# MANTENIMIENTO DE SOFTWARE

# Definición de Mantenimiento

- El estándar IEEE 1219 [IEEE, 1993] define el Mantenimiento del Software como "la modificación de un producto software después de haber sido entregado [a los usuarios o clientes] con el fin de corregir defectos, mejorar el rendimiento u otros atributos, o adaptarlo a un cambio en el entorno".

# Definición de Mantenimiento (2)

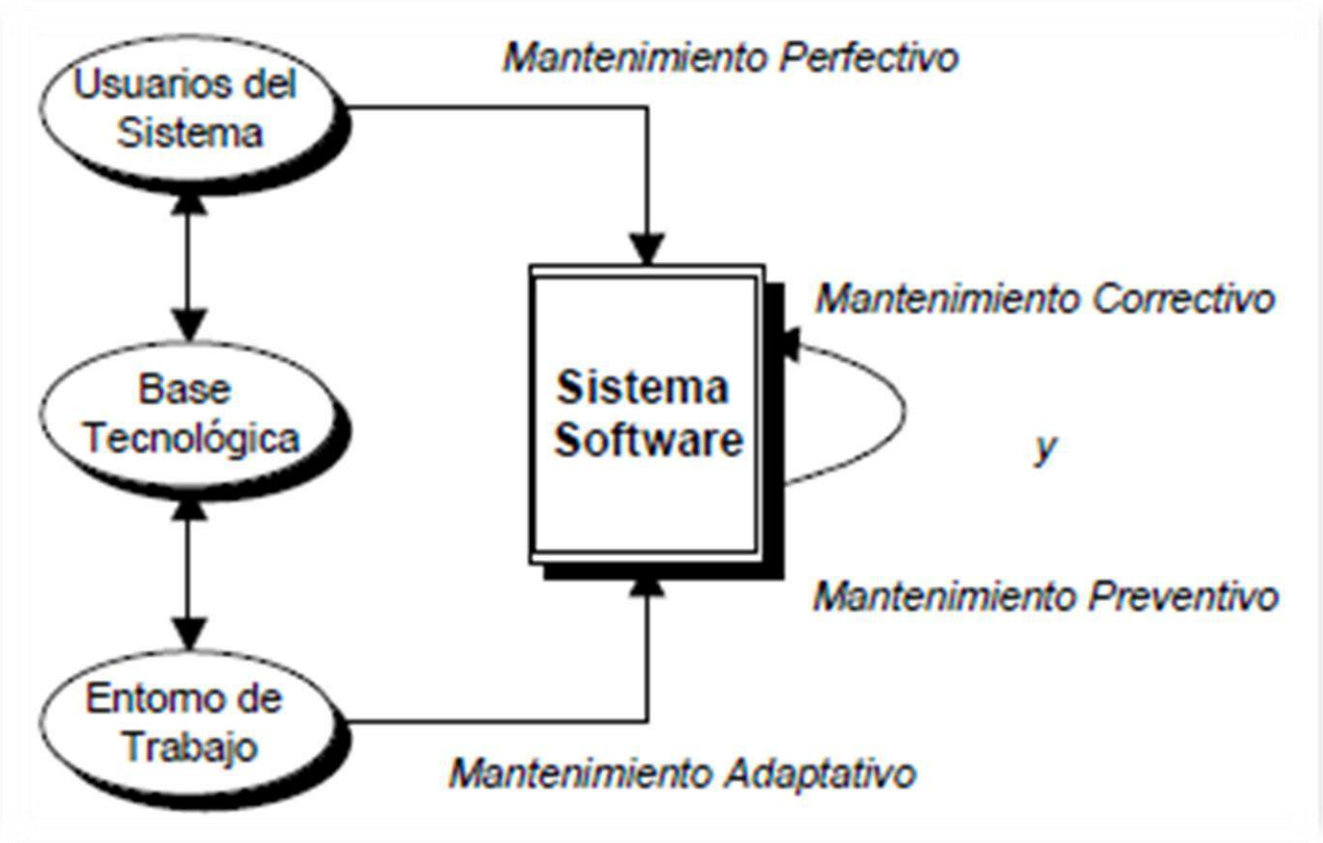
- En el estándar ISO 12207, de Procesos del Ciclo de Vida del Software [ISO/IEC, 1995] se establece que "el Proceso de Mantenimiento contiene las actividades y tareas realizadas por el mantenedor. Este proceso se activa cuando el producto software sufre modificaciones en el código y la documentación asociada, debido a un problema o a la necesidad de mejora o adaptación. El objetivo es modificar el producto software existente preservando su integridad. Este proceso incluye la migración y retirada del producto software. El proceso termina con la retirada del producto software". El mantenedor es la organización que proporciona el servicio de mantenimiento.

