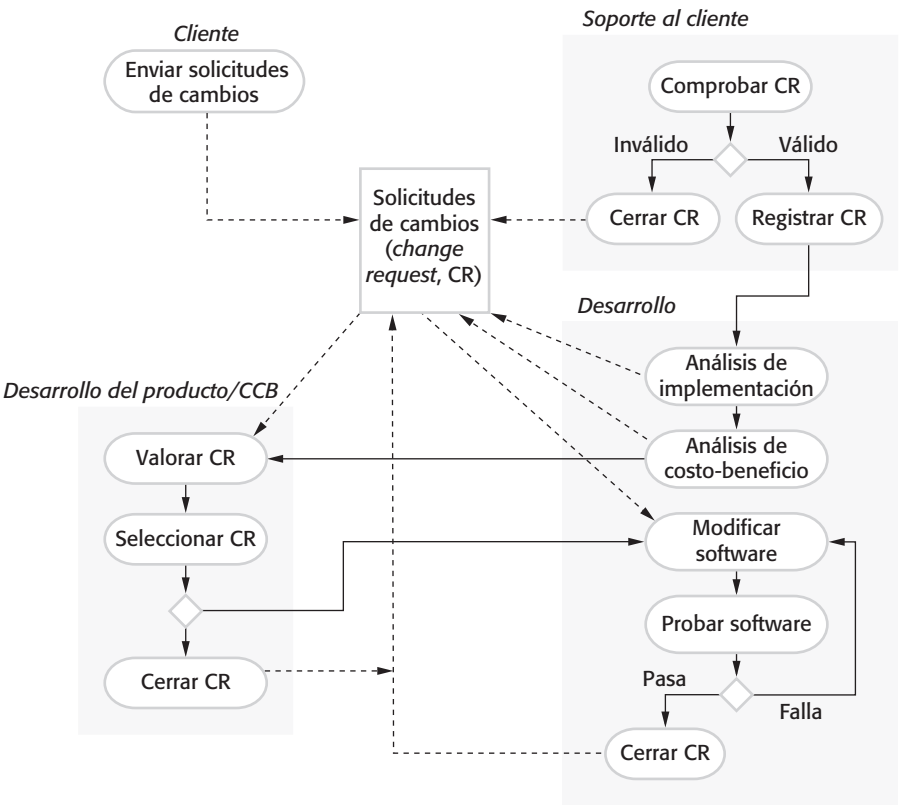

Figura 25.3 Proceso de administración del cambio

25.1 Administración del cambio

El cambio es un hecho en la vida de los grandes sistemas de software. Las necesidades y los requerimientos organizacionales cambian durante la vida de un sistema, los bugs deben repararse y los sistemas adaptarse a cambios en su entorno. Para garantizar que los cambios se apliquen al sistema de forma controlada, se necesita un conjunto de procesos de gestión de cambio soportado por herramientas. La administración del cambio tiene la intención de asegurar que la evolución del sistema sea un proceso gestionado en el que se da prioridad a los cambios más urgentes y rentables.

El proceso de administración del cambio se ocupa de analizar los costos y beneficios de los cambios propuestos, aprobar aquellos que lo ameritan e indagar cuál o cuáles de los componentes del sistema se modificaron. La figura 25.3 es un modelo de un proceso de administración que muestra las principales actividades de administración del cambio. Existen muchas variantes de este proceso en uso pero, para ser efectivos, los procesos de administración del cambio deben tener siempre un medio que compruebe, costee y apruebe los cambios. Este proceso debe entrar en efecto cuando el software se transfiere para la liberación a los clientes o la utilización dentro de una organización.

El proceso de administración del cambio se inicia cuando un "cliente" completa y envía una petición de cambio en que se describe el cambio requerido al sistema. Éste podría ser

Formato de petición de cambio

Proyecto: SICSA/AppProcessing **Número:** 23/02
Solicitante del cambio: I. Sommerville **Fecha:** 20/01/09
Cambio solicitado: El estatus de los solicitantes (rechazado, aceptado, etcétera) debe ser visible en la lista de despliegue de solicitantes.

