# Mantenimiento del Software

S2

*Francisco Ruiz, Macario Polo*

Grupo Alarcos

Dep. de Informática

ESCUELA SUPERIOR DE INFORMÁTICA

UNIVERSIDAD DE CASTILLA-LA MANCHA


http://alarcos.inf-cr.uclm.es/doc/mso/

Ciudad Real, 2000/2001

# Índice - Sesión 2

- Dificultades del MS
  - Código Heredado
    - Leyes del Mantenimiento del Software
  - Problemas inherentes al MS
  - Efectos secundarios
    - sobre el código
    - sobre los datos
    - sobre la documentación
- Soluciones al Problema del MS
  - de Gestión
    - Mejora de los recursos dedicados al mantenimiento
    - Gestión de la Calidad
    - Gestión Estructurada del MS
    - Organización del equipo humano
    - Documentación de los cambios
  - Soluciones Técnicas
  - Soluciones - Conclusiones

# Dificultades del MS

- 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?*

- Las principales dificultades que hacen que la respuesta a esta pregunta no sea fácil y esté muy condicionada son:
  - La existencia de los llamados "*legacy code*" (código heredado).
  - Problemas inherentes al mantenimiento del software.
  - Efectos secundarios o laterales no previstos ni deseados.

# Código Heredado (i)

- Con el paso de los años se ha ido produciendo un volumen muy grande de software. 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.

# Código Heredado (ii)

- Los problemas específicos del mantenimiento de código heredado han sido caracterizados en las llamadas ***Leyes del Mantenimiento del Software***, propuestas por Lehman:
  - *Continuidad del Cambio.*
  - *Incremento de la Complejidad.*
  - *Evolución del Programa.*
  - *Conservación de la Estabilidad Organizacional.*
  - *Conservación de la Familiaridad.*

# Leyes del Mantenimiento del Software (i)

- **Continuidad del Cambio**: Un programa utilizado en un entorno del mundo real esta destinado a cambiar, ya que, en caso contrario, será utilizado cada vez menos en dicho entorno (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.

# Leyes del Mantenimiento del Software (ii)

- **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.

# Leyes del Mantenimiento del Software (iii)

- **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 producto software, el incremento en el número de cambios incluidos con cada versión (release) es aproximadamente constante.

# Problemas inherentes al MS (i)

- Además de las dificultades de mantenimiento que señalan las leyes anteriores, existen otros problemas clásicos que complican el mantenimiento y que son debidos a la propia naturaleza del proceso:
  - De carácter técnico:
    - Ausencia metodológica.
    - Tendencia a la desestructuración.
    - Disminución de la comprensibilidad.
    - Poca participación de los usuarios.
  - De gestión.
- Todos estos problemas se pueden atribuir -parcialmente- al gran número de programas existentes que han sido desarrollados sin utilizar la **ingeniería del software** (no es una panacea, pero aporta soluciones parciales a los diversos problemas planteados con el MS).

# Problemas inherentes al MS (ii)

- **Ausencia metodológica**: A menudo, el mantenimiento es realizado de una manera *ad hoc* en un estilo libre establecido por el propio programador. 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.
- **Tendencia a la desestructuración**: Cambio tras cambio, los programas tienden a ser menos estructurados. Esto se manifiesta en una documentación desfasada, 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 MS muy altos.

# Problemas inherentes al MS (iii)

- **Disminución de la comprensibilidad**: Es muy habitual que los sistemas que están siendo sometidos a mantenimiento sean cada vez más difíciles de cambiar. 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 (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 mantenedor no ha detectado.
- **Poca participación de los usuarios**: 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.

# Problemas inherentes al MS (iv)

- **Problemas de gestión**: 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.

