

Evolución y Mantenimiento del Software



Evolución y/o Mantenimiento

- **El desarrollo del software no se detiene cuando un sistema se entrega, continúa a lo largo de toda su vida.**
- **Después de distribuir un sistema, inevitablemente debe modificarse con la finalidad de mantenerlo útil.**



- Cambios empresariales
- Cambios en las necesidades del usuario
- Corrección de errores
- Adaptarlo a cambios de software o hardware
- Mejora de rendimiento ...

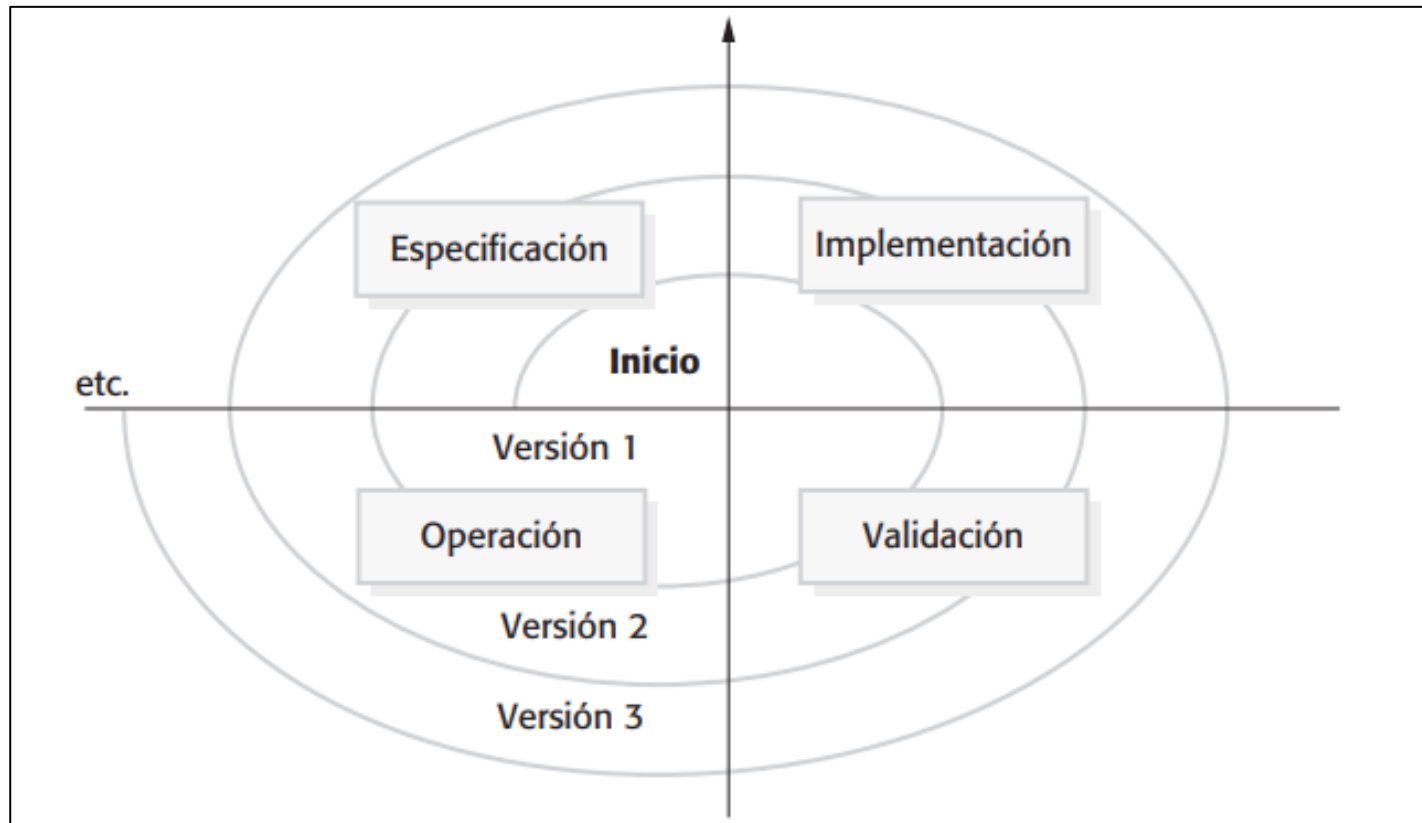


Evolución o Mantenimiento?

