Estructura básica del proceso unificado de desarrollo de software

Robin Alberto Castro Gil
Universidad Icesi
rcastro@icesi.edu.co

Fecha de recepción: 13-01-2004 Fecha de aceptación: 14-04-2004

ABSTRACT

For a long time, the traditional waterfall model has been used in software development. Over this time it has demonstrated that it does not reflect properly the inherent complexity of the software development process. The problems that this model presents stem from its structure: a sequence of large stages that have complete documentation as a milestone before being able to continue to the following stage.

To solve this problem it is neccesary to use iterative and incremental methods that together with other key practices, such as risk management and the adaptable planning, provide a natural guidance to the software development process. IBM's RUP is taken into special consideration. RUP is based on the spiral model and organizes iterations into stages and phases in order to obtain a more solid, clear, and adjustable structure of the particular needs of every organization.

This article aims to describe iterative development iterative in general and the advantages that its use offers in software develoment processes.

KEYWORDS

Spiral Model, Iterative Development, Rational Unified Process.

RESUMEN

Durante mucho tiempo se ha utilizado el tradicional modelo en cascada, el cual ha demostrado que no refleja adecuadamente la complejidad inherente al proceso de desarrollo de software. Los problemas que presenta este modelo nacen de su propia estructura, al ser una secuencia de grandes etapas que requieren como hitos la documentación completa antes de continuar con la siguiente etapa.

Para solucionar este problema se debe hacer uso de métodos iterativos e incrementales, que unidos a otras prácticas claves como la orientación al manejo de riesgos y la planeación adaptable, permiten de forma natural guiar adecuadamente el proceso de desarrollo de software. En especial se considerará el RUP[1] de IBM, basado en el modelo en espiral que organiza las iteraciones por etapas y fases para obtener una estructura más sólida, clara y ajustable a las necesidades particulares de cada organización.

Este artículo tiene como objetivo describir en forma general el desarrollo iterativo, y las ventajas que ofrece su utilización en los procesos de desarrollo de software.

PALABRAS CLAVES

Modelo en espiral, desarrollo iterativo, proceso unificado de rational

Clasificación: B

1. RUP: Rational Unified Process de IBM.

INTRODUCCIÓN

En los proyectos de tecnología se han incorporado paulatinamente los cambios en los métodos de ingeniería, pero, desafortunadamente, no se ha hecho lo mismo en su administración. A continuación se mostrará una visión general de los cambios en los ciclos de vida de los proyectos, señalando sus ventajas, desventajas y el por qué de dicha evolución.

Existe un conjunto de modelos de ciclo de vida en la gerencia de proyectos que han evolucionado desde el tradicional modelo en cascada, pasando por algunas propuestas de mejoramiento como son el caso de cascadas con fases solapadas o cascada con subproyectos. Estos modelos no permiten una identificación temprana de los riesgos, los cuales pueden aparecer en las etapas finales del desarrollo o implantación, momento en que un cambio en el diseño del producto, en la arquitectura del sistema o en la infraestructura de software y hardware puede llevar al fracaso completo del proyecto.

Estos modelos son muy prácticos para proyectos pequeños y con muy bajos niveles de riesgos, tales como proyectos de nuevas versiones de algún software existente, donde haya requerimientos claros y cuya arquitectura e infraestructura de software y hardware no van a cambiar mucho respecto a la versión anterior.

Como respuesta al problema presentado por los modelos anteriores, los ciclos de vida evolucionaron y se han presentado propuestas como el modelo de entrega por etapas y el de entrega evolutiva. Cada uno de ellos adicionó mayores niveles de complejidad a la administración, pero aseguran poseer un marco de trabajo más sólido y ajustado para el desarrollo de proyectos con niveles moderados de riesgo. Se requería de todas formas un modelo de ciclo de vida de proyectos que trabajara adecuadamente con niveles altos de riesgo, así que se desarrolló el modelo en espiral. Este modelo tiene como objetivos la identificación de los riesgos para determinar la viabilidad del proyecto y definir planes de manejo para garantizar desde las fases iniciales la eliminación o mitigación de los riesgos donde es menos costoso y la entrega desde las faces iniciales de productos probados, lo que permite un proceso continuo de pruebas y retroalimentación.

