Liberación y Mantenimiento

¿Qué implica la liberación de un sistema?

1. Ayudar a los usuarios a **entender** y **usar** el sistema
   Tener en cuenta múltiples tipos de usuarios
2. Entrenamiento para la operación del sistema
3. Elaboración de la documentación
4. Solución de problemas
5. Conversión
   En caso de tratarse de una migración desde un sistema anterior o distinto
6. Instalación
   Puesta en funcionamiento del sistema en el ambiente de producción

Entrenamiento

Debemos entrenar a dos grupos de usuarios: los usuarios finales y los operadores/ administradores del sistema.
A los usuarios se les debe presentar cuáles son las funciones del sistema y cómo deben usarlo.
A los operadores/administradores, cuáles son las funciones de soporte y cómo funcionan.
Dentro de cada grupo de usuarios encontramos distintas necesidades ya que podemos contar con usuarios frecuentes ó eventuales, así como también usuarios con experiencia ó nuevos.

Revisión del entrenamiento

Debemos evaluar el entrenamiento para saber cómo encararlo. Para ello debemos tener en cuenta factores como el grado de uso del sistema, la eficiencia en el uso y el cumplimiento de objetivos del entrenamiento.
El entrenamiento debe tomar en cuenta además, las características y preferencias personales, el estilo de trabajo de los usuarios y las presiones de la organización.

Ayudas al entrenamiento

1. Elaborar documentos de entrenamiento, teniendo cuidado con el tamaño y teniendo en cuenta el aseguramiento de la legibilidad y una buena relación costo/beneficio; utilizar guías o referencias
2. Implementar una ayuda en línea
3. Realizar demostraciones de uso del sistema
4. Talleres
   Realizarlos cuando existan conflictos por disponibilidad del sistema, validación temprana u olvido por desuso.
5. Usuarios expertos
   Sirven como entrenadores para los otros usuarios y pueden brindar el primer nivel de

soporte aportando a la descentralización (no hacer que toda la responsabilidad del entrenamiento recaiga sobre nosotros)

Documentación

La documentación es parte del enfoque adecuado de un entrenamiento. Esta facilita el soporte y la solución de problemas por parte de los mismos usuarios.
Su importancia aumenta conforme aumenta el número de usuarios y la variedad de los mismos.
Se deben tener en cuenta atributos de calidad indispensables a la hora de elaborarla, como:

1. Legibilidad
    a. Estructura
    b. Tamaño
    c. Ilustraciones
    d. Facilidad para ubicar información relevante
2. Completitud
3. Correctitud

Esta debe atender las distinas visiones y necesidades de usuarios finales, administradores/ operadores, instalación/configuración, personalización/mantenimiento y visión general/detalles específicos.
Para los operadores en particular se deben atender:

1. Configuraciones de hardware y software
2. Procedimientos para:
    a. Autorizar acceso a usuarios
    b. Agregar o suprimir equipo periférico
    c. Generar copias de respaldo
    d. Solucionar problemas

Soporte y solución de problemas

Se debe brindar una guía de mensajes de error con ayuda para detectar, informar y manejar problemas, de forma de ser utilizada para la solución de los mismos.
Una buena idea es inlcuír una guía rápida con las funciones más importantes del sistema.
La ayuda y el entrenamiento en línea, así como la asistencia activa, son parte fundamental del soporte.

Conversión

Implica la sustitución de un sistema anterior por uno nuevo. La misma puede ser manual o automatizada, e incluir una carga inicial de datos básicos o información histórica del sistema anterior. Para este último caso se debe analizar la calidad de los datos.
Las estrategias de conversión conocidas son Big-Bang (cambiar de un sistema a otro completamente), Paulatina (es negativa porque deben convivir los dos sistemas durante un tiempo determinado, pero aporta para el ajuste de procedimientos durante el cambio) o Procesamiento en paralelo (los dos sistemas funcionan pero uno se mantiene en producción y el otro funciona en el área de prueba/control).

Instalación

Abarca la instalación del software de forma que quede disponible y operativo.
Su complejidad depende de la tecnología utilizada, las restricciones funcionales (por ejemplo, de tiempo) y los requerimientos de disponibilidad.
La facilidad de la instalación afecta la liberación inicial y las sucesivas liberaciones durante el mantenimiento.

¿Qué implica el mantenimiento?

Cualquier modificación realizada a un sistema luego de entrar en operación se considera mantenimiento. Así como el hardware sufre de deterioro, el software también a lo largo del tiempo. Esto se debe al cambio en las estructuras y la creciente complejidad producto de los sucesivos mantenimientos realizados a lo largo del tiempo.

Existen tres tipos de sistemas: S (Specifiable), P (Problem-Solving) y E (Embedded).