# Efectos secundarios

- La posibilidad de error al cambiar un procedimiento lógico tan complejo como el que constituyen la mayor parte de los programas actuales es muy grande. Por esta razón, una de las principales dificultades del MS es el riesgo del llamado ***efecto bola de nieve,*** de manera que los cambios producidos por una petición de mantenimiento introducen efectos secundarios que implicarán nuevas peticiones de mantenimiento en el futuro. Estos efectos secundarios suponen nuevos defectos que aparecen como consecuencia de las modificaciones realizadas.
- Según las consecuencias que se derivan, los efectos secundarios del MS son de tres clases:
  - efectos sobre el código,
  - efectos sobre los datos, y
  - efectos sobre la documentación.

# Efectos secundarios sobre el código

- Todos los desarrolladores de software han "*sufrido*" en algún momento los problemas originados por olvidar añadir un ";" o por confundir un signo de puntuación con otro. Las consecuencias de estos "*despistes*" pueden ser muy importantes y sirven para corroborar que los efectos secundarios por cambios en el código son difíciles de prever.
- Las modificaciones en el código fuente que tienen una mayor probabilidad de inducir a nuevos errores son:
  - Cambios en el diseño que suponen muchos cambios en el código.
  - Eliminación o modificación de un subprograma.
  - Eliminación o modificación de una etiqueta.
  - Eliminación o modificación de un identificador.
  - Cambios para mejorar el rendimiento.
  - Modificación de la apertura/cierre de ficheros.
  - Modificación de operaciones lógicas.

# Efectos secundarios sobre los datos

- Las estructuras de datos constituyen una parte fundamental y básica en cualquier producto software, por lo que cualquier cambio que se produzca en ellas puede conducir a fallos importantes del sistema.
- Los efectos secundarios de este tipo pueden aparecer debido a los siguientes cambios:
  - Redefinición de constantes locales o globales.
  - Modificación de los formatos de registros o archivos.
  - Cambio en el tamaño de una matriz u otras estructuras similares.
  - Modificación de la definición de variables globales.
  - Reinicialización de indicadores de control o punteros.
  - Cambios en los argumentos de los subprogramas.
- Para reducir esta clase de efectos secundarios es importante una correcta documentación de todos los datos, incluyendo tablas de referencias cruzadas datos-procesos.

# Efectos secundarios sobre la documentación

- Los efectos secundarios de esta clase se producen cuando los cambios sobre el código de una aplicación no se reflejan en la documentación de diseño y/o en la documentación de usuario. Si la documentación técnica no se corresponde con el estado actual del software, se producirán efectos secundarios debidos a una incorrecta caracterización de las propiedades de dicho software. Por otro lado, la estima que los usuarios tendrán del producto software se reducirá considerablemente si comprueban que la documentación no se adapta a los ejecutables.
- Los cambios que con mayor probabilidad pueden producir efectos secundarios sobre la documentación son:
  - Modificar el formato de las entradas interactivas.
  - Nuevos mensajes de error no documentados.
  - Tablas o índices no actualizados.
  - Texto no actualizado correctamente.

# Soluciones al Problema del MS

- Las diversas propuestas para resolver este problema pueden dividirse en dos categorías:

  - **Soluciones de Gestión** (organizativas):
    - Mejora de los recursos dedicados al mantenimiento
    - Gestión de la calidad
    - Gestión estructurada del proceso (Metodologías)
    - Organización del equipo humano
    - Documentación de los cambios

  - **Soluciones Técnicas** (herramientas y Metodologías).

# Soluciones de Gestión

- En términos financieros, el MS puede ser visto como un continuo consumidor de recursos, mientras que los beneficios no están claros ni cuantificados. Para ayudar a evitar esta situación se necesita un **mayor apoyo por parte de la dirección** de las organizaciones para las actividades de mantenimiento. Para ello es necesario que los gestores veteranos (seniors) de las organizaciones sean conscientes de:
  - La importancia de las tecnologías de la información para la organización; y
  - Que el software es un activo corporativo que puede suponer una ventaja competitiva.
- Los gestores que estén descontentos con la situación y que quieran cambiarla, tendrán que adquirir un compromiso personal y visible con las soluciones organizativas propuestas. Tales soluciones se concretan fundamentalmente en dos aspectos: **los recursos y la calidad**.