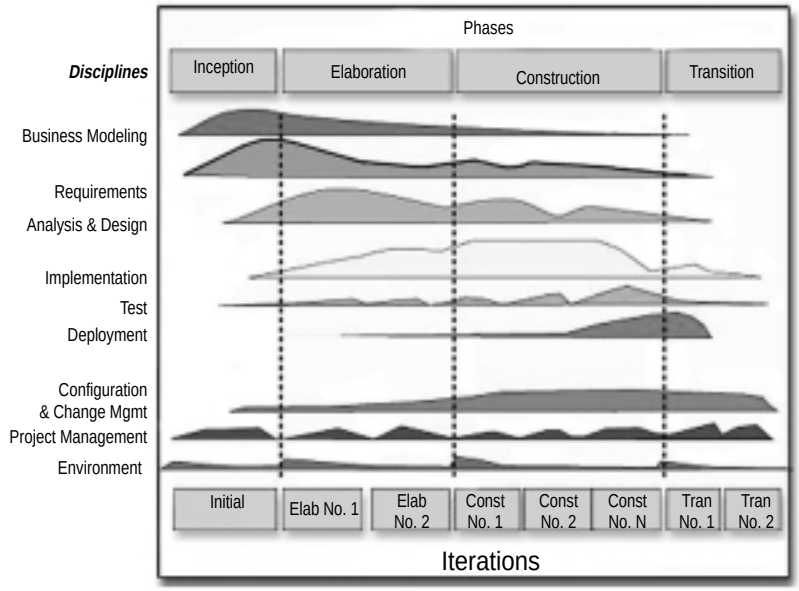
PRÁCTICAS CLAVES DEL PROCESO DE DESARROLLO DE SOFTWARE

• *Desarrollo iterativo*

El desarrollo iterativo es un método de construcción de productos cuyo ciclo de vida está compuesto por un conjunto de iteraciones, las cuales tienen como objetivo entregar versiones del software. Cada iteración se considera un subproyecto que genera productos de software y no sólo documentación, permitiendo al usuario tener puntos de verificación y control más rápidos e induciendo un proceso continuo de pruebas y de integración desde las primeras iteraciones.

Las iteraciones están compuestas por el conjunto de disciplinas o actividades ya conocidas en el proceso de desarrollo de software. Estas son la especificación de requerimientos, el análisis y diseño, las pruebas, la administración de la configuración y el proceso de gerencia de proyectos. En la Figura 1 se muestra gráficamente la relación existente entre las disciplinas o actividades y las iteraciones.

Figura 1. Disciplinas a través de las iteraciones.



Fuente: IBM RUP Rational Unified Process® Versión 2002.05.00. Rational Software Corporation.

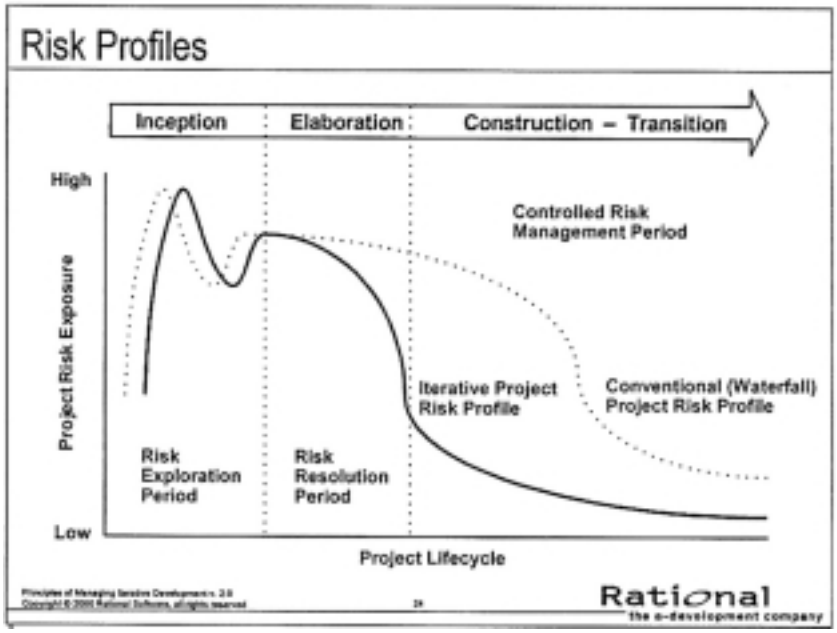
A través del tiempo se han formalizado algunos métodos de desarrollo iterativo, tales como Evo, Microsoft Solution Framework, modelo en espiral WinWin y UP. Este artículo se centrará en el método de desarrollo iterativo conocido como Proceso Unificado o UP.