Analizador del cambio: R. Looek **Fecha de análisis:** 25/01/09
Componentes afectados: ApplicantListDisplay, StatusUpdater

Componentes asociados: StudentDatabase

Valoración del cambio: Relativamente simple de implementar al cambiar el color de despliegue de acuerdo con el estatus. Debe agregarse una tabla para relacionar el estatus con los colores. No se requieren cambios a los componentes asociados.

Prioridad del cambio: Media
Implementación del cambio:
Esfuerzo estimado: 2 horas
Fecha para equipo SGA app.: 28/01/09 **Fecha de decisión CCB:** 30/01/09
Decisión: Aceptar cambio. Implementarse el cambio en la versión 1.2
Implementador del cambio: **Fecha de cambio:**
Fecha de envío a QA: **Decisión de QA:**
Fecha de envío a CM:
Comentarios:

Figura 25.4 Formato de petición de cambio parcialmente completado

el reporte de un bug, en el que se describan sus síntomas, o una petición para agregar alguna funcionalidad al sistema. Algunas compañías tratan por separado los reportes de bug y los nuevos requerimientos, pero, en principio, ambos son simplemente peticiones de cambio. Estas últimas pueden enviarse mediante un formato de petición de cambio (CRF, por las siglas de *change request form*). Aquí se usa el término *cliente* para incluir a cualquier participante que no sea parte del equipo de desarrollo, de modo que los cambios puede sugerirlos, por ejemplo, el departamento de marketing de una compañía.

Los formatos electrónicos de petición de cambios registran información que se comparte entre todos los grupos implicados en la administración del cambio. Conforme se procesa la petición del cambio, se agrega información al CRF para registrar las decisiones tomadas en cada etapa del proceso. Por lo tanto, en cualquier momento representa una fotografía instantánea del estado de petición del cambio. Además de registrar el cambio requerido, el CRF registra las recomendaciones concernientes al cambio, los costos estimados del cambio, y las fechas cuando se solicitó, aprobó, implementó y validó el cambio. El CRF también puede incluir una sección donde un desarrollador enfatice cómo puede implementarse el cambio.

En la figura 25.4 se muestra un ejemplo de formato de petición de cambio parcialmente completado. Éste es un ejemplo de un tipo de CRF que puede usarse en un proyecto grande y complejo de ingeniería en sistemas. Para proyectos más pequeños, se recomienda que las peticiones de cambio se registren de manera formal y el CRF se enfoque en la descripción del cambio requerido, con menos énfasis en los conflictos de implementación. Como desarrollador del sistema, usted decide cómo implementar dicho cambio y estima el tiempo requerido para ello.

Después de enviar una petición de cambio, ésta se verifica para asegurarse de que sea válida. Esta verificación puede venir tanto del cliente como del equipo de soporte de la aplicación, o para peticiones internas de un miembro del equipo de desarrollo. La comprobación es necesaria porque no todas las peticiones de cambio requieren acción. Si la petición de cambio es un reporte de bug, tal vez éste ya haya sido reportado. En ocasiones, lo que la gente considera como problemas en realidad son malas interpretaciones de lo que se espera que haga el sistema. Algunas veces, las personas solicitan características que ya se implementaron, pero que desconocen. Si algo de esto sucede, la petición de cambio se cierra y el formato se actualiza indicando la razón para el cierre. Si es una petición de cambio válida, entonces se registra como una petición sobresaliente para un análisis posterior.

