PROCESO DE DESARROLLO DE SOFTWARE

VERSIÓN 1.0

Basado en la articulación de

RUP y CMMI

priorizando su calidad

Carmen Inés Báez Pérez
Martha Isabel Suárez Zarabanda

Universidad de Boyacá
CENTRO DE INVESTIGACIONES PARA EL DESARROLLO " CIPADE"
FACULTAD DE CIENCIAS E INGENIERÍA

Báez Pérez, Carmen Inés
Proceso de desarrollo de software : basado en la articulación de RUP y CMMI priorizando su calidad / Carmen Inés Báez Pérez, Martha Isabel Suárez Zarabanda.-- Tunja :Universidad de Boyacá, 2013.
90 p. : il. ; 24 cm.
ISBN 978-958-8642-42-0
1. CMMI (Programa para computador) 2. RUP (Programa para computador) 3. Desarrollo de programas para computador 4. Desarrollo de software de aplicaciones I. Suárez Zarabanda, Martha Isabel II.Tít.
005.43 cd 21 ed.
A1422402

CEP-Banco de la República-Biblioteca Luis Ángel Arango

PROCESO DE DESARROLLO DE SOFTWARE

Basado en la articulación de RUP y CMMI priorizando su calidad

Martha Isabel Suárez Zarabanda
Ingeniera de Sistemas, Especialista en Telemática y Diplomada en Docencia Universitaria de la Universidad de Boyacá. Especialista en Entornos Virtuales de Aprendizaje de Virtualeduca en convenio con la OEI. Magíster en Dirección Estratégica y Tecnologías de la Información con el Centro Panamericano de Estudios Superiores de México. Directora de la División de Informática, Tecnología y Telecomunicaciones de la Universidad de Boyacá.

Carmen Inés Báez Pérez
Ingeniera de Sistemas, Especialista en Telemática y Diplomada en Docencia Universitaria de la Universidad de Boyacá. Magíster en Ciencias de la Información y las Comunicaciones de la Universidad Distrital de Bogotá. Directora del programa de Ingeniería de Sistemas de la Universidad de Boyacá.

Universidad de Boyacá
CENTRO DE INVESTIGACIONES PARA EL DESARROLLO " CIPADE"
FACULTAD DE CIENCIAS E INGENIERÍA
PROGRAMA INGENIERÍA DE SISTEMAS

RECTORA
Dra. Rosita Cuervo Payeras

VICERRECTOR ACADÉMICO
Ing. Rodrigo Correal Cuervo

VICERRECTOR DESARROLLO INSTITUCIONAL
Ing. Andrés Correal Cuervo

VICERRECTORA INVESTIGACIÓN,
CIENCIA Y TECNOLOGÍA
Ing. Patricia Quevedo Vargas

VICERRECTORA EDUCACIÓN VIRTUAL
Ing. Carmenza Montañez Torres

DIRECTORA
Centro de Investigaciones para el Desarrollo "CIPADE"
Ing. Patricia Quevedo Vargas

DISEÑO - DIAGRAMACIÓN - EDICIÓN DE CONTENIDOS
División de Publicaciones

Carrera 2ª Este N° 64-169
TELS.: (8) 7452742 - 7450000 Ext. 3106 - 3104

www.uniboyaca.edu.co / publicaciones@uniboyaca.edu.co

TUNJA - BOYACÁ - COLOMBIA

ISBN: 978-958-8642-42-0

Primera impresión: septiembre de 2013
Oferta nacional: 100

Segunda impresión: mayo de 2014
Oferta nacional: 150

Reimpresión: abril de 2015
Oferta nacional: 100

DOI:https://doi.org/10.24267/9789588642420

CAPÍTULO 3

CAPÍTULO 4

CAPÍTULO 5

LISTA DE FIGURAS

LISTA DE TABLAS

Presento con agrado a la comunidad académica, el libro Proceso de Desarrollo de *Software* Basado en la Articulación de RUP y CMMI priorizando su calidad, el cual es resultado de la experiencia docente e investigativa de las ingenieras Carmen Inés Báez Pérez Directora del programa de Ingeniería de Sistemas y Martha Isabel Suárez Zarabanda Directora de la División de Informática, Tecnología y Telecomunicaciones, pertenecientes al grupo de investigación Giprocasde la Facultad de Ciencias e Ingeniería de la Universidad de Boyacá.

Esta publicación comprende inicialmente la explicación en términos generales de la calidad en un proceso de ingeniería de *software*, aborda una revisión del RUP (Rational Unified Process), como marco de proceso de desarrollo de *software* y CMMI (Capability Maturity Model Integration), como modelo de métricas de calidad y madurez de dicho proceso.