• *Orientación al manejo del riesgo*

Cada proyecto tiene asociado intrínsecamente un conjunto de riesgos que requieren un plan de manejo claramente establecido, documentado y con una implementación eficaz. De esta manera se pretende evitar posibles retrasos en los tiempos de entrega, problemas de calidad en el producto o en el peor de los casos, que puedan afectar la culminación del proyecto. Estos procesos pueden ser tan complejos y elaborados como la importancia del proyecto lo requiera.

En las etapas iniciales se implementan las funcionalidades con mayor exposición al riesgo y las de mayor complejidad, mejorando la posibilidad de éxito del proyecto.

Figura 2. Perfiles de riesgo.



Fuente: Principles of Managing Iterative Development v.2.0 Rational Software Corporation.

En la fase inicial del proyecto, el nivel de exposición al riesgo en ambos modelos es casi igual, pero en las fases siguientes es completamente diferente para cada modelo (Ver Figura 2). Este comportamiento se debe al período de exploración de riesgos del modelo en espiral, donde se identifican los riesgos, se priorizan y se define un plan de manejo para mitigarlos. Se procede a la fase de elaboración donde se implementan aquellos casos de uso que atacan los riesgos de más alta prioridad, lo cual se denomina período de resolución de riesgos. Al final de esta fase se debe tener definida la arquitectura del sistema, así como la infraestructura en la que se soportará.

• *Orientación al cliente*

Cuando se inicia un proyecto de desarrollo de software se conoce la importancia de la participación del cliente para lograr su terminación exitosa, pero usualmente cometemos el error de olvidar esta norma básica, lo que implica que la participación del cliente se restringe al inicio y finalización del proyecto, lo que en la mayoría de los casos produce un alto grado de insatisfacción en el usuario, al no obtener el producto con las especificaciones esperadas.

El cliente es quien realmente conoce el valor que aportará el producto que está siendo desarrollado y puede definir las prioridades desde la perspectiva organizacional. Esto quiere de-

cir que es necesario contar con su participación en el proceso de planificación de las fases y de las iteraciones. Posteriormente se requiere su participación en cada iteración para proveer retroalimentación temprana al equipo de desarrolladores, garantizando el cumplimiento de las expectativas que tiene, además de ofrecerle una visibilidad permanente del estado del proyecto, asegurando su compromiso para terminarlo exitosamente.

Se debe tener en cuenta que el cliente no se interesa por los aspectos técnicos de alta complejidad y riesgo, razón por la cual se debe combinar esta práctica con una orientación al manejo del riesgo.

• *Desarrollo evolutivo*

Cuando se trabaja con una especificación de requerimientos monolítica, se cae en el error de creer que se comprende completamente el concepto del producto sin haberlo validado con el cliente con algo más que documentos y modelos abstractos. Este proceso inicia con un concepto poco claro del producto a construir, y sólo se tiene claridad en la medida que se vaya desarrollando y verificando el producto con el cliente. Este tipo de proyectos se asemejan más al patrón que siguen los proyectos de investigación y desarrollo de nuevos productos.

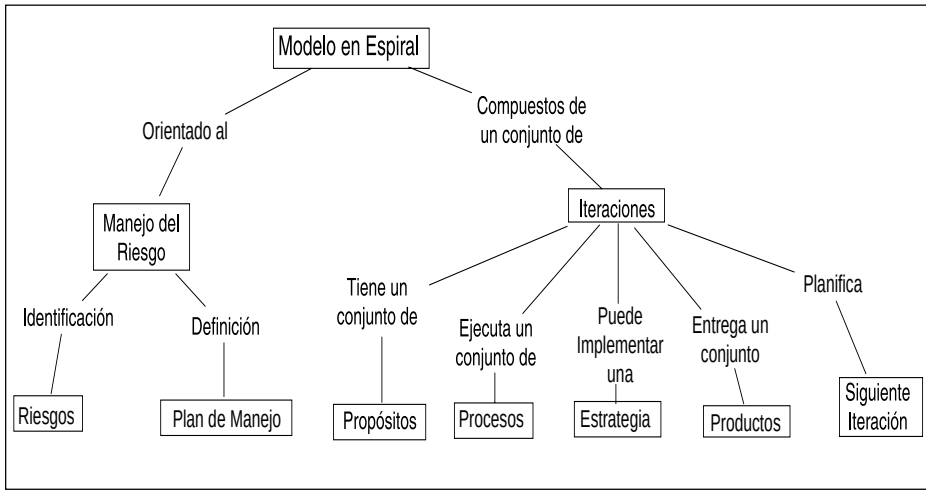
Estas prácticas claves en el desarrollo de software son implementadas en algunos métodos tales como Evo[2], el modelo en espiral y el proceso unificado (UP), siendo los dos últimos los que mayor trascendencia han tenido y serán explicados a continuación.