# Mejora de los recursos dedicados al MS

*El recurso fundamental y clave para el MS es el humano.*

- Por tanto, una manera de mejorar el mantenimiento podría ser constituir un grupo separado de programadores dedicados a mantener código antiguo.
- Sin embargo, debido al carácter poco atractivo de este trabajo, es habitual que el personal nuevo recién incorporado sea asignado a esta actividad.
- Estos programadores inexpertos deben intentar comprender la lógica de diseño del sistema, a pesar de que no pueden comprender el modelo conceptual del software debido a que carecen de experiencia de uso de las técnicas de ingeniería del software y de conocimiento del dominio de lo que el programa realiza. Así, raramente saben cómo encontrar y corregir defectos o realizar modificaciones.

# Gestión de la Calidad (i)

- **Gestión de la calidad**: El aumento de los recursos humanos y económicos dedicados al MS puede suponer una solución a corto plazo, pero para resolver el problema a largo plazo se hace necesario adoptar una aproximación que permita mejorar la calidad del proceso en su conjunto.
- Entre las mejores técnicas de gestión de la calidad del software se incluyen:
  - Uso de técnicas estándares para la descomposición del software en entidades funcionales;
  - Empleo estricto de estándares de documentación del software;
  - Diseño paso a paso en cada nivel de descomposición del software;
  - Uso de código estructurado; y
  - Definición de todas las interfaces y estructuras de datos importantes antes de comenzar el diseño detallado.

# Gestión de la Calidad (ii)

- **¿qué es Calidad?**: es la adecuación de un producto o proceso a las especificaciones previamente establecidas.
- Los métodos para **aumentar la calidad**, tanto de un producto software como del proceso de su producción, se parecen cada vez más a los empleados en la industria en general:
  - Adecuación a estándares preestablecidos;
  - QSA (Quality Software Assurance);
  - TQ (Total Quality)
- Adicionalmente, pueden utilizarse **métricas** de producto (para medir los atributos del producto software) y métricas de procesos (para evaluar la calidad del proceso).
- Otra forma de poder mejorar la calidad es utilizar **mejores herramientas** de desarrollo de software (por ejemplo, un entorno único que integre editor, compilador y depurador).

# Gestión Estructurada del MS (i)

- Es muy importante emplear una **gestión estructurada** del proceso de MS.
- Este Mantenimiento Estructurado aparece como resultado de la **aplicación de una metodología** de ingeniería del software.
- La existencia de técnicas y herramientas adecuadas para Gestionar la *Configuración del Software* reduce la cantidad de esfuerzo requerido en el mantenimiento y mejora la calidad general de los cambios.

**Configuración del Software**: Documentación e información sobre los requerimientos, especificación, diseño y pruebas en cada una de las versiones para cada uno de los elementos software.

**Elemento Software**: cada uno de los componentes de un producto software: código de un módulo, fichero ejecutable, fichero de ayuda en línea, especificación de requisitos, documento de análisis, esquema de la base de datos, etc.

# Gestión Estructurada del MS (ii)

- Cuando el mantenimiento no es estructurado, se sufren las ***consecuencias de la falta de metodología***:
  - dolorosa evaluación del código (muchas veces poco legible),
  - complicada comprensión del sistema por la pobre documentación interna (desconocimiento de la estructura del programa, las estructuras de datos globales, las interfaces y otros requisitos de diseño y/o rendimiento),
  - dificultad para descubrir las consecuencias de los cambios en el código, y por último,
  - imposibilidad de realizar pruebas de regresión (repetición de pruebas anteriores) al no existir ningún registro de pruebas.
- Antes de optar por deshacerse de un programa y reescribirlo de nuevo, es necesario hacer un estudio detallado para evaluar las ventajas e inconvenientes de una y otra opción.