Las investigadoras proponen una estrategia que integra a RUP y CMMI aplicado a productos de un proceso de desarrollo de *software*, con el fin de priorizar en la calidad, orientada al cumplimiento de estándares y normas tal como lo estable CMMI, esta consideración permite en el momento del desarrollo, contemplar lineamientos de calidad, construcción de artefactos definidos por el proceso propuesto, cumpliendo metas en cada fase y actividades, optimizándolo y garantizando sus productos, orientados a la solución de necesidades y requerimientos organizacionales y aportando al cumplimiento de sus objetivos misionales.

Ratificando el compromiso de la Universidad de Boyacá y el de sus docentes con la investigación de la región, se pone a consideración este producto investigativo, que incide en la formación de los ingenieros de sistemas y en los profesionales del área que se encuentran vinculados laboralmente en la rama de la ingeniería de *software*.

CLAUDIA PATRICIA QUEVEDO VARGAS
Vicerrectora de Investigación Ciencia y Tecnología

CAPÍTULO 1

CONCEPTUALIZACIÓN GENERAL

Objetivo General: Presentar aspectos generales utilizados en el área de la Ingeniería de *Software*, los cuales pueden ser útiles en la lectura del presente libro.

Hoy en día se consideran diferentes elementos para brindar asistencia a la evaluación realizada con el CMMI (Capability Maturity Model Integration). Algunos de ellos se empleaban por separado, antes de integrarse a este conjunto de modelos. A continuación se presenta una breve descripción de cada uno.

CALIDAD DEL *SOFTWARE*

Según R. Pressman[1], esta definición hace referencia a la "concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo *software* desarrollado profesionalmente".

[1] PRESSMAN, Roger. Ingeniería de *software*: Un enfoque práctico. 5 ed. México: McGraw-Hill, 2003, p. 125.

Al hablar de la calidad de un *software*, se hace referencia a su flexibilidad, eficiencia, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad. Por lo tanto, podría decirse que se trata del conjunto de sus propiedades y características, es decir, de las cualidades determinantes de su creación y utilidad.

Es posible medir dicha calidad después de elaborado el *software*. Naturalmente, en tal proceso se registran variaciones de acuerdo con el tipo de programa: no es lo mismo evaluar un sistema de un avión que un paquete contable. Ahora bien, los costos de la medición llegan a incrementarse y a ser muy elevados cuando se detectan problemas derivados de imperfecciones en el diseño. Incluso, el punto donde se localice un error puede obligar a mantener un control de la calidad durante todas las etapas del ciclo de vida del *software*.

GESTIÓN DE CALIDAD

Según las normas ISO 9000, esta gestión es un conjunto de actividades -cuya competencia corresponde generalmente a la dirección de las organizaciones- que determina la calidad, los objetivos y las responsabilidades en un contexto específico. Se implementa por medios como la planificación, el control, el aseguramiento (garantía) y el mejoramiento de los procesos, en un marco de optimización[2].

FACTORES QUE INCIDEN EN LA CALIDAD DEL *SOFTWARE*

Operatividad del producto

Las características operativas son[3]:

Corrección. ¿La aplicación hace lo que se le pide? ¿En qué medida satisface sus propias especificaciones y conlleva a la obtención de los objetivos encomendados por el cliente?

Fiabilidad. ¿Lo hace de forma fiable todo el tiempo? ¿Hasta qué grado puede esperarse un correcto cumplimiento de las operaciones preestablecidas?

---

[2] REYES, Percy. Construyendo *software* de alta calidad. [en línea]. (2005). [consultado 2 oct. 2006]. Disponible en <http://www.elguille.info/colabora/NET2005/Percy-net_ConstruyendoSoftCalidad.htm>

[3] Ibid.

Eficiencia. ¿Qué recursos de *hardware* y *software* son necesarios? ¿Cuántos de estos recursos precisa la aplicación para realizar las acciones con los tiempos de respuesta adecuados?

Facilidad de uso. ¿Es sencilla y cómoda de manejar? ¿En qué proporción se requiere un esfuerzo para aprender su manejo, trabajar con ella, introducir datos y conseguir resultados?

Integridad. ¿Puede controlarse su uso? ¿Hasta qué punto es posible regular el acceso al *software* o a los datos almacenados, e igualmente, impedir el ingreso de personal no autorizado?

Todo *software* satisface unas necesidades que se reflejan en sus requisitos funcionales y no funcionales. Estos requisitos se convierten en los objetivos de un proceso de producción, y por ende, de los artefactos generados.

El cumplimiento de tales exigencias reviste gran importancia, ya que éstas rigen el avance de un proceso y definen las condiciones de los productos resultantes, lo cual determina, en consecuencia, la complacencia misma del cliente. Es oportuno mencionar que existe la posibilidad de controlar dicho cumplimiento por medio de hitos definidos para cada una de las actividades integrantes de las distintas fases.

Cuando un usuario mide la calidad del *software* de acuerdo con la satisfacción de sus necesidades o al encontrar una adaptación del programa a sus expectativas, está valorando la fiabilidad misma de los productos generados. Asimismo, si las respuestas obtenidas coinciden con valores y datos considerados como óptimos para el proceso administrativo o de servicio implementado, podrá establecerse que el *software* arroja información confiable.