MODELO EN ESPIRAL

El modelo en espiral se centra en algunas prácticas fundamentales del desarrollo de software, tales como la orientación al manejo de riesgos, la orientación al cliente y el desarrollo iterativo (Ver Figura 3). El modelo se organiza en un conjunto de iteraciones que pueden considerarse a sí mismas como pequeños proyectos que siguen el ciclo de vida completo. Las primeras iteraciones tienen como objetivo identificar los riesgos del proyecto para determinar su viabilidad, y en caso de seguir adelante, definir un plan de manejo para mitigarlos o eliminarlos. Adicionalmente, el usuario participa activamente en la priorización de los casos de uso a desarrollar y en el proceso de pruebas, con lo cual se logra obtener una funcionalidad estable y operativa desde las primeras iteraciones del proyecto.

2. Evo, "Evolutionary Project Management". Probablemente el primer método Iterativo e Incremental de Desarrollo de Software, publicado en 1976. Creado por Tom Gilb. **Fuente:** Agile & Iterative Development [Graig Larman].

Figura 3. Mapa conceptual del modelo en espiral.

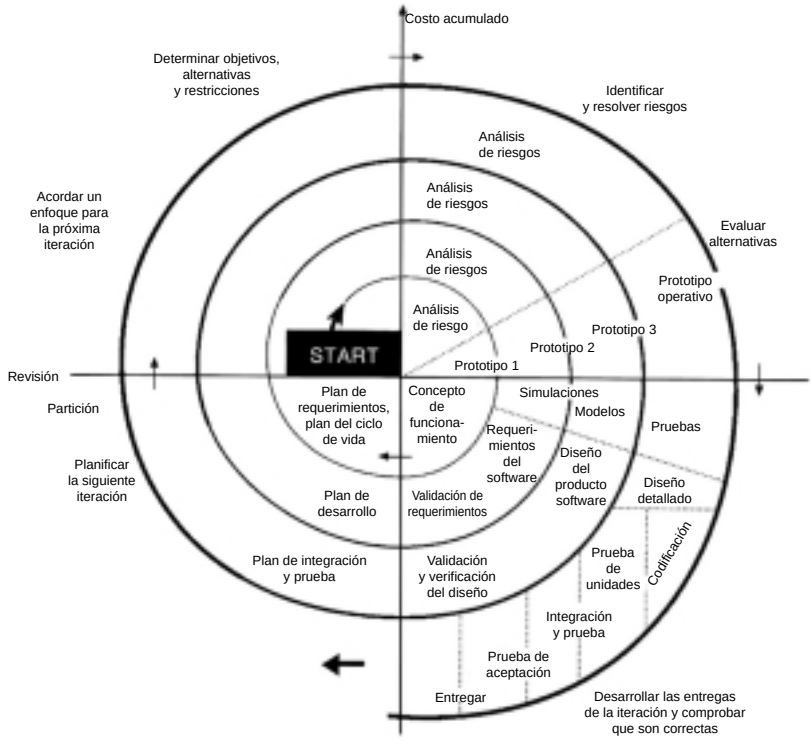


Fuente: Barry Boehm, "A Spiral Model of Software Development and Enhancement", ACM SIGSOFT Software Engineering Notes, August 1986.

El modelo en espiral tiene muchas ventajas respecto a los modelos anteriores por su orientación a la resolución temprana de riesgos, por la definición de la arquitectura del sistema en sus fases iniciales y por su proceso continuo de verificación de la calidad. En términos generales este modelo tiene un nivel alto de complejidad y requiere mucha destreza administrativa y experiencia por parte del gerente del proyecto y su grupo de trabajo para manejarlo adecuadamente.