Sistemas S

Son sistemas cuya solución no está destinada a cambiar. Un cambio en la misma implicaría la resolución de otro problema distinto y no el original para el cual fue creado.
Se admite solo la realidad como un ambiente cambiante, pero tanto el problema como la especificación de requerimientos, el sistema y la información que devuelve el mismo se mantienen invariables.

Sistemas P

Estos sistemas son candidatos a cambiar. Si bien el problema es fijo y por lo tanto se mantiene, la especificación de requerimientos es una abstracción del mismo por lo que pueden existir soluciones mejores al mismo problema. Se consideran candidatos a cambiar, la realidad, la especificación de requerimientos, el sistema y la información que el mismo despliega.

Sistemas E

Sistemas que se encuentran embebidos en la realidad al punto de que todos los componentes son candidatos a cambiar. En estos sistemas, la realidad envuelve al resto de los componentes, haciendo variar el problema y con ello la especificación, el sistema y la información que despliega. El sistema termina formando parte de la realidad que modela, es decir, termina afectando a la realidad, pudiendo redefinir el problema y como consecuencia redefinirse a sí mismo.

Cambios durante el ciclo de vida

Todo puede cambiar, por lo que debemos considerar el cambio durante el desarrollo. Se debe facilitar el cambio en el transcurrir del mantenimiento, ya que no es generalmente posible

construir el sistema correcto la primera vez. El mantenimiento es una evolución del sistema.

Sin embargo, existen los llamados “sistemas legados” (legacy), construidos para otras necesidades y ambientes que luego de alcanzada una etapa en el mantenimiento, pierden el sentido de evolución del que hablamos.

Los sistemas legados son aquellos que han sido construidos hace años, normalmente con tecnología y herramientas hoy obsoletas, que pueden resultar críticos para las organizaciones por varias razones:

1. Es la aplicación central del negocio
2. Difícil de sustituir (costo, riesgo)
3. Alto costo de mantenimiento/operación (requiere actualización del personal, documentación, conocimiento, etc.)

Un desarrollo típico requiere entre seis meses y dos años de dedicación, mientras que el mantenimiento transcurre durante 5, 10 o más años. Con el paso del tiempo, la evolución del software que se obtiene con el mantenimiento, comienza a declinar, producto del aumento del costo del mismo (la complejidad aumenta), la reducción de la confiabilidad, complicaciones en la adaptabilidad, disminución del desempeño, la existencia de funciones de poca utilidad (deprecated functions) y por último, la existencia de sistemas que hacen lo mismo a menor costo.

Las leyes de la evolución del software de Lehman (año 80)

A partir de la observación de sistemas grandes en organizaciones grandes, Lehman formuló las siguientes leyes.

- Ley de continuidad del cambio
  El software cambia o se vuelve menos útil.
- Ley de complejidad creciente
  El deterioro de la estructura y la complejidad crecen, a menos que se haga algo para disminuirlas.
- Ley fundamental de la evolución de un programa
  Dinámica, con tendencia e invariantes estadísticos.
- Ley de conservación de la estabilidad organizacional
  La producción tiende a ser estadísticamente invariante (agregar más personal no la aumenta).
- Ley de conservación de la familiaridad (complejidad percibida)
  El contenido de las sucesivas versiones es estadísticamente invariante.

Naturaleza del mantenimiento

Los cuatro tipos de mantenimiento:

1. Correctivo
   Busca solucionar algún problema que surge en el uso diario.
2. Adaptativo

Surge como respuesta a cambios en el ambiente.
3. Perfectivo
   Mejora algún aspecto ya presente en el sistema.
4. Preventivo
   Introduce cambios para prever fallas.

En base a estudios realizados sobre 487 proyectos en el año ‘81, el 50% del mantenimiento realizado es perfectivo, mientras que el 25% es adaptativo, el 21% correctivo y el 4% restante, preventivo, lo que indica que la mayor parte del tiempo, el mantenimiento se trata de mejorar aspectos que están presentes en el software desde el principio.

Surgen problemas al realizar el mantenimiento, ya que se debe balancear la necesidad del cambio con la disponibilidad del sistema. Sumado a esto, surgen problemas de personal, problemas técnicos, necesidades conflictivas entre los involucrados, y problemas de costo.

Problemas de personal

- Comprensión limitada
    - Un 47% del esfuerzo es dedicado solo a entender el sistema
    - Un cambio provoca un impacto que puede traer como consecuencia la necesidad de volver a analizar el sistema
    - El personal suele reportar problemas con información incompleta o incorrecta
- Prioridades de la gerencia
    - Decidir si el mantenimiento y la mejora son más importantes que desarrollar un sistema nuevo
    - Según estudio, la motivación representa el 11,9% de los problemas, y las prioridades múltiples o cambiantes, el 8%

Problemas técnicos

- Especificaciones de diseño inadecuadas
- Programas y documentos de mala calidad
- Dificultades en pruebas
    - Problemas en el ambiente de testing
    - Dificultades con volúmenes de datos

