualmente 60-90 días).

Si una aplicación se puede modular de forma que cada función principal se pueda completar en menos de 3 meses, es un candidato del DRA. Cada una de las funciones puede ser afrontada por un equipo DRA diferente y ser integradas en un solo conjunto.

Ventajas:

- Reutilización.
- Paralelismo en el desarrollo.
- Adecuado para tecnología orientada a objetos (basada en componentes).

Inconvenientes:

- Para proyectos grandes, el DRA requiere recursos humanos suficientes para crear los equipos DRA correctos.
- Requiere compromiso (de clientes y desarrolladores)
- No es adecuado para todo tipo de proyectos

1.2.6. MODELO INCREMENTAL

Se construye una implementación parcial del sistema, para posteriormente ir aumentando la funcionalidad con diferentes versiones. Para llevar a cabo los incrementos se pueden aplicar reiteradamente varias secuencias basadas en el modelo en cascada.

Ventajas:

- Más fácil determinar si los requerimientos planeados son correctos.
- Los errores se detectan antes.
- Mayor flexibilidad.

Inconvenientes:

- Dificultad para definir el núcleo que formará parte del primer incremento.
- Dificultad en la definición de los incrementos.
- Las soluciones de incrementos anteriores pueden no ser válidas para incrementos posteriores.

1.2.7. MODELO EVOLUTIVO

Se caracteriza por permitir desarrollar versiones cada vez más completas del software. Es útil para introducir en el mercado una versión limitada del software que permita cumplir la presión competitiva.

Es posible combinar los modelos evolutivos con los incrementales.

1.2.8. MODELO EN ESPIRAL

Consiste en aplicar un conjunto sucesivo de ciclos (vueltas en la espiral) hasta completar el software.

Se definen una serie de pasos para cada ciclo o vuelta de la espiral. La forma más habitual es tener un ciclo dividido en los siguientes 4 pasos:

- Determinación de objetivos, alternativas y restricciones.
- Evaluación de alternativas, considerando el análisis de riesgos.
- Desarrollo del siguiente nivel del producto.
- Planificación del siguiente ciclo.

Durante cada fase se puede seleccionar el ciclo de desarrollo que se desee: cascada, V etc.

Ventajas:

- Permite adaptar el proceso de desarrollo a las necesidades cambiantes del proyecto y al conocimiento que se va adquiriendo.
- Permite el manejo de prototipos, enlazándolo con el análisis de riesgos.
- Gestiona los riesgos de forma explícita.

Inconvenientes:

- La evaluación de riesgos puede disparar el coste.
- Complejidad del modelo.
- Incertidumbre en el número de iteraciones necesarias.
- Algunos autores consideran que no es operativo en la práctica.