- **Evolución del software:** una sola organización es responsable tanto del desarrollo inicial del software como de su evolución posterior.
- **Mantenimiento del software:** el desarrollo inicial del software y su evolución posterior es realizado por organizaciones diferentes.



El proceso de desarrollo de software se debe considerar como un proceso en espiral a lo largo de la vida del sistema.



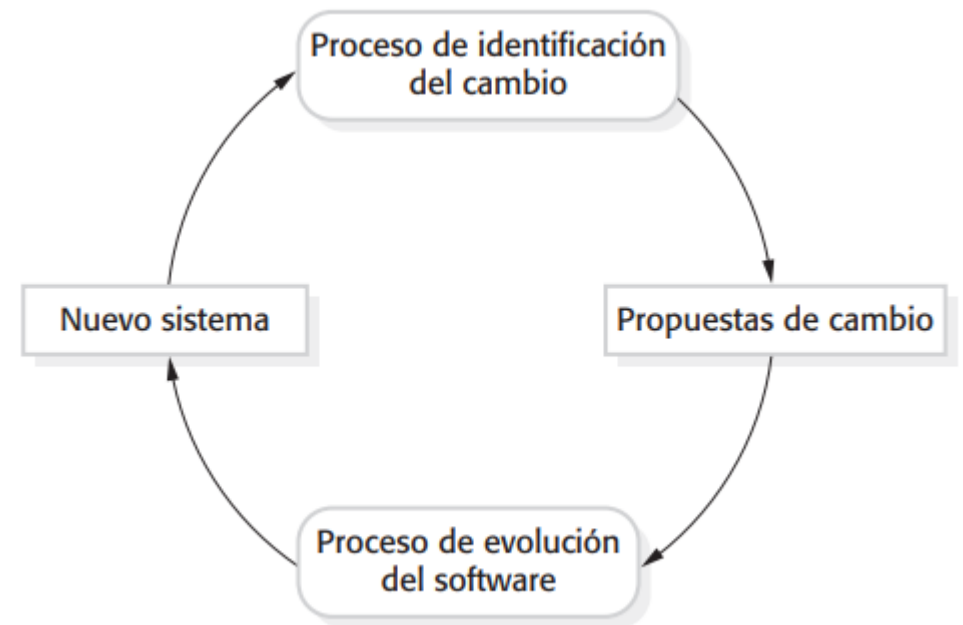
Proceso de evolución del software

Las propuestas de cambio son el motor para la evolución de un sistema.

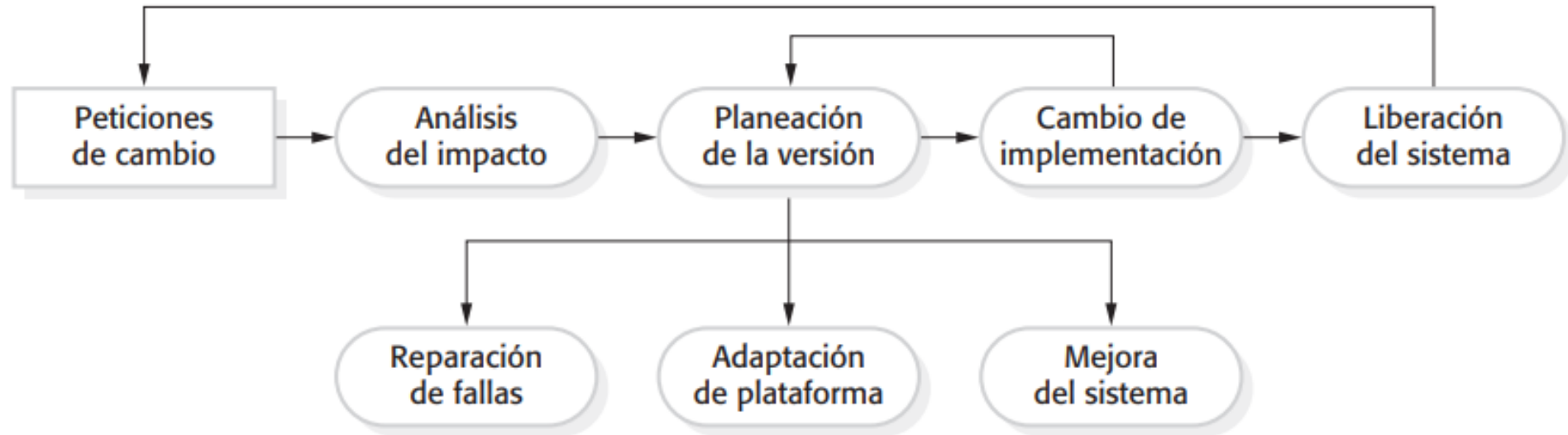
Proviene de:

- Requerimientos no implementados
- nuevos requerimientos
- reportes de errores
- ideas para la mejora del software

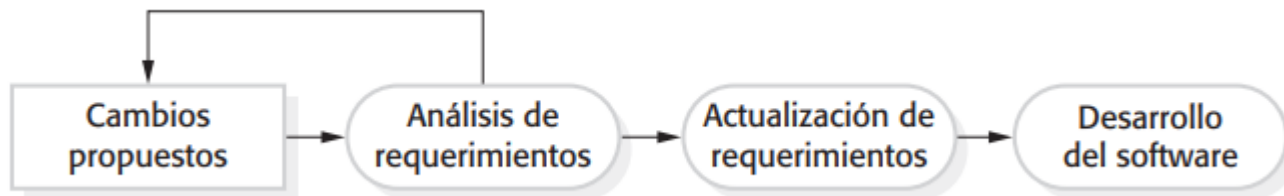
Los procesos de identificación de cambios y evolución del sistema son cíclicos y continúan a lo largo de la vida de un sistema



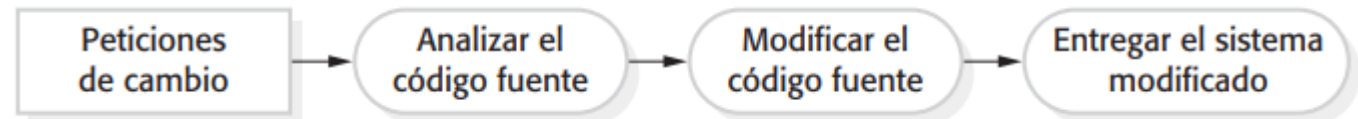
Proceso de evolución del software



Implementación del cambio (ideal)



Implementación del cambio (reparación de emergencial)



Leyes de Lehman.

Primera Ley (Cambio Continuo):

“Los sistemas deben ser continuamente adaptados o se convierten progresivamente en menos satisfactorios”

Segunda Ley (Complejidad incremental):

“Cuando un sistema evoluciona se incrementa su complejidad a menos que se trabaje para mantenerla o reducirla”

Tercer Ley (Evolución prolongada del programa o Autorregulación):

“El proceso de evolución de un sistema es autorregulado con una distribución de las medidas del producto y del proceso cercana a la normal”.

Cuarta Ley (Conservación de la estabilidad organizacional):

“La velocidad (y efectividad) de desarrollo de un sistema en evolución permanece invariante durante su ciclo de vida”

Quinta Ley (Conservación de la familiaridad):

“Cuando un sistema evoluciona, todos aquellos que están asociados a él deben mantener un conocimiento de su contenido y comportamiento para tratar de conseguir una evolución satisfactoria”

Sexta Ley (Crecimiento continuo):

“Las funcionalidades del sistema tienen que crecer constantemente para mantener la satisfacción del usuario a lo largo de su ciclo de vida”

Séptima Ley (Reducción de la calidad):

“La calidad de los sistemas comienza a disminuir a menos que se mantengan de forma rigurosa y se adapten a los cambios en su entorno de funcionamiento”

Octava Ley de Lehman (Realimentación del sistema):

“El proceso de evolución del sistema es consecuencia de un proceso de retroalimentación a diferentes niveles”



Leyes de Lehman.