Para peticiones válidas de cambio, la siguiente etapa del proceso consiste en evaluar y costear el cambio. Por lo general, esto es responsabilidad del equipo de desarrollo o del de mantenimiento, pues ellos están en condiciones de determinar lo que se requiere para la implementación del cambio. Debe comprobarse el efecto del cambio sobre el resto del sistema. Para hacer esto, hay que identificar todos los componentes afectados por el cambio. Si realizar el cambio significa que se necesitarán más modificaciones en alguna otra parte del sistema, esto aumentará el costo de implementar el cambio. A continuación, se valoran los cambios requeridos a los módulos del sistema. Finalmente, se estima el costo de efectuar el cambio y se toman en cuenta los costos de modificar los componentes asociados.

Continuando con este análisis, un grupo separado debe determinar si realizar el cambio al software es rentable desde una perspectiva empresarial. Para sistemas militares y gubernamentales este grupo se conoce usualmente como consejo de control del cambio (CCB, por las siglas de *change control board*). En la industria puede llamarse "grupo de desarrollo del producto", el cual es el responsable de tomar las decisiones sobre cómo debe evolucionar el sistema de software. Este grupo debe revisar y aprobar todas las peticiones de cambio, a menos que los cambios impliquen simplemente corregir errores menores en pantallas de despliegue, páginas Web o documentos. Estas peticiones menores deben transmitirse al equipo de desarrollo sin un análisis detallado, pues esto podría ser más costoso que implementar el cambio.

El CCB o el grupo de desarrollo del producto consideran el efecto del cambio desde un punto de vista estratégico y organizacional más que técnico. Decide si el cambio en cuestión está justificado económicamente y prioriza los cambios aceptados para su implementación. Los cambios aceptados se transmiten de regreso al grupo de desarrollo; las peticiones de cambio rechazadas se cierran y ya no se emprenden más acciones. Los factores significativos que deben tomarse en cuenta para decidir si un cambio debe aprobarse o no son los siguientes:

1. *Las consecuencias de no realizar el cambio* Cuando se valora una petición de cambio se debe considerar lo que ocurrirá si éste no se implementa. Si el cambio se relaciona con una falla reportada del sistema, la gravedad de dicha falla tiene que tomarse en cuenta. Si la falla del sistema causa la caída de este último, resulta muy grave, y no hacer el cambio puede perturbar el uso operacional del sistema. Por otra parte, si la falla tiene un efecto menor (por ejemplo, se presentan los colores equivocados en la pantalla), entonces no es importante corregir rápidamente el problema, de modo que el cambio tendrá una prioridad menor.

Clientes y cambios

Los métodos ágiles enfatizan la importancia de que los clientes participen en el proceso de priorización del cambio. El representante del cliente ayuda al equipo a decidir sobre los cambios que deben implementarse en la siguiente iteración de desarrollo. Aunque esto es efectivo para sistemas que están en desarrollo para un solo cliente, puede constituir un problema en el desarrollo de productos donde no hay un cliente real trabajando con el equipo. En esos casos, el equipo tiene que tomar sus propias decisiones respecto a la priorización del cambio.

http://www.SoftwareEngineering-9.com/Web/CM/agilechanges.html

2. *Los beneficios del cambio* ¿El cambio es algo que beneficiará a muchos usuarios del sistema o simplemente es una propuesta que beneficiará sobre todo a quien propone el cambio?
3. *El número de usuarios afectados por el cambio* Si sólo algunos usuarios resultan afectados, entonces al cambio se le puede asignar una baja prioridad. De hecho, hacer el cambio no resulta aconsejable si pudiera tener efectos adversos sobre la mayoría de los usuarios del sistema.
4. *Los costos de hacer el cambio* Si hacer el cambio afecta a muchos componentes del sistema (lo que, por lo tanto, aumenta las posibilidades de introducir nuevos bugs) y/o tarda mucho tiempo en implementarse, entonces se puede rechazar el cambio, debido a los elevados costos implicados.
5. *El ciclo de liberación del producto* Si una nueva versión del software se libera a los clientes, tal vez tenga sentido demorar la implementación del cambio hasta la siguiente liberación planeada (véase la sección 25.3).