Este modelo es ideal para manejar proyectos que requieran la incorporación de nuevas tecnologías, o para desarrollar productos completamente nuevos o con un nivel alto de inestabilidad de los requerimientos. Típicamente se maneja una iteración inicial donde se define el alcance del proyecto, se identifican y priorizan los riesgos y se realiza el modelo de casos de uso inicial para determinar qué casos de uso definirán la arquitectura del sistema. Con la información anterior se procede a definir el plan de manejo de riesgos según su prioridad, así como los casos de uso que serán implementados en las siguientes iteraciones (Ver Figura 4).

Figura 4. Modelo en espiral.



Fuente: Desarrollo y Gestión de Proyectos Informáticos. Steve McConnell, McGraw Hill.

ITERACIONES

Una iteración es una secuencia de actividades dentro de un plan establecido, con unos criterios claros de evaluación, que se organiza con el propósito de entregar parte de la funcionalidad del producto. En las primeras iteraciones se desarrollan o implementan los casos de uso que tienen mayor complejidad y que llevan inherente un alto nivel de riesgo que puede afectar el éxito del proyecto. De esta forma, con cada iteración que se realiza, los riesgos del proyecto se reducen acorde con el plan establecido, el cual está en permanente revisión para monitorear la posible aparición de nuevos riesgos y ajustarlo si es necesario.

La selección de los casos de uso para cada iteración se hace teniendo en cuenta cuál es el mínimo conjunto de ellos que se requiere para implementar la funcionalidad que mayor riesgo tenga. Se debe realizar este proceso hasta que la lista de riesgos haya sido cubierta completamente.

Cada iteración puede tener uno o varios propósitos, lo cual determina la duración de la misma; a su vez se ejecutan varios procesos que van desde

la especificación de requerimientos hasta las pruebas de unidad e integración, lo cual se asimila a un ciclo de vida en cascada. Adicionalmente, al final de cada iteración se deben evaluar los resultados del trabajo y se planea detalladamente la siguiente iteración.

INCORPORANDO EL PROCESO UNIFICADO DE DESARROLLO DE RATIONAL

La concepción de un sistema de información va mucho más allá de levantar los requerimientos, elaborar un conjunto de modelos y comenzar a programar. Esta concepción limitada ha permitido que durante años no podamos hacer uso adecuado de los conceptos y las herramientas con los que contamos. En este punto podemos considerar que la definición de la arquitectura del software se convierte en el eje orientador que permite controlar el desarrollo iterativo e incremental del sistema, a través de su ciclo de vida. Esta arquitectura se define en las primeras fases del proyecto, básicamente en la de elaboración, y se refina a través de todo el proyecto.

El RUP se fundamenta en seis prácticas: el desarrollo iterativo, la administración de requerimientos, la arquitectura basada en componentes, en el modelamiento visual, en la verificación continua de la calidad y la administración del cambio. Estas seis prácticas orientan el modelo y con ellas se pretende solucionar muchos de los problemas asociados al software. Adicionalmente hay muchos aspectos de diseño que son bien conocidos, pero que en realidad han sido muy poco implementados en los proyectos de software; estos son: facilidad de uso, modularidad, encapsulamiento y facilidad de mantenimiento. Es necesario entonces definir una arquitectura sólida basada en componentes, para construir mejores y más flexibles soluciones de software para las necesidades organizacionales.

Los cambios en un proyecto no pueden ser detenidos dado que la evolución del entorno de cada organización es continua, pero sí pueden ser administrados de manera que su impacto pueda ser estimado para determinar si dicho cambio se incluye o no y si el proyecto debe ser reajustado. Cada cambio en el proyecto debe tener especificado cuándo y cómo se va a realizar, quién lo va a hacer y qué productos se ven involucrados en ese cambio. En ese punto es donde el control de cambios y la trazabilidad de los componentes a través de los diversos modelos adquieren una gran importancia.