Ley	Descripción
Cambio continuo	Un programa usado en un entorno real debe cambiar; de otro modo, en dicho entorno se volvería progresivamente inútil.
Complejidad creciente	A medida que cambia un programa en evolución, su estructura tiende a volverse más compleja. Deben dedicarse recursos adicionales para conservar y simplificar su estructura.
Evolución de programa grande	La evolución del programa es un proceso autorregulador. Los atributos del sistema, como tamaño, tiempo entre versiones y número de errores reportados, son casi invariantes para cada versión del sistema.
Estabilidad organizacional	Durante la vida de un programa, su tasa de desarrollo es aproximadamente constante e independiente de los recursos dedicados al desarrollo del sistema.
Conservación de familiaridad	A lo largo de la existencia de un sistema, el cambio incremental en cada liberación es casi constante.
Crecimiento continuo	La funcionalidad ofrecida por los sistemas tiene que aumentar continuamente para mantener la satisfacción del usuario.
Declive de calidad	La calidad de los sistemas declinará, a menos que se modifiquen para reflejar los cambios en su entorno operacional.
Sistema de retroalimentación	Los procesos de evolución incorporan sistemas de retroalimentación multiagente y multiciclo. Además, deben tratarse como sistemas de retroalimentación para lograr una mejora significativa del producto.



Mantenimiento del software

Es el proceso general de cambiar un sistema después de que éste se entregó.

El término usualmente se aplica a software personalizado, en el que grupos de desarrollo separados intervienen antes y después de la entrega.



Mantenimiento del software



Proceso de modificar un producto de software después de su puesta en producción



Corregir un mal funcionamiento

Mejorar atributos de calidad

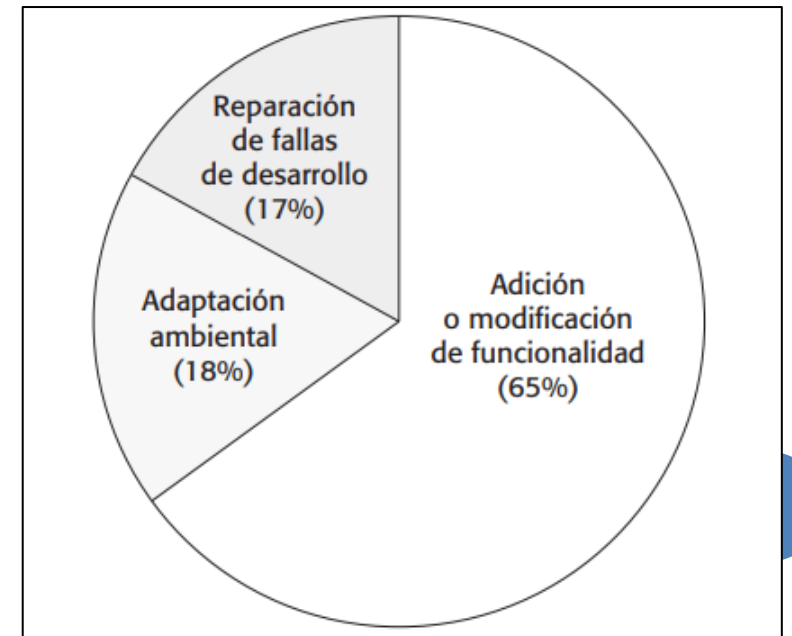
Adaptarlo a los cambios del entorno



Los costos del Mantenimiento de software

El mantenimiento del software toma una proporción más alta del presupuesto de TI que el nuevo desarrollo
Dos tercios del presupuesto corresponden a mantenimiento y un tercio a desarrollo

La mayor parte del presupuesto de mantenimiento se destina a la implementación de nuevos requerimientos



Los costos del Mantenimiento de software

Resulta más costoso agregar funcionalidad después de que un sistema está en operación, que implementar la misma funcionalidad durante el desarrollo