La administración del cambio para productos de software (por ejemplo, un producto de sistema CAD), en vez de sistemas que se desarrollan específicamente para cierto cliente, tiene que manejarse en una forma relativamente diferente. En los productos de software, el cliente no participa de manera directa en las decisiones concernientes a la evolución del sistema, de manera que la relevancia del cambio no representa un problema para la compañía del cliente. Las peticiones de cambio para dichos productos provienen del equipo de soporte del cliente, el equipo de marketing de la compañía y los mismos desarrolladores. Dichas peticiones pueden reflejar sugerencias y retroalimentación de los clientes o análisis de lo que ofrecen los productos competitivos.

El equipo de soporte del cliente puede enviar peticiones de cambio asociadas con los bugs que los clientes descubrieron y reportaron después de que se entregó el sistema. Los clientes pueden usar una página Web o el correo electrónico para reportar los bugs. Entonces, un equipo de gestión de bugs comprueba que estos reportes sean válidos y los traduce a peticiones formales de cambio del sistema. El personal de marketing se reúne con los clientes e investiga productos competitivos. Pueden sugerir cambios que deban incluirse para facilitar la venta de una nueva versión de un sistema a clientes nuevos y existentes. Los propios desarrolladores del sistema pueden tener buenas ideas referentes a nuevas características que podrían agregarse al sistema.

El proceso de petición de cambio mostrado en la figura 25.3 se usa después de que un sistema se entregó a los clientes. Durante el desarrollo, cuando se crean nuevas versiones

```
// SICSA project (XEP 6087)
//
// APP-SYSTEM/AUTH/RBAC/USER_ROLE
//
// Objeto: currentRole
// Autor: R. Looek
// Fecha de creación: 13/11/2009
//
// © St Andrews University 2009
//
// Historial de modificación
// Versión  Modificador  Fecha       Cambio              Razón
// 1.0      J. Jones     11/11/2009  Agregar encabezado  Enviado a CM
// 1.1      R. Looek     13/11/2009  Nuevo campo         Pet. de cambio R07/02
```

Figura 25.5 Historia de derivación

del sistema mediante construcciones diarias (o más frecuentes), por lo general se usa un proceso de administración del cambio más sencillo. Los problemas y cambios aún deben registrarse, pero los cambios que sólo afectan a componentes y módulos individuales no necesitan valorarse de manera independiente; se transmiten directamente al desarrollador del sistema. Éste los acepta u ofrece razones por las que no son necesarios tales cambios. Sin embargo, una autoridad independiente, como el arquitecto del sistema, debe valorar y priorizar los cambios que afectan a aquellos módulos del sistema que produjeron diferentes equipos de desarrollo.

En algunos métodos ágiles, como en la programación extrema, los clientes participan directamente en la decisión de implementar un cambio. Cuando proponen un cambio a los requerimientos del sistema, trabajan con el equipo para valorar el efecto de dicho cambio y deciden entonces si éste tendría prioridad sobre las características planeadas para el siguiente incremento del sistema. No obstante, los cambios que implican mejoramiento del software se dejan a discreción de los programadores que trabajan en el sistema. La refactorización, en la que el software se mejora de manera continua, no se ve como una carga, sino como parte necesaria del proceso de desarrollo.

Conforme el equipo de desarrollo modifica los componentes de software, debe mantener un registro de los cambios hechos a cada componente. Algunas veces a esto se le conoce como historial de derivación de un componente. Una buena forma de conservar el historial de derivación es en un comentario estandarizado al principio del código fuente del componente (figura 25.5). Este comentario debe hacer referencia a la petición de cambio que provocó el cambio en el software. Entonces uno puede escribir rutinas sencillas que busquen todos los componentes y procesen los historiales de derivación para generar reportes de cambio de componentes. En el caso de documentos, los registros de los cambios incorporados en cada versión se anotan por lo general al frente del documento en una página aparte. Esto se discute en el capítulo Web sobre documentación.