También cabe destacar que un usuario de un sistema de información requiere tiempos de respuesta óptimos y mínimos para el desarrollo normal de un proceso. De nada valdría un *software* que, a pesar de cumplir con todos los requerimientos, no tuviera en cuenta los lapsos pertinentes (el invertido por la persona en el manejo de las diferentes actividades y el utilizado por el programa para brindar soluciones). Estos tiempos deben marchar a la par con el factor seguridad, es decir, el cuidado que ha de tener el *software* con la información administrada, según sea el rol desempeñado por cada cliente en un determinado proceso y en un ambiente de fácil manejo de las aplicaciones.

Revisión del producto

En una revisión, los factores que describen la capacidad para soportar cambios son:

Facilidad de mantenimiento. Se refiere al esfuerzo requerido para detectar y reparar errores en una aplicación. ¿Pueden localizarse los fallos?

Flexibilidad. La exigencia necesaria para modificar su funcionamiento. ¿Es posible añadirle nuevas opciones?

Facilidad de prueba. El esfuerzo necesario para evaluarla de tal manera que se ajuste a lo especificado en los requisitos. ¿Pueden probarse todas las opciones?

La capacidad de un *software* para soportar cambios es una condición significativa en la medición de su calidad. En todas las fases de producción de un programa se presentan transformaciones, ya sea porque no se realizó una adecuada ingeniería o debido a variaciones de los requerimientos del usuario. Las modificaciones pueden ocasionar fallos perceptibles para el usuario final, aunque también generan iteraciones que permiten corregir y evolucionar.

Transición del producto

Los factores que describen la adaptabilidad a nuevos entornos son:

Portabilidad. Es el esfuerzo requerido para transferir la aplicación a otro hardware u otro sistema operativo. ¿Puede trasladarse a distintas máquinas?

Reusabilidad. Hace referencia a la probabilidad de incorporar sus partes a diferentes aplicaciones. ¿En qué medida es posible efectuar esta reutilización?

Interoperabilidad. Corresponde a la exigencia necesaria para comunicarla con otras aplicaciones o sistemas informáticos. ¿Puede realizarse esta comunicación?

Claro está que la elaboración de un *software* no termina con su entrega al usuario final. Por el contrario, en este punto comienza una etapa de "confianza" entre desarrolladores y clientes, en la cual estos últimos esperan un apoyo tecnológico durante la utilización del producto. Así, es factible descubrir fallas o evidenciar nuevas exigencias que demandan modificaciones, actualizaciones o ejecuciones del proceso producción.

MÉTRICAS DE LA CALIDAD DEL *SOFTWARE*

La evolución de la industria del *software*, al igual que ocurre en todas las actividades con una dinámica de negocios, ha traído consigo una variedad de objetos, mercados y competencias. Tal realidad ha obligado a crear estándares con el fin de garantizar procesos óptimos y productos de excelencia. De este modo, se estructuran los marcos referenciales destinados a asegurar al cliente la adquisición de programas de calidad, que atiendan sus requisitos y exhiban un correcto funcionamiento.

En el contexto de estos estándares, cuando se miden los resultados de un proceso de producción de *software*, es indispensable tener en cuenta variables que inciden en los resultados, ya sea positiva o negativamente. Ante todo, es preciso examinar el programa desde la óptica de los conceptos que al serle aplicados le confieren una impresión favorable.

Los estándares, traducidos como métricas, se han concentrado en puntos claves que garantizan el producto evaluado y le brindan al usuario las coordenadas necesarias para escoger aquel que se ajusta a sus supuestos. Éstas métricas son[4]:

Métricas del *software*. Funcionalidad, complejidad, eficiencia. Se relacionan con el desarrollo mismo.

Métricas técnicas. Al centrarse en las características del *software* (complejidad lógica, grado de modularidad) miden su estructura y evalúan cómo está hecho.

Métricas de calidad. Indican cómo se ajusta a los requisitos implícitos y explícitos del cliente.

Métricas de productividad. Se enfocan en el rendimiento del proceso de ingeniería. Por consiguiente, indican qué tan productivo puede llegar a ser el *software* a diseñar.

Métricas orientadas a la persona. Se aplican al personal que desarrolla el sistema. Proporcionan información, especialmente desde el punto de vista humano, concerniente a la efectividad de las herramientas y metodologías utilizadas.

[4] PÁEZ, Otoniel. Métricas, estimación y planificación en proyectos de software. [en línea]. [consultado 2 oct. 2006]. Disponible en <http://www.willydev.net/descargas/WillyDEV_PlaneaSoftware.Pdf>

Métricas orientadas al tamaño. Se asocian con el proceso de producción. Permiten establecer el tiempo y el número de personas que se requieren para terminar el *software*.