# Definición de Mantenimiento (3)

En otras fuentes bibliográficas clásicas aparecen definiciones similares a las anteriores.

- Pressman [1998] dice que "la fase mantenimiento se centra en el cambio que va asociado a la corrección de errores, a las adaptaciones requeridas a medida que evoluciona el entorno del software, y a cambios debidos a las mejoras producidas por los requisitos cambiantes del cliente".

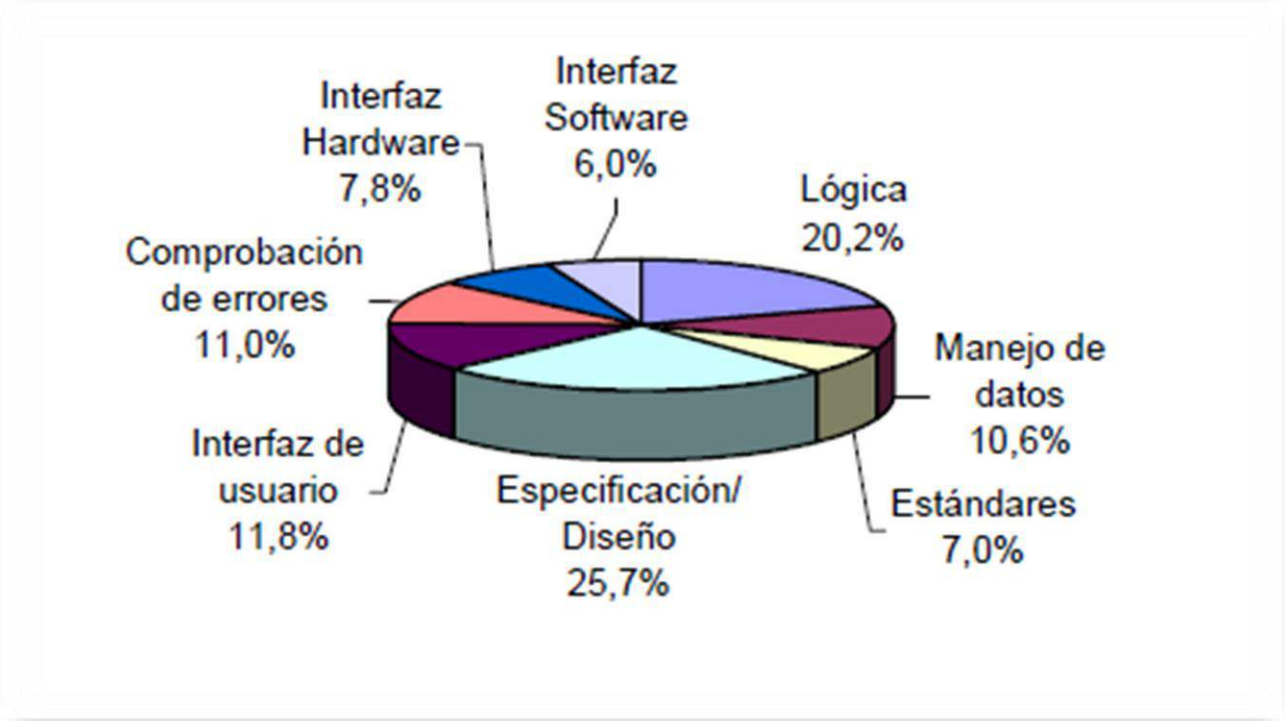
En las anteriores definiciones de mantenimiento aparecen indicados, directa o indirectamente, cuatro tipos de mantenimiento: correctivo, adaptativo, perfectivo y preventivo.