La administración del cambio, por lo general, recibe soporte de herramientas especializadas de software. Éstas pueden ser herramientas relativamente sencillas basadas en la Web, como Bugzilla, que se usa para reportar problemas con muchos sistemas de código abierto. O bien, pueden usarse herramientas más complejas para automatizar todo el proceso de manejar las peticiones de cambio desde la propuesta inicial del cliente hasta la aprobación del cambio.

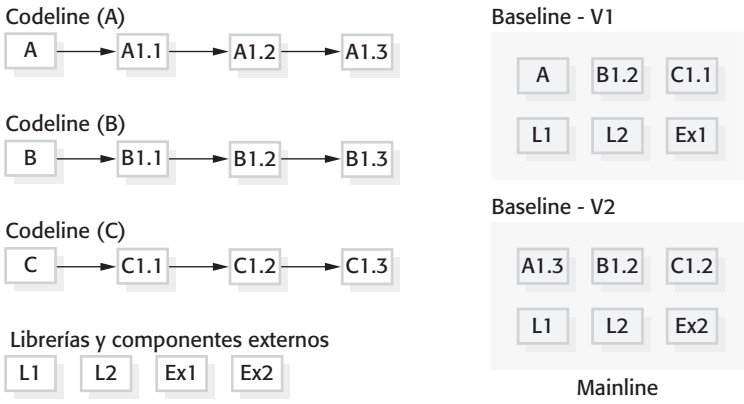

Figura 25.6 Líneas de código (*codelines*) y líneas base (*baselines*)

25.2 Gestión de versiones

La gestión de versiones (VM, por las siglas de *version management*) es el proceso de hacer un seguimiento de las diferentes versiones de los componentes de software o ítems de configuración, y los sistemas donde se usan dichos componentes. También incluye asegurar que los cambios hechos a dichas versiones por los diferentes desarrolladores no interfieran unos con otros. Por lo tanto, se puede considerar a la gestión de versiones como el proceso de administrar líneas de código y líneas base.

La figura 25.6 ilustra las diferencias entre línea de código y línea base. En esencia, una línea de código es una secuencia de versiones de código fuente con las versiones más recientes en la secuencia derivadas de las versiones anteriores. Las líneas de código se aplican regularmente a componentes de sistemas, de manera que existen diferentes versiones de cada componente. Una línea base es una definición de un sistema específico. Por consiguiente, la línea base especifica las versiones del componente que se incluyen en el sistema más una especificación de las librerías usadas, archivos de configuración, etcétera. En la figura 25.6 se observa que diferentes líneas base usan distintas versiones de los componentes de cada línea de código. En el diagrama se sombrearon los recuadros que representan componentes en la definición línea base para indicar que en realidad son referencias a componentes en una línea de código. La línea principal es una secuencia de versiones del sistema desarrolladas a partir de una línea base original.

Las líneas base pueden especificarse mediante un lenguaje de configuración, lo que permite definir cuáles componentes se incluyen en una versión de un sistema particular. Es posible especificar de manera explícita una versión de componente específica (X.1.2, por ejemplo) o simplemente especificar el identificador del componente (X). Si usa el identificador, esto significa que en la línea base debe usarse la versión más reciente del componente.

Las líneas base son importantes porque muchas veces es necesario volver a crear una versión específica de un sistema completo. Por ejemplo, una línea de producto puede ejemplificarse de modo que existan versiones de sistema individuales para diferentes clientes. Posiblemente se tenga que volver a crear la versión entregada a un cliente específico si, por ejemplo, dicho cliente reporta bugs en su sistema que deban repararse.

Para soportar la gestión de versiones, siempre se deben usar herramientas de gestión de versiones (llamadas en ocasiones sistemas de control de versiones o sistemas de control