Métricas orientadas a la función. Estas medidas indirectas señalan la funcionalidad o utilidad del programa.

PROCESO

Un proceso es un conjunto de prácticas que se ejecutan con un propósito determinado y permiten transformar entradas en salidas de gran valor. Usualmente, abarca herramientas, métodos, materiales y personas[5].

Modelo de Procesos

Este modelo es un conjunto estructurado de elementos que describen características de procesos efectivos y de calidad. Indica "qué hacer", no "cómo hacer", ni "quién lo hace". Proporciona diversos beneficios: un punto de partida, los conocimientos obtenidos en experiencias pasadas, un lenguaje común, una visión compartida o un marco para priorizar acciones. Los procesos incluidos en este modelo son aquellos sobre los cuales la experiencia ha demostrado su efectividad, por lo tanto, esta es una forma de puntualizar el significado de "mejora" para la organización.

Puntos claves del capítulo

- Para lograr el éxito en todo proceso cuyo fin sea generar *software*, es preciso incluir acciones de gestión y administración, con tópicos de calidad que garanticen el cumplimiento de los requerimientos.
- La aplicación del concepto de métrica provee una visión general del proceso orientado al óptimo desarrollo del *software*.

[5] HUAROTO, Concha. Propuesta para implantar CMMI en una empresa con múltiples unidades desarrolladoras de *software*. [en línea]. Lima: Universidad Mayor de San Marcos, 2005, p. 15. [consultado 27 mar. 2008]. Disponible en <http://sisbib.unmsm.edu.pe/BibVirtualData/Tesis/Basic/concha_hn/cap2.pdf>

CAPÍTULO 2

MODELO CMMI

Objetivo General: Proporcionar los conceptos generales del CMMI, un componente básico del proceso de producción de *software*.

El SEI (*Software* Engineering Institute, Instituto de Ingeniería de *Software*) publicó en 1993 el Modelo de Madurez de Capacidad (CMM) con el cual buscaba mejorar los procesos relacionados con el desarrollo de *software*. Posteriormente, el Departamento de Defensa de Estados Unidos exigió a sus proveedores la acreditación en CMM, circunstancia que convirtió al modelo en un estándar de calidad dentro de la industria del *software*. Por su parte, el CMMI es una evolución del CMM que integra diferentes modelos de calidad:

- Capability Maturity Model for *Software* (CMMI- SW).
- Electronic Industries Alliance Interim Standard (EIA/IS).
- Integrated Product Development Capability Maturity Model (IPD-CMM).

CONCEPTO GENERAL

El CMMI es un conjunto de modelos que permite conocer la madurez de los procesos relacionados con las tecnologías de la información en una organización. Mediante este método es posible identificar las áreas con deficiencias y corregir los errores, lo cual conlleva a la definición de proyectos de buena calidad. El CMMI clasifica a las empresas en cinco niveles que establecen la madurez de las acciones efectuadas en ellas en lo concerniente a la producción de *software*.

Al integrar distintas funciones y fijar metas y prioridades orientadas a mejorar los procesos, oficia como una guía respecto a la dirección que éstos han de llevar. Asimismo, brinda puntos de referencia valiosos para la valoración de las acciones desarrolladas.

CATEGORÍAS DEL CMMI

El CMMI trabaja con cuatro categorías:

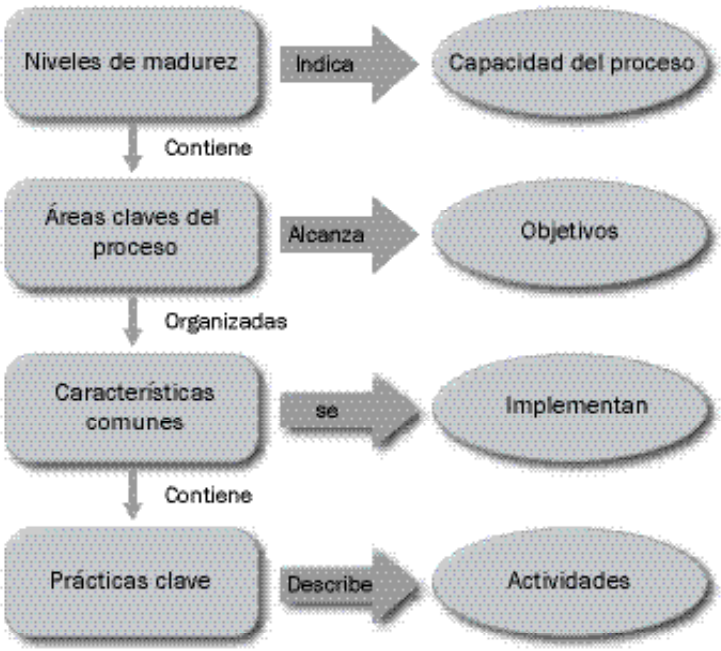
Ingeniería de sistema. Cubre la construcción de un sistema con o sin *software*.