Mantenimiento Perfectivo
Usuarios del Sistema
Base Tecnológica
Entorno de Trabajo
Sistema Software
Mantenimiento Correctivo
y
Mantenimiento Preventivo
Mantenimiento Adaptativo

# Mantenimiento Correctivo

- El mantenimiento correctivo tiene por objetivo localizar y eliminar los posibles defectos de los programas. Un defecto en un sistema es una *característica del sistema con el potencial de causar un fallo*. Un fallo ocurre cuando el comportamiento de un sistema es diferente del establecido en la especificación.
- Entre otros, los fallos en el software pueden ser de:
  - **Procesamiento**, por ejemplo, salidas incorrectas de un programa.
  - **Rendimiento**, por ejemplo, tiempo de respuesta demasiado alto en una búsqueda de información.
  - **Programación**, por ejemplo, inconsistencias en el diseño de un programa.
  - **Documentación**, por ejemplo, inconsistencias entre la funcionalidad de un programa y el manual de usuario.

# Origen de los defectos del software



# Mantenimiento Adaptativo

- Este tipo de mantenimiento consiste en la modificación de un programa debido a cambios en el entorno (hardware o software) en el cual se ejecuta.
- Estos cambios pueden afectar al sistema operativo (cambio a uno más moderno), a la arquitectura física del sistema informático (paso de una arquitectura de red de área local a Internet/Intranet) o al entorno de desarrollo del software (incorporación de nuevos elementos o herramientas como ODBC).
- El tipo de cambio necesario puede ser muy diferente: desde un pequeño retoque en la estructura de un módulo hasta tener que re escribir prácticamente todo el programa para su ejecución en un ambiente distribuido en una red.

# Mantenimiento Adaptativo

- Los cambios en el entorno software pueden ser de dos clases:
  - **En el entorno de los datos,** por ejemplo, al dejar de trabajar con un sistema de ficheros clásico y sustituirlo por un sistema de gestión de bases de datos relacionales.
  - **En el entorno de los procesos,** por ejemplo, migrando a una nueva plataforma de desarrollo con componentes distribuidos, Java, ActiveX, etc.
- El mantenimiento adaptativo es cada vez más usual debido principalmente al cambio, cada vez más rápido, en los diversos aspectos de la informática: ***nuevas generaciones de hardware cada dos años, nuevos sistemas operativos -ó versiones de los antiguos- que se anuncian regularmente, y mejoras en los periféricos o en otros elementos del sistema***.
- Frente a esto, la vida útil de un sistema software puede superar fácilmente los diez años [Pressman, 1993].

# Mantenimiento Perfectivo

- Cambios en la especificación, normalmente debidos a **cambios en los requisitos** de un producto software, implican un nuevo tipo de mantenimiento llamado perfectivo.

- Desde algo tan simple como cambiar el formato de impresión de un informe, hasta la incorporación de un nuevo módulo aplicativo. Podemos definir el mantenimiento perfectivo como el conjunto de actividades para mejorar o añadir nuevas funcionalidades requeridas por el usuario.

# Mantenimiento Perfectivo

- Algunos autores dividen este tipo de mantenimiento en dos:
  - **Mantenimiento de Ampliación**: orientado a la incorporación de nuevas funcionalidades.
  - **Mantenimiento de Eficiencia**: que busca la mejora de la eficiencia de ejecución.

- Este tipo de mantenimiento aumenta cuando un producto software tiene éxito comercial y es utilizado por muchos usuarios, ya que cuanto más se utiliza un software, más peticiones de los usuarios se reciben demandando nuevas funcionalidades o mejoras en las existentes.

# Mantenimiento Preventivo

- Este último tipo de mantenimiento consiste en la modificación del software para mejorar sus propiedades (por ejemplo, aumentando su calidad y/o su mantenimiento) sin alterar sus especificaciones funcionales. Por ejemplo, se pueden incluir sentencias que comprueben la validez de los datos de entrada, re estructurar los programas para mejorar su legibilidad, o incluir nuevos comentarios que faciliten la posterior comprensión del programa. Este tipo de mantenimiento es el que más partido saca de las técnicas de ingeniería inversa y reingeniería.