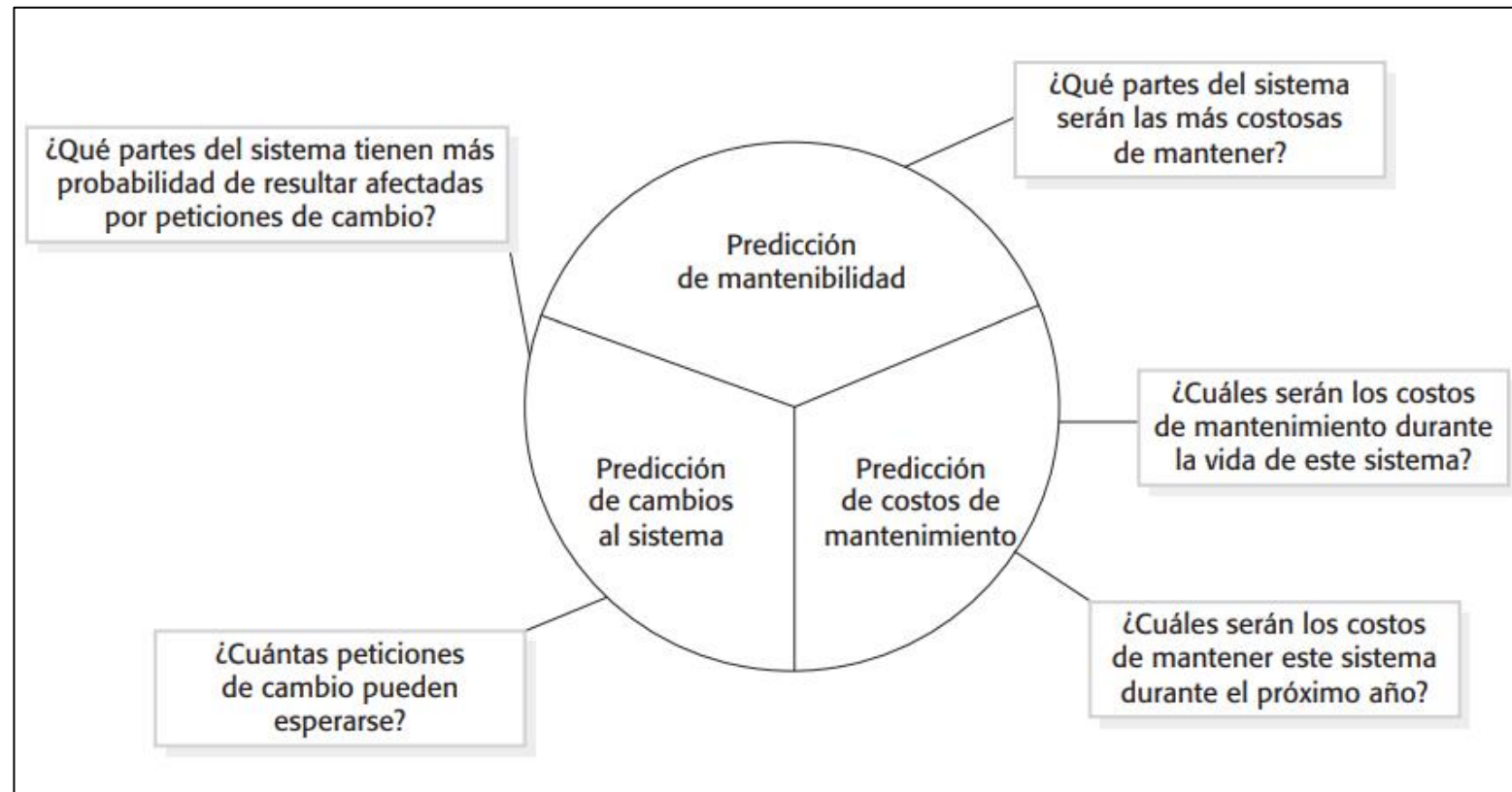
- **Estabilidad del equipo** Después la entrega, en general el equipo de desarrollo se separa.
- **Calidad de desarrollo deficiente** se ahorra esfuerzo durante el desarrollo aunque el software sea más difícil de cambiar en el futuro.
- **Habilidades del personal** El personal de mantenimiento con frecuencia es relativamente inexperto.
- **Antigüedad y estructura del programa** Conforme se realizan cambios al programa, su estructura tiende a degradarse.