Ingeniería de software. Abarca la producción de soluciones *software*.

Integración de productos y procesos de desarrollo. Comprende la relación a largo plazo con el cliente.

Relación con proveedores. Se refiere a las dinámicas relacionadas con la subcontratación de partes para el sistema.

Figura 1. Esquema del CMMI



Fuente: PHILLIPS, Mike y KONRAD, Mike. Beyond CMMI. [en línea]. [consultado 2 oct. 2006]. Disponible en http:// <www.sei.cmu.edu>

CMMI-SW

La escasa aplicación de técnicas de ingeniería y la falta de planeación en el desarrollo de los programas, son las principales razones que conllevan a producir y entregar *software* con errores, en plazos superiores a los planificados y con costos por fuera de lo estimado.

Ante esta realidad, la aplicación de un modelo de desarrollo de *software* en una organización constituye un cambio de enfoque necesario, centrado en una premisa esencial: hay que centrarse en el resultado, no en los procesos, pues éstos, conjuntamente con la tecnología y las personas, conforman la base fundamental para lograr un producto de calidad.

En este sentido, el CMMI-SW ofrece una mayor cobertura de las distintas áreas de los procesos y agrega el concepto de representación continua. Sus objetivos son estudiar el desarrollo del *software* en una organización, al igual que evaluar, según una escala de niveles, su madurez. Esta última condición, al actuar como un indicador de la capacidad para construir programas de calidad, proporciona información valiosa en el camino hacia la optimización de las acciones emprendidas por las empresas en la producción de *software*.

Así pues, con este conjunto de modelos se busca determinar la madurez de los procesos relacionados con *software* y tecnologías de la información en una organización. De tal modo, se detectan las falencias existentes en diversas áreas, se corrigen los posibles errores y se mejora el *software*. Además, gracias a las métricas se definen los costos y se estima el tiempo exacto de elaboración de un proyecto.

Es oportuno anotar que en algunas ocasiones, la aplicación de un modelo convencional resulta insuficiente debido a su poca efectividad, o peor aun, hay casos en que no se maneja modelo alguno.

Ventajas de mejorar el proceso

Ya que la calidad de un sistema se basa en la calidad del proceso, la mejora de este último conlleva a la optimización del primero. Dicho perfeccionamiento es aplicado en la obtención de los objetivos del negocio, por consiguiente, guarda correspondencia con las metas de mejoramiento[6].

VENTAJAS DEL CMMI-SW

Este conjunto de modelos ofrece las siguientes ventajas[7]:

- Liga las actividades de ingeniería y gerencia con los objetivos de negocio.
- Al brindar una visión clara sobre las acciones del ciclo vital y de la ingeniería del producto, posibilita el cumplimiento de las expectativas del cliente.
- Integra lo aprendido por medio de la experiencia de otras áreas.

---

[6] *SOFTWARE* ENGINEERING INSTITUTE (SEI). Beyond CMMI v1.2. [en línea]. [consultado 2 oct. 2006]. Disponible en <http// www.sei.cmu.edu>

[7] Ibid.

- Incorpora funciones adicionales de organización para los productos y servicios.
- Solo aplica estándares ISO de importancia.

NIVELES DE MADUREZ

Los cinco niveles de madurez manejados por el CMMI-SW son:

Nivel 1: Inicial

- Los resultados de calidad son consecuencia de las personas y de las herramientas involucradas.
- Se efectúa un acercamiento intuitivo al desarrollo del *software*.
- Es preciso crear medidas que sirvan para estimar y planificar el futuro.

Nivel 2: Repetible

- Se emprenden prácticas básicas de gestión de proyectos y requisitos, al igual que de control de versiones. Asimismo, se adoptan elementos de trabajos ejecutados por subcontratistas en proyectos de características similares.
- Se evalúa el tamaño funcional o físico del sistema, como también los recursos, esfuerzos, costos y el calendario de labores.

Nivel 3: Definido

- Es factible conocer la forma cómo se construyó el sistema.
- Los procesos de gestión e ingeniería, es decir, aquellos que son comunes para el desarrollo y mantenimiento del *software*, se documentan y estandarizan.
- Se realizan mediciones y análisis de las actividades con el propósito de detectar eventuales errores.

Nivel 4: Gestionado

- Se evalúa la efectividad de todas las actividades.
- Se mide detalladamente la calidad del producto y del proceso. Esto se verifica de forma cuantitativa, con base en las métricas establecidas, en una dinámica que desde los primeros momentos del proyecto acude a la realimentación con el fin de seleccionar prioridades en las prácticas y conocer la manera como se emplean los recursos.

Nivel 5: Optimizado

- Existe una mejora continua de los procesos.
- Las mediciones de las actividades son usadas para alcanzar esa mejora. Por consiguiente, como respuesta a los resultados obtenidos, se eliminan y añaden acciones, mientras que la estructura de algunas de ellas es reorganizada.