# Gestión Estructurada del MS (iii)

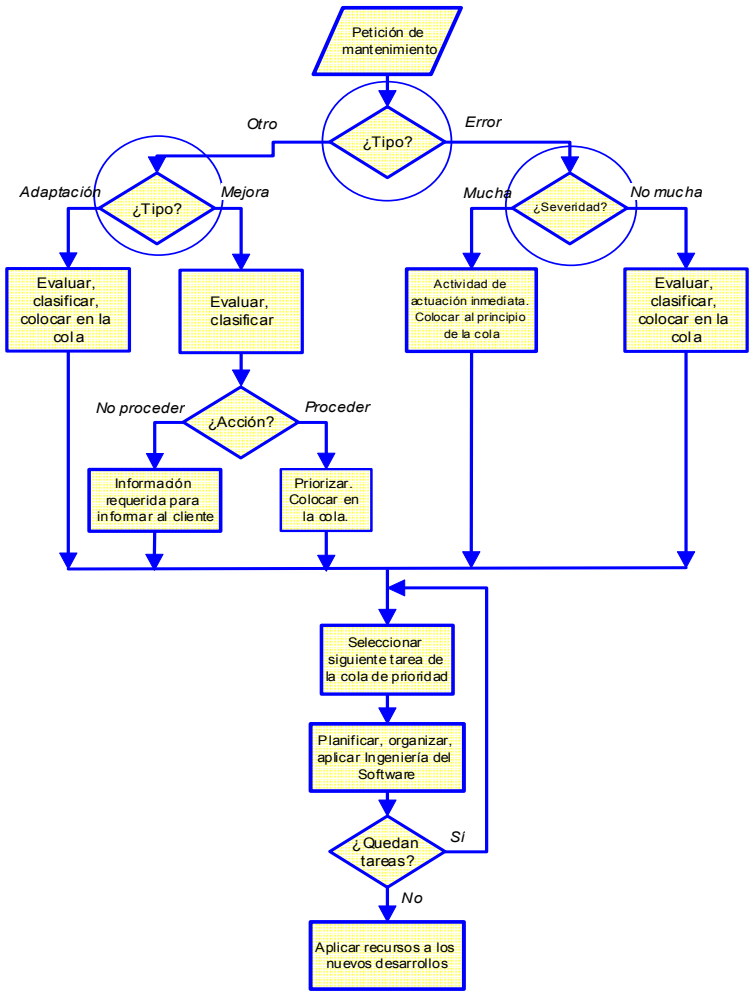
- Pueden **atenuarse las dificultades** al mantener código heredado siguiendo algunas sugerencias propuestas por Yourdon :
  - Prevenir antes que curar: obtener la mayor información posible sobre el programa antes de que surjan las emergencias de mantenimiento.
  - Conocer y entender el flujo de control general del programa. En caso de que no exista, dibujar los diagramas de estructura y de flujo de alto nivel.
  - Evaluar la documentación.
  - Añadir comentarios al código para facilitar su entendimiento posterior.
  - Utilizar las ayudas que proporcionan los compiladores: listados de referencias cruzadas, tablas de símbolos, etc.
  - Al realizar cambios, respetar el estilo y formato previos en la medida de lo posible.
  - Señalar las instrucciones cambiadas en el código.
  - Asegurarse antes de eliminar código (guardando una copia por si acaso).
  - Utilizar variables propias para evitar los posibles efectos secundarios que pueden surgir al utilizar las variables existentes previamente.
  - Llevar un registro completo de todas las actividades de mantenimiento.
  - Añadir comprobación de errores.

# Gestión Estructurada del MS (iv)

- Debe emplearse una metodología para gestionar y realizar el MS.
- Las características y necesidades del mantenimiento de software son diferentes que las del desarrollo. Por tanto es necesario disponer de metodologías específicas:

  *MANTEMA*

- Es una necesidad manifestada por las empresas de servicios informáticos relacionados con el mantenimiento (outsourcing, externalización, etc.).