Necesidades conflictivas

- Se requiere elegancia y principios de diseño aplicados, junto con la urgencia de obtener la solución
- Solucionar un problema en un sistema que no se domina
    - Debe ser una solución adecuada pero a la vez rápida
    - Afecta el impacto del cambio
- Necesidades de corto plazo y de largo plazo
    - Problemas por los costos actuales y los futuros
- Política de versiones
    - Se requiere una planificación para la incorporación de cambios

Problemas de costo

Todos los problemas contribuyen a elevar el costo de mantenimiento. A su vez, existe una tendencia creciente de los costos del mismo. En los años 80, entre un 40% y un 60% del costo total de un sistema, correspondían a los costos de mantenimiento. En los años 2000 se predice un 80% del total.
Los factores principales de esta incidencia, son la secuencia de reparaciones y mejoras centradas en el corto plazo, que aumentan los costos de mantenimientos subsiguientes (si lo arreglamos “así nomás”, cuando vayamos a arreglar de nuevo se nos complica), y por el otro lado, la necesidad de especializar al personal para poder ejecutar los cambios necesarios.

Otros factores que inciden en el costo del mantenimiento son:

1. Estabilidad del equipo de mantenimiento
2. Responsabilidad contractual
   Si el responsable se desliga del mantenimiento, tal vez no haya desarrollado para que el mismo sea posible y sencillo.
3. Habilidades del personal
   A menudo con poca formación/entrenamiento en herramientas y dominio de la aplicación.
4. Edad y estructura de los programas
   Con el tiempo los programas “envejecen” y se degrada la estructura, haciéndolos más difícil de entender y modificar.

Técnicas y herramientas

Para la gestión de la configuración se utilizan el Comité de control de cambios (CCC) y el Procedimiento de cambios. Por otro lado se requiere de Análisis de impacto sobre los cambios a producir.

Comité de control de cambios

Su objetivo es controlar los cambios (mejoras o correcciones de defectos). Este proceso involucra al cliente, los usuarios y los desarrolladores.
Pasos que sigue el comité:

1. Se recibe la solicitud de cambio o reporte de problema
2. CCC lo califica como defecto o cambio
3. CCC evalúa la severidad y el impacto del cambio
4. CCC prioriza con respecto a otras solicitudes
5. CCC asigna la atención de la solicitud

El procedimiento asegura que se cumplan los pasos y su registro, guardando información de cuándo, quién, por qué, qué cambió y si se aprobó.
Existen dos procedimientos según el tipo de solicitud de cambio: solicitud de cambio normal, o de emergencia. Vemos los pasos para una solicitud de cambio normal:

1. Evaluación

2. Priorización
3. Análisis de impacto
4. Implementación
5. Pruebas
6. Puesta en producción

Herramientas para el mantenimiento

- Editores de texto
- Compiladores y linkers
- Debuggers
- Analizadores estáticos de código
- Generadores de referencias cruzadas (para analizar el impacto de los cambios)
- Gestión de la configuración
    - Manejo de versiones
    - Control de acceso concurrente a las versiones
    - Facilidades para reconstruir un ejecutable (ejemplo: make)
    - Comparadores de archivo (ejemplo: diff)
    - Gestión de paquetes de cambio

Métricas del proceso

Estas permiten evaluar la mantenibilidad del sistema, pudiendo utilizarse como indicadores de problemas en el desarrollo. Se evaluan factores como:

1. Número de solicitudes de mantenimiento correctivo
2. Tiempo promedio para análisis de impacto
3. Tiempo promedio para implementar solicitud de cambio
4. Número de solicitudes de cambio pendientes

Un incremento en cualquiera de estos valores puede significar un problema de mantenibilidad.

Métricas del producto

Se mide la complejidad del mismo:

- Estructuras de control
- Estructuras de datos
- Tamaño de procedimientos y módulos

Estos indicadores permiten predecir el esfuerzo de mantenimiento.
Podemos elaborar una historia sobre el esfuerzo de mantenimiento por componente, lo que permitiría evaluar la conveniencia de re-escribir o re-diseñar en lugar de mantener dicho componente. Estudios muestran que la concentración del esfuerzo de mantenimiento está en pocos módulos.

Rejuvenecimiento del software

Es una actividad que trata de mejorar la calidad global de un producto, normalmente un sistema heredado. Hay tres metodologías:

1. Redocumentar

Analizar el código fuente y elaborar la documentación correspondiente de forma de proveer información para asistir al mantenimiento.

2. Reestructurar
   Transformar las estructuras de código en código bien estructurado. Implica transformar la arquitectura.
3. Ingeniería reversa
   Recrear el diseño y la especificación a partir del código.
4. Reingeniería
   Ingeniería reversa seguida de ingeniería directa para ajustar la especificación, rediseñar y construir. Es el proceso más largo.