ÁREAS DE PROCESO

Figura 2. Áreas de proceso CMMI

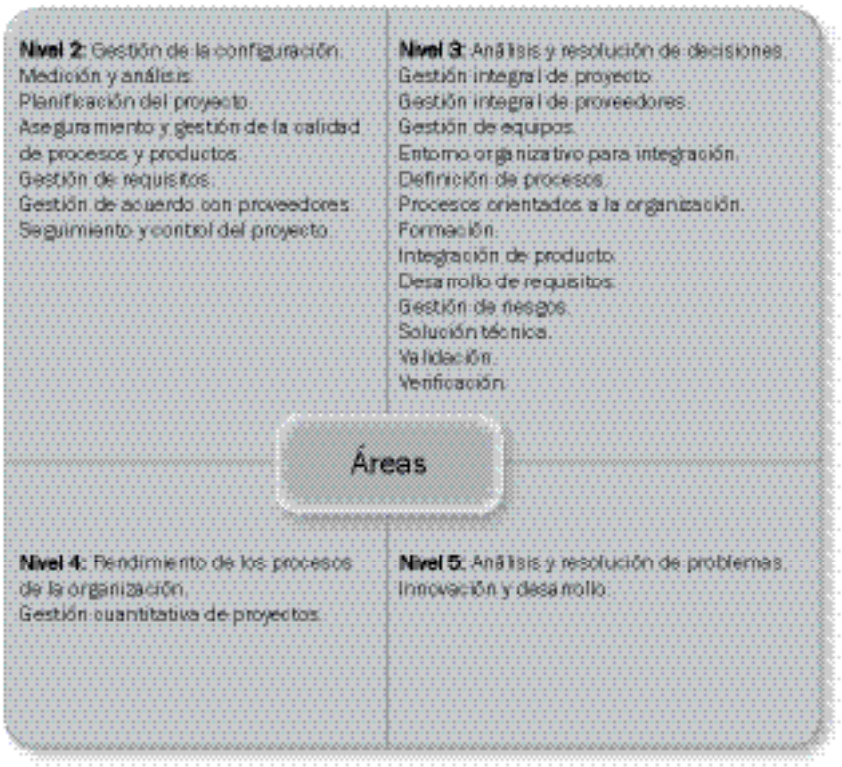
Fuente: las autoras

Las áreas de proceso son prácticas vinculadas entre sí que se ejecutan de forma conjunta para conseguir determinados objetivos[8]. Igualmente, pueden entenderse como un árbol de acciones relacionadas que al ejecutarse conjuntamente, satisfacen un sistema de metas significativas para la mejora sustancial de un área específica. Es común apreciar las áreas de proceso CMMI-SW en dos representaciones: escalonada y continua.

[8] Ibid.

La figura 3 presenta una visión general de los aspectos del proyecto que se mejoran en cada uno de los niveles del CMMI, dado que independientemente del nivel, siempre se trabaja sobre las mismas áreas.

Figura 3. Vistas del proceso CMMI

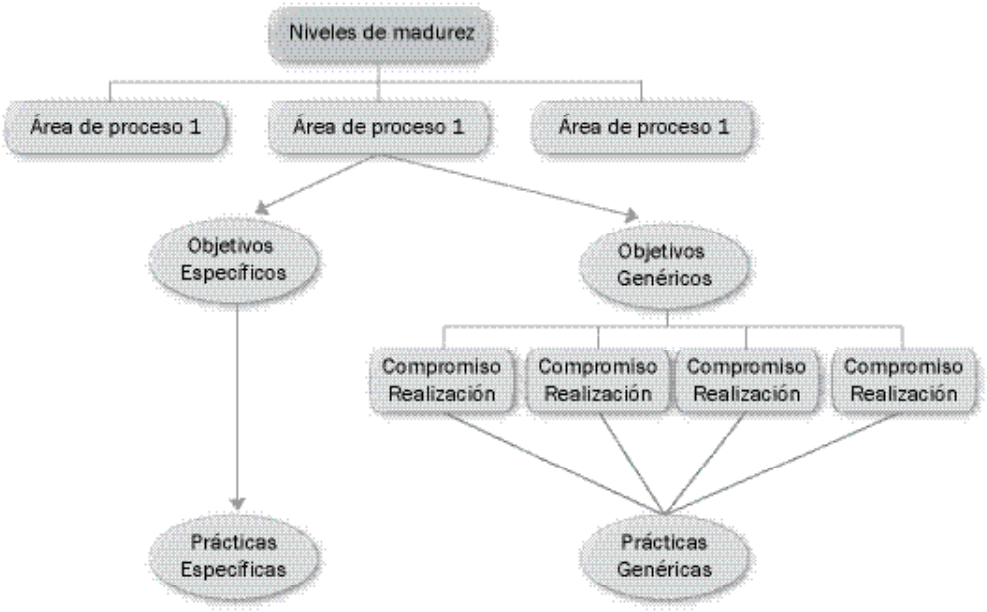


Fuente: SOFTWARE ENGINEERING INSTITUTE (SEI). Beyond CMMI v1.2. Adaptado por las autoras.