# Organización del equipo humano

- Puesto que las tareas relacionadas con el mantenimiento comienzan mucho antes de que se realice la primera petición de mantenimiento, es muy aconsejable *establecer una organización del equipo de mantenimiento desde las fases iniciales del desarrollo*, definiendo claramente las personas que participarán en cada actividad para tratar de evitar que el mantenimiento se realice "como se pueda".
- Esta organización puede ser creada formalmente o simplemente constituirse de hecho, pero, en cualquier caso, se deberán *establecer claramente los procedimientos de evaluación, control, supervisión e información de cada petición de mantenimiento*.
- Asignar responsabilidades antes de comenzar las actividades de mantenimiento reduce considerablemente la confusión y ayuda a evitar en gran parte las incomodidades de la persona que se siente "mareada" al cambiarla apresuradamente de tarea: de desarrollo a mantenimiento y viceversa.

# Documentación de los cambios

• *Es muy importante realizar una correcta documentación de los cambios:* Por esta razón, es conveniente que las peticiones de mantenimiento se realicen utilizando un formulario estandarizado. Así mismo, el equipo de mantenimiento deberá elaborar un informe de cambios para cada petición de mantenimiento que deberá incluir un estudio del esfuerzo requerido para satisfacer la petición, la naturaleza de las modificaciones necesarias, y la prioridad (urgencia) del cambio.

• Las principales informaciones que se pueden registrar para cada cambio:

- Información del programa.
- Tamaño (LDC) del programa fuente.
- Tamaño del ejecutable.
- Lenguaje de programación utilizado.
- Fecha de instalación del programa.
- Número de fallos.
- Número de sentencias añadidas, eliminadas y modificadas en el cambio.
- Número de personas-hora.
- Identificación del responsable del cambio (ingeniero de software).
- Identificación de la petición de mantenimiento.
- Tipo de mantenimiento.
- Fechas de comienzo y final del mantenimiento.
- Beneficios netos que supone el cambio.

# Soluciones Técnicas (i)

- Las soluciones técnicas al problema del MS son de dos clases:
  **herramientas y métodos**.

- Las primeras sirven para soportar de forma más efectiva y cómoda los segundos.
- Estas herramientas han sido diseñadas para ayudar al personal de mantenimiento a comprender el programa y a probar sus modificaciones para asegurar que no han sido introducidos errores.
- Muchas de estas herramientas son iguales o similares a las utilizadas para la prueba (test) del software: formateador, analizador estático, estructurador, documentador, depurador interactivo, generador de datos de prueba y comparador.

# Soluciones Técnicas (ii)

- Los principales métodos empleados en el MS son:
  - **Reingeniería**: consiste en el examen y modificación de un sistema para reconstruirlo en una nueva forma.
  - **Ingeniería Inversa**: es el proceso de analizar un sistema para identificar sus componentes y las interrelaciones que existen entre ellos, así como para crear representaciones del sistema en otra forma o en un nivel de abstracción más elevado.
  - **Reestructuración** del software: consiste en la modificación del software para hacerlo más fácil de entender y cambiar o menos susceptible de incluir errores en cambios posteriores. Se diferencia de la ingeniería inversa en que el software reestructurado tiene el mismo nivel de abstracción que el original.

- Se estudian en otra sesión.

# Soluciones - Conclusiones

- Necesidad de emplear **Metodologías**, si puede ser, específicas para el proceso de MS.
- Necesidad de **Herramientas**:
  - Muchas metodologías han fracasado por no disponer de herramientas que permitan su automatización.
  - Además, hacen falta medidas para poder tomar decisiones.
  - Para ello debemos disponer de métricas que nos permitan medir los aspectos de interés en MS.
- Utilizar técnicas de Control y Gestión de la **Calidad**:
  - Gestión de Riesgos,
  - Gestión de Configuraciones,
  - Auditoría.