# Mantenimiento Preventivo

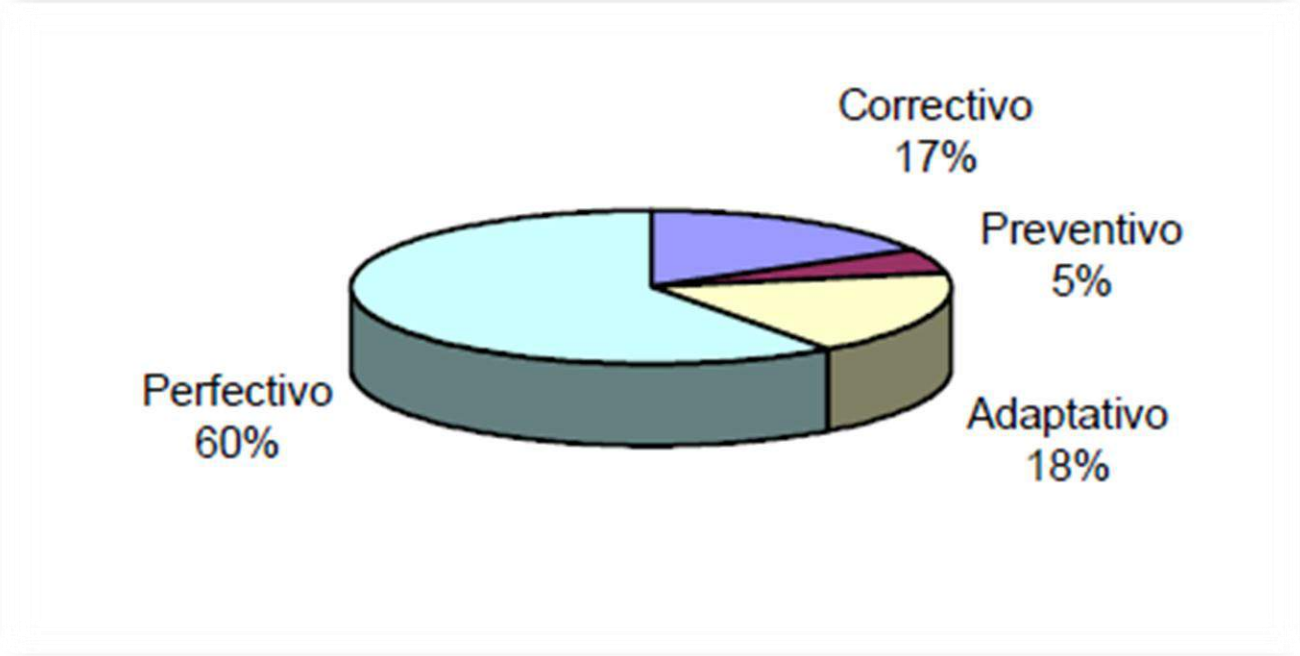
- En algunos casos se ha planteado el Mantenimiento para la Reutilización, consistente en modificar el software (buscando y modificando componentes para incluirlos en bibliotecas) para que sea mas fácilmente reutilizable. En realidad este tipo de mantenimiento es preventivo, especializado en mejorar la propiedad de reusabilidad del software.

# ACTIVIDADES DE MANTENIMIENTO

# Introducción

- El desconocimiento de las actividades que implica el mantenimiento del software puede inducir a minusvalorar su importancia, y se tiende a asociar el mantenimiento del software con la corrección de errores en los programas.

- Por esta causa, la impresión mas generalizada entre los gestores, usuarios, e incluso entre los propios informáticos, es que la mayor parte del mantenimiento que se realiza en el mundo es de tipo correctivo. Sin embargo, varios autores ([McKee, 1984], [Frazer, 1992], [Basili et al., 1996]) indican que esta impresión es equivocada, mostrando cómo los mayores porcentajes de esfuerzo se dedican a mantenimiento perfectivo (véase Figura 3, tomada de Frazer [1992]).


Correctivo
17%
Preventivo
5%
Perfectivo
60%
Adaptativo
18%

# Tipos de Actividades

- Basili et al. [1996] identifican las siguientes once actividades, que se realizan con cada modificación del software:
  - **Análisis de impacto y de costes/beneficios**: se dedica esta actividad a analizar diferentes alternativas de implementación y/o a comprobar su impacto en la planificación, coste y facilidad de operación.
  - **Comprensión del cambio**: puede consistir en localizar el error y determinar su causa, o en comprender los requisitos de una mejora solicitada.
  - **Diseño del cambio**: se refiere al diseño propuesto para el cambio, pudiéndose incluir un rediseño del sistema.
  - **Codificación y pruebas unitarias**: se codifica y prueba el funcionamiento de cada componente modificado.
  - **Inspección, certificación y consultoría**: esta actividad se dedica a inspeccionar el cambio, comprobar otros diseños, reuniones de inspección, etc.