Representación escalonada

En el modelo escalonado o centrado en la madurez, el CMMI-SW establece cinco niveles para clasificar a las organizaciones. Esta categorización se realiza en función de los objetivos conseguidos por cada área de proceso, de acuerdo con una gestión basada en principios de ingeniería. Se trata, por lo tanto, de un modelo de evaluación y mejora[9].

Figura 4. Representación escalonada del modelo



Fuente: PALACIO BAÑERES, Juan. Sinopsis de los modelos CMM y CMMI

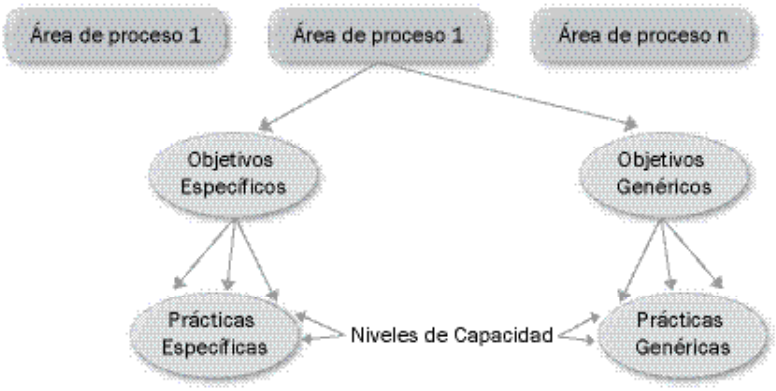
Representación continua

El modelo por etapas o de representación continua, está enfocado en mejorar la capacidad de los procesos de una empresa. Esta capacidad esperada, cabe precisar, se encuentra contenida en los niveles de madurez, cada uno de los cuales provee los fundamentos pertinentes para optimizaciones futuras. Dichos principios, a su vez, se agrupan en cuatro categorías de acuerdo con su finalidad: Gestión de proyectos, Ingeniería, Gestión de procesos y Soporte a otras categorías[10].

[9] *Ibid.*

[10] *HUAROTO, Op. Cit., p. 1.*

Figura 5. Representación continua del modelo



Fuente: Fuente: PALACIO BAÑERES, Juan. Sinopsis de los modelos CMM y CMMI

USOS DEL CMMI-SW

El modelo puede aplicarse desde dos perspectivas:

- Vista de evaluación: comprende la búsqueda de los componentes mínimos exigidos para satisfacerlo.
- Vista de mejora de proceso: corresponde a la búsqueda de lo más adecuado para la organización, es decir, de todo aquello que potencie su optimización.

Componentes requeridos

Metas Genéricas (GG)

Establecen lo que una organización debe alcanzar en un nivel de capacidad específico. La obtención de tales propósitos en un área de proceso, significa mejorar la ejecución de la misma[11].

Metas Específicas (SG)

Abarcan características únicas cuya ejecución es necesaria para satisfacer un área de proceso. Se trata entonces de componentes requeridos por el modelo, los cuales se utilizan en distintas valoraciones para determinar si un área en particular se ha cumplido debidamente[12].

[11] *HUAROTO, Op. Cit., p. 3.*

[12] *HUAROTO, Op. Cit., p. 3.*

Componentes esperados

Prácticas Genéricas (GP)

Una práctica de este tipo se aplica a cualquier área, pues permite mejorar el funcionamiento y el control de cualquier proceso.

Prácticas Específicas (SP)

Son actividades de importancia para la realización de la meta asociada. Al ser componentes esperados del modelo, describen las acciones proyectadas para lograr los objetivos de un área de proceso[13].

Prácticas Típicas

Son componentes informativos del modelo que proporcionan salidas a las prácticas, tanto específicas como genéricas. A estas salidas se les denomina ejemplos o productos típicos del trabajo. Es pertinente indicar que, a menudo, distintos productos resultan eficaces, pero no son tomados en cuenta[14].

Sub-prácticas

Son descripciones detalladas que brindan orientaciones apropiadas para interpretar las prácticas específicas y las genéricas. Por lo tanto, poseen un carácter de componentes informativos. De ellas solo se toman las ideas útiles para la mejora del proceso[15].

EL NIVEL 2 DE LAS ÁREAS DE PROCESO EN EL CMMI-SW

Este nivel es el más importante del CMMI, ya que corresponde a la planeación de todo cuanto va a realizarse. De hecho, lo concebido desde este ámbito resulta de gran valía para el progreso del análisis, del diseño y de la programación. Si cada procedimiento ha sido planificado y se encuentra suficientemente documentado, en la eventualidad de que surjan inconvenientes es mucho más factible descubrir dónde se registró el error, como también plantear alternativas para solucionarlo. A continuación se describen las áreas de proceso de este nivel.