Predicción del Mantenimiento de software

Predecir qué cambios pueden proponerse al sistema y qué partes serán las más difíciles de mantener.

Estimar los costos del mantenimiento durante cierto lapso de tiempo.



Predicción del Mantenimiento de software

Se debe valorar:

- **El número y la complejidad de las interfaces del sistema:** Cuanto más grande es el número de interfaces es más probable que se requieran cambios al agregar nuevos requerimientos.
- **El número de requerimientos de sistema inherentemente inestables:** Los requerimientos que reflejan políticas y procedimientos de la organización son más inestables que los que se basan en características de un dominio estable.
- **Los procesos empresariales donde se usa el sistema:** A medida que evolucionan los procesos empresariales se generan necesidades de cambio. Cuantos más procesos use un sistema, habrá más demandas de cambio.



Tipos de mantenimiento

```
graph TD; A[Tipos de mantenimiento] --> B[Preventivo]; A --> C[Correctivo]; A --> D[Perfectivo]; A --> E[Adaptativo]; B --> B1[Facilitar las posteriores correcciones, adaptaciones y mejoras]; C --> C1[Corregir defectos]; D --> D1[Extender los requerimientos funcionales o realizar mejoras de calidad]; E --> E1[Adaptar el software a las modificaciones del entorno];
```

Preventivo

Facilitar las posteriores correcciones, adaptaciones y mejoras

Correctivo

Corregir defectos

Perfectivo

Extender los requerimientos funcionales o realizar mejoras de calidad

Adaptativo

Adaptar el software a las modificaciones del entorno

Mantenimiento preventivo



Mejorar propiedades del software sin alterar sus especificaciones funcionales



Ejemplos

- Refactorización del código para mejorar su legibilidad.
- Modificar el software identificando componentes reusables.
- Comprobación de la validez de los datos de entrada

Mantenimiento correctivo




Localizar, corregir y eliminar los defectos del software



Ejemplos

- De procesamiento: salidas incorrectas
- De rendimiento: tiempo de respuesta alto en una búsqueda
- De programación: inconsistencias en el diseño de un programa
- De documentación: inconsistencias entre la funcionalidad de un programa y el manual de usuario.

Mantenimiento perfecto



Satisfacer cambios en requisitos
funcionales o no funcionales



Ejemplos

- Agregado de una funcionalidad no contemplada.
- Mejora de la usabilidad del sistema.
- Mejora de la velocidad de respuesta optimizando algoritmos.

Mantenimiento adaptativo



Modificar el software para adaptarlo a un nuevo entorno operativo



Ejemplos

- Cambio del sistema operativo.
- Cambio o actualización del gestor de bases de datos.
- Cambios de la configuración de hardware
- Cambios en la organización.
- Cambios legislativos.

Actividades del Mantenimiento de software

Las actividades de mantenimiento se agrupan en tres categorías funcionales:

- **Comprensión del software y de los cambios a realizar (Comprender):** es necesario el conocimiento a fondo de la funcionalidad, objetivos, estructura interna y requisitos del software. Alrededor del 50% de tiempo de mantenimiento se dedica a esta actividad.
- **Modificación del software (Corregir):** crear y modificar las estructuras de datos, la lógica de procesos, las interfaces y la documentación. Los programadores deben evitar los efectos laterales provocados por sus cambios. Esta actividad representa $\frac{1}{4}$ del tiempo total de mantenimiento.
- **Realización de pruebas (Comprobar):** realizar pruebas selectivas que nos aseguren la corrección del software.

Categoría	Actividad	% Tiempo
Comprensión del software y de los cambios a realizar	Estudiar las peticiones	18%
	Estudiar la documentación	6%
	Estudiar el código	23%
Modificación del software	Modificar el código	19%
	Actualizar la documentación	6%