- **Pruebas de integración**: se refiere a comprobar la integración de los componentes modificados con el resto del sistema.
- **Pruebas de aceptación**: en esta actividad, el usuario comprueba, junto al personal encargado del mantenimiento, la adecuación del cambio a sus necesidades.
- **Pruebas de regresión**: en esta actividad se somete el software modificado a casos de pruebas previamente almacenados y por los que ya pasó.
- **Documentación del sistema**: se revisa y reescribe, en caso necesario, la documentación del sistema para que se ajuste al producto software ya modificado.
- **Otra documentación (del usuario, por ejemplo):** se revisa y reescribe, en caso necesario, los diferentes manuales de usuario y otra documentación, excepto la documentación del sistema.
- **Otras actividades**, como las dedicadas a la gestión del proyecto de mantenimiento.

- Otros resultados interesantes de este mismo estudio se refieren a la distribución del esfuerzo en cada grupo de actividades según se trate de mantenimiento correctivo o perfectivo, resultando que:
  - La proporción de esfuerzo dedicado a comprensión es mucho mayor en el caso de mantenimiento correctivo que en perfectivo.
  - La proporción de esfuerzo empleado en inspección, certificación y consultoría es mucho mayor en el caso de mantenimiento perfectivo que en correctivo.
  - La proporción de esfuerzo dedicado a diseño, codificación y pruebas es muy similar en ambos tipos de mantenimiento.

# Distribución del esfuerzo en actividades de mantenimiento

| Grupo | Actividades | Porcentaje de esfuerzo |
|---|---|---|
| Análisis y comprensión | Análisis de impacto y de costes/beneficios.<br>Comprensión del cambio. | 13% |
| Diseño | Diseño del cambio.<br>50% de inspección, certificación y consultoría. | 16% |
| Implementación | Codificación y pruebas unitarias.<br>50% de inspección, certificación y consultoría. | 29% |
| Pruebas | Pruebas de integración.<br>Pruebas de aceptación.<br>Pruebas de regresión. | 24% |
| Otras | Documentación del sistema.<br>Otra documentación.<br>Otras actividades. | 18% |

# Dificultades del Mantenimiento

- La problemática del mantenimiento se resume en realizar el mantenimiento del software de forma tan rigurosa que la calidad no se deteriore como resultado de este proceso.

- La pregunta a formular es la siguiente: ¿cómo debe mantenerse el software para preservar su fiabilidad? A continuación veremos las circunstancias que hacen que la respuesta a esta pregunta no sea fácil y esté muy condicionada.

# Código Heredado

- En la actualidad, la mayor parte de éste software está formado por código antiguo "heredado" (del inglés legacy code); es decir, código de aplicaciones desarrolladas hace algún tiempo, con técnicas y herramientas en desuso y probablemente por personas que ya no pertenecen al colectivo responsable en este momento del mantenimiento del software concreto.
- En muchas ocasiones la situación se complica porque el código heredado fue objeto de múltiples actividades de mantenimiento. La opción de desechar este software y reescribirlo para adaptarlo a las nuevas necesidades tecnológicas o a los cambios en la especificación es muchas veces inadecuada por la gran carga financiera que supuso el desarrollo del software original y la necesidad económica de su amortización.

- Los problemas específicos del mantenimiento de código heredado han sido caracterizados en las llamadas Leyes del Mantenimiento del Software [Lehman, 1980]:
  - *Continuidad del Cambio* – Un programa utilizado en un entorno del mundo real debe cambiar, ya que si no cada vez será menos usado en dicho entorno. En otras palabras, esto significa que, tan pronto como un programa ha sido escrito, está ya desfasado. Las razones que conducen a esta afirmación son varias: a los usuarios se les ocurren nuevas funcionalidades cuando comienzan a utilizar el software; nuevas características en el hardware pueden permitir mejoras en el software; se encuentran defectos en el software que deben ser corregidos; el software debe instalarse en otro sistema operativo o máquina; o el software necesita ser más eficiente

- *Incremento de la Complejidad* – A la par que los cambios transforman los programas, su estructura se hará progresivamente más compleja salvo que se haga un esfuerzo activo para evitar este fenómeno. Esto significa que al realizar cambios en un programa (excluyendo el mantenimiento preventivo), la estructura de dicho programa se hace más compleja cuando los programadores no pueden o no quieren usar técnicas de ingeniería del software

- *Evolución del Programa* – La evolución de un programa es un proceso autorregulado. Las medidas de determinadas propiedades (tamaño, tiempo entre versiones y número de errores) revelan estadísticamente determinadas tendencias e invariantes.