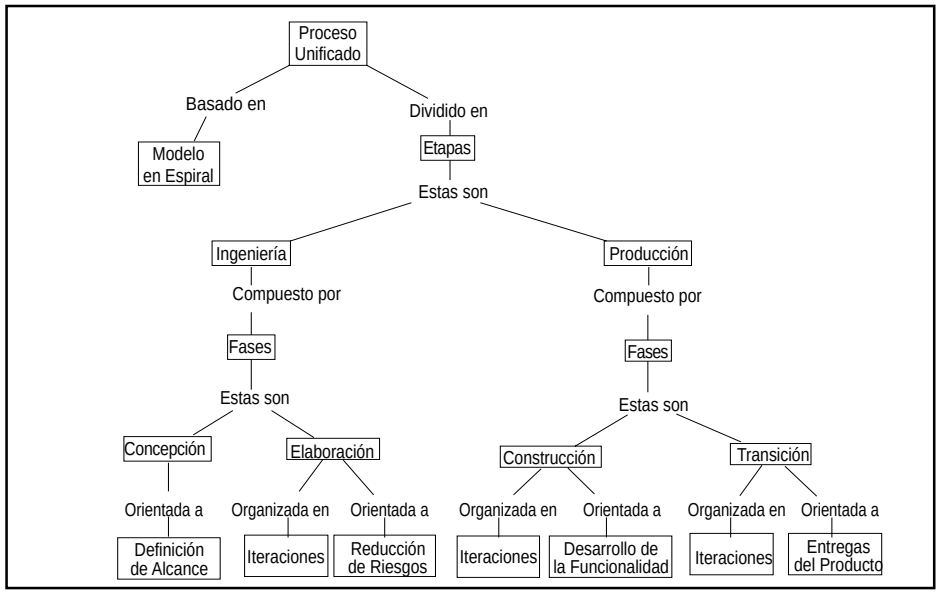
Existen algunos aspectos que se deben tener en cuenta para desarrollar exitosamente un proyecto. A continuación se enumeran algunos de ellos:

1. Se debe tener definida claramente la metodología de trabajo de cada fase del proceso del desarrollo de software, en especial las fases de administración de requerimientos y control de cambios, los cuales son los eslabones más débiles del proceso de desarrollo de software en nuestras organizaciones. La responsabilidad de definir, documentar y velar que se cumpla a cabalidad la metodología de trabajo es del grupo de ingeniería de procesos.

2. La participación activa de los usuarios y los acuerdos en los tiempos pactados, teniendo en cuenta los datos generados de los procesos de estimación y planificación, son responsabilidad del jefe del proyecto, pero deben ser elaborados con integrantes claves del equipo del proyecto.
3. El Grupo SQA[3] debe definir, documentar y actualizar el proceso de aseguramiento de la calidad del software, gestionar los recursos necesarios para que sea operativo desde el comienzo del proyecto, entregar el plan de calidad y velar por su cumplimiento a lo largo del ciclo de vida del proyecto.
4. El proceso de incorporación y utilización de nuevas tecnologías es quizás uno de los aspectos más críticos dentro del proyecto y de mayores riesgos. La definición de una metodología de administración del cambio tecnológico, clara y muy práctica, facilitaría considerablemente el trabajo realizado en la fase de elaboración, lo cual permitiría determinar la viabilidad de la incorporación de dicha tecnología en el proyecto.

Teniendo en cuenta los aspectos mencionados previamente, Rational que recientemente fue comprada por IBM, elaboró un marco de referencia para el proceso de desarrollo de software basado en el modelo en espiral. Este método se conoce como RUP "Rational Unified Process". Para una mejor organización, el RUP agrupa las iteraciones en etapas y fases que facilitan la administración del proyecto (Ver Figura 5).

Figura 5. Mapa conceptual del Proceso Unificado de Rational



Fuente: Principles of Managing Iterative Development v.2.0 Rational Software Corporation

3. SQA: Grupo de Aseguramiento de la Calidad.

ETAPAS DEL PROCESO UNIFICADO

1. Etapa de ingeniería

Esta etapa agrupa las fases de concepción y de elaboración, lo que básicamente le da por objetivos la conceptualización del sistema y el diseño inicial de la solución del problema.

Se inicia el proceso de administración de los requerimientos con la identificación y especificación de casos de usos, así como el proceso de aseguramiento de la calidad a través de los casos de prueba.

Se identifican los riesgos y se establece su plan de manejo, se ajusta ese plan según la tabla de priorización de riesgos y la de casos de usos vs. riesgos, para determinar en qué orden y en qué iteraciones se desarrollarán los artefactos de software que son la solución a los casos de uso.

Se identifican los recursos necesarios, tanto económicos como humanos, acordes con las necesidades del proyecto. Se da comienzo al proceso de estimación y planificación inicial a un nivel macro para todo el proyecto y posteriormente se realiza una estimación detallada de tiempos y recursos de las fases de concepción y elaboración.

1.1. *Fase de concepción*

Esta fase tiene como propósito definir y acordar el alcance del proyecto con los patrocinadores, identificar los riesgos asociados al proyecto, proponer una visión muy general de la arquitectura de software y producir el plan de las fases y el de iteraciones.

1.1.1 *Planeación de las fases y de las iteraciones*

A partir del modelo de casos de uso y de la lista de riesgos, se puede determinar qué casos de uso deben implementarse primero para atacar los riesgos de mayor exposición. Con base en la información previa se realiza el proceso de planificación general y un plan de trabajo detallado para la siguiente fase, así como el plan para la siguiente iteración.

Se debe establecer una relación clara y directa entre los casos de uso y los casos de prueba para facilitar que el proceso de aseguramiento de la calidad del software se ejecute adecuadamente. El plan de pruebas debe planearse en esta fase, ejecutarse desde la primera iteración de la fase de elaboración y refinarse sucesivamente durante el ciclo de vida del proyecto.

1.2. *Fase de elaboración*

Los casos de uso seleccionados para desarrollarse en esta fase permiten definir la arquitectura del sistema, se realiza la especificación de los casos de uso seleccionados y el primer análisis del dominio del problema, se diseña la solución preliminar del problema y comienza la ejecución del plan de manejo de riesgos, según las prioridades definidas en él.

Al final de la fase se determina la viabilidad de continuar el proyecto y si se decide proseguir, dado que la mayor parte de los riesgos han sido mitigados, se escriben los planes de trabajo de las etapas de construcción y transición y se detalla el plan de trabajo de la primera iteración de la fase de construcción.

2. Etapa de producción

En esta etapa se realiza un proceso de refinamiento de las estimaciones de tiempos y recursos para las fases de construcción y transición, se define un plan de mantenimiento para los productos entregados en la etapa de ingeniería, se implementan los casos de uso pendientes y se entrega el producto al cliente, garantizando la capacitación y el soporte adecuados.

2.1. *Fase de construcción*

El propósito de esta fase es completar la funcionalidad del sistema, para ello se deben clarificar los requerimientos pendientes, administrar el cambio de los artefactos construidos, ejecutar el plan de administración de recursos y mejoras en el proceso de desarrollo para el proyecto.

2.2. *Fase de transición*

El propósito de esta fase es asegurar que el software esté disponible para los usuarios finales, ajustar los errores y defectos encontrados, capacitar a los usuarios y proveer el soporte técnico necesario. Se debe verificar que el producto cumpla con las especificaciones entregadas por las personas involucradas en el proyecto al inicio del mismo.

CONCLUSIONES

En este artículo se presenta una visión general de algunos métodos orientados por las prácticas de desarrollo iterativo y manejo del riesgo.

Este tipo de métodos no han sido aplicados en muchas de las empresas locales, probablemente por su complejidad de administración, desaprovechando sus considerables ventajas respecto a los métodos tradicionales. Es necesario entonces desarrollar mecanismos de apropiación tecnológica más eficaces, que permitan mantener actualizadas nuestras prácticas organizacionales.

Los marcos de referencia aquí mencionados definen qué debe hacerse y en algunos casos cómo hacerlo. El trabajo del gerente de proyectos consiste en identificar cuál marco de referencia se ajusta a su organización y realizar el proceso de apropiación de la metodología y asimilación de las guías de trabajo al interior del equipo del proyecto y del área de informática.

BIBLIOGRAFÍA

Craig Larman. *Agile and iterative development: a manager's guide*. Addison Wesley, 2004.

Ivar Jacobson, Grady Booch y James Rumbaugh. *The Unified Software Development Process*. Rational Software Corporation. Addison-Wesley, 1999.

Steve McConnell. *Desarrollo y Gestión de Proyectos Informáticos*. McGraw Hill, 1996.

Rational Software Corporation. *Principles of Managing Iterative Development v.2.0*. 2001.

Project Management Institute PMI. *A Guide to the Project Management Body of Knowledge PMBOK® Guide*. 2000 Edition.

Rational Software Corporation. *RUP Rational Unified Process® Version 2002.05.00*, 2002.

Sandra Victoria Hurtado Gil. *Representación de la arquitectura de software usando UML*. Revista Sistemas

& Telemática No. 1 - Enero-Junio 2003. Universidad ICESI.

CURRÍCULO

Robin Alberto Castro Gil es Ingeniero de la Universidad Icesi, realizó la especialización en Gerencia de la Producción en la Universidad Icesi. Actualmente es Jefe de la Oficina de Desarrollo de Sistemas de la Universidad Icesi y presidente de la Asociación de Usuarios de Oracle de Colombia-ASUOC.