- *Conservación de la Estabilidad* Organizacional – A lo largo del tiempo de vida de un programa, la carga que supone el desarrollo de dicho programa es aproximadamente constante e independiente de los recursos dedicados.

- *Conservación de la Familiaridad* – Durante todo el tiempo de vida de un sistema, el incremento en el número de cambios incluidos con cada versión (release) es aproximadamente constante. Casi veinte años después, el mismo autor vuelve a confirmar las leyes anteriores [Lehman et al., 1998]: la afirmación "los grandes programas no llegan nunca a completarse y están en constante evolución" se ve confirmada por el hecho de que, como ya hemos mencionado, algo más del 60% de las modificaciones que se realizan en el software se refieren a mantenimiento perfectivo.

# Problemas de Mantenimiento

- Además de las dificultades de mantenimiento que señalan las leyes anteriores, existen otros problemas clásicos que complican el mantenimiento [Schneidewind, 1987]:
  - A menudo, el mantenimiento es realizado de una manera ad hoc en un estilo libre establecido por el propio programador (esta opinión también es compartida por Pressman [1993], quien afirma que "raramente existen organizaciones formales, de modo que el mantenimiento se lleva a cabo como se pueda"). No en todas las ocasiones esta situación es debida a la falta de tiempo para producir una modificación diseñada cuidadosamente. Prácticamente todas las metodologías se han centrado en el desarrollo de nuevos sistemas y no han tenido en cuenta la importancia del mantenimiento. Por esta razón, no existen o son poco conocidos los métodos, técnicas y herramientas que proporcionan una solución global al problema del mantenimiento.

- Cambio tras cambio, los programas tienden a ser menos estructurados. Esto se manifiesta en una documentación desfasada (como afirman Baxter y Pigdeon [1997] al indicar que "la documentación completa o inexistente del sistema es uno de los cuatro problemas más importantes del mantenimiento de software"), código que no cumple los estándares, incremento en el tiempo que los programadores necesitan para entender y comprender los programas o en el incremento en los efectos secundarios producidos por los cambios. Todas estas situaciones implican casi siempre unos costes de mantenimiento del software muy altos.

- Es muy habitual que los sistemas que están siendo sometidos a mantenimiento sean cada vez más difíciles de cambiar (lo cual, como confirman Griswold y Notkin [1993], provoca que el mantenimiento sea cada vez más costoso). Esto se debe al hecho de que los cambios en un programa por actividades de mantenimiento dificultan la posterior comprensión de la funcionalidad del programa. Sommerville [1992] también apunta que "cualquier cambio conlleva la corrupción de la estructura del software y, a mayor corrupción, la estructura del programa se torna menos comprensible y más difícil de modificar".
- Por ejemplo, el programa original puede basarse en decisiones de programación no documentadas a las que no puede acceder el personal de mantenimiento. En estas situaciones, es normal que el software no pueda ser cambiado sin correr el riesgo de introducir efectos laterales no deseados debidos a interdependencias entre variables y procedimientos que el personal de mantenimiento no ha detectado.

- La falta de una metodología adecuada suele conducir a que los usuarios participen poco durante el desarrollo del sistema software. Esto tiene como consecuencia que, cuando el producto se entrega a los usuarios, no satisface sus necesidades y se tienen que producir esfuerzos de mantenimiento mayores en el futuro.
- Además de los problemas de carácter técnico anteriores, también pueden existir problemas de gestión. Muchos programadores consideran el trabajo de mantenimiento como una actividad inferior menos creativa- que les distrae del trabajo -mucho más interesante- del desarrollo de software. Esta visión puede verse reforzada por las condiciones laborales y salariales y crea una baja moral entre las personas dedicadas al mantenimiento. Como resultado de lo anterior, cuando se hace necesario realizar mantenimiento, en vez de emplear una estrategia sistemática, las correcciones tienden a ser realizadas con precipitación, sin pensarse de forma suficiente, no documentadas adecuadamente y pobremente integradas con el código existente. No es extraño, pues, que el propio mantenimiento conduzca a la introducción de nuevos errores e ineficiencias que conducen a nuevos esfuerzos de mantenimiento con posterioridad.

- Todos estos problemas se pueden atribuir – parcialmente al gran número de programas existentes que han sido desarrollados sin tener en cuenta la ingeniería del software. Esta área de la informática no es una panacea, pero, por lo menos, aporta soluciones parciales a los diversos problemas planteados con el mantenimiento del software.