---

[13] *HUAROTO, Op. Cit., p. 4.*

[14] *HUAROTO, Op. Cit., p. 5.*

[15] *HUAROTO, Op. Cit., p. 6.*

Gestión de requisitos

Los objetivos de esta área son, por un lado, gestionar los requisitos y componentes del proyecto, y de otra parte, detectar debilidades en el plan y en los elementos de trabajo.

Las prácticas específicas, en relación con los requisitos, son:

1. Entenderlos.
2. Lograr un compromiso con ellos.
3. Gestionar sus cambios.
4. Identificar inconsistencias entre ellos y el trabajo del proyecto.

Planificación del proyecto

Busca concretar los planes que han de definir el presupuesto, el cronograma y las actividades del proyecto. Ahora bien, el programa adoptado debe actualizarse frecuentemente, pues los requerimientos cambian de manera constante.

Las metas específicas son:

1. Establecer estimaciones. A su vez, las prácticas específicas de esta meta son:
   a. Estimar el alcance del proyecto.
   b. Realizar cálculos del producto del trabajo y de los atributos de tarea.
   c. Definir el ciclo de vida del proyecto.

2. Desarrollar el plan. En este punto es preciso elaborar el documento que se empleará para manejar y controlar la ejecución del proyecto. Las prácticas a efectuar son:
   a. Conformar el presupuesto y el calendario.
   b. Identificar y analizar los riesgos.
   c. Planear la gestión de la información.
   d. Planificar los recursos.
   e. Formular el plan.

3. Consolidar compromisos -con las personas y las actividades involucradas- para poner en marcha el plan.

Son dos las prácticas emprendidas:

a. Revisar los planes que afectan al proyecto.

b. Crear un plan de responsabilidades.

Seguimiento y control del proyecto

Cuando el curso del proyecto se desvía del plan trazado, es preciso examinar el estado de todos los componentes para tomar los correctivos apropiados.

Las metas a cumplir son:

1. Seguimiento del proyecto respecto al plan. Sus prácticas son:

   a. Estructuración de los parámetros de seguimiento.

   b. Seguimiento de las responsabilidades.

   c. Seguimiento de los riesgos.

   d. Seguimiento de la gestión de la información.

2. Gestión de los correctivos. Estas medidas se adoptan cuando los resultados no son los esperados. Sus prácticas son:

   a. Analizar las tareas.

   b. Tomar acciones correctivas.

   c. Gestionarlas.

Gestión de acuerdo con los proveedores

Este proceso busca auditar las compras de productos a los proveedores con quienes existen convenios. Sólo se cumple si se verifica una práctica, como es acordar un acuerdo con los aprovisionadores. Para esto es necesario:

1. Determinar el tipo de adquisición.

2. Seleccionar los proveedores.

3. Lograr un acuerdo con ellos.

Medición y análisis

Consiste en implantar un método de medición que ayudar a cubrir las necesidades de información de la gerencia.

Las metas a cumplir son:

1. Alinear las mediciones y actividades de análisis. Para lograr esto, deben desarrollarse las siguientes prácticas:
   a. Puntualizar los objetivos de medición.
   b. Especificar las medidas.

2. Proporcionar resultados de medición. Las prácticas que han de ser aplicadas son:
   a. Recoger mediciones de los datos.
   b. Almacenar los datos.
   c. Comunicar los resultados.

Aseguramiento y gestión de la calidad de productos y procesos

Para tener certeza de que los procesos y elementos de trabajo discurren de manera idónea, la ejecución de cada uno de ellos debe ser evaluada objetivamente y por medio de acciones de gestión. En este sentido, debe alcanzarse una meta específica, como es analizar con ecuanimidad tales componentes, para lo cual se recurre a dos prácticas:

   a. Valorar los procesos.
   b. Examinar los servicios y productos del trabajo.

Gestión de la configuración

Esta área busca mantener la integridad y el control de los elementos de trabajo. Para tal fin, se consideran dos metas:

1. Establecer la línea base. Sus prácticas son:
   a. Identificar los elementos de la configuración.

b. Consolidar un sistema de gestión de la configuración.

c. Poner en marcha las líneas base.

2. Supervisar y controlar los cambios. Sus prácticas son:

a. Enunciar las peticiones de supervisión de cambios.

b. Controlar los elementos de la configuración.

PUNTOS CLAVE DEL CAPÍTULO

- Todo proceso de producción de *software* debe pensarse siempre en función de la gestión de la calidad.
- Definir las métricas de calidad de *software* conlleva a la obtención de un producto en que se cumplen totalmente los requisitos de calidad exigidos.

CAPÍTULO 3

PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE RUP

Objetivo General: Proporcionar los conceptos generales del RUP, un componente básico del proceso de producción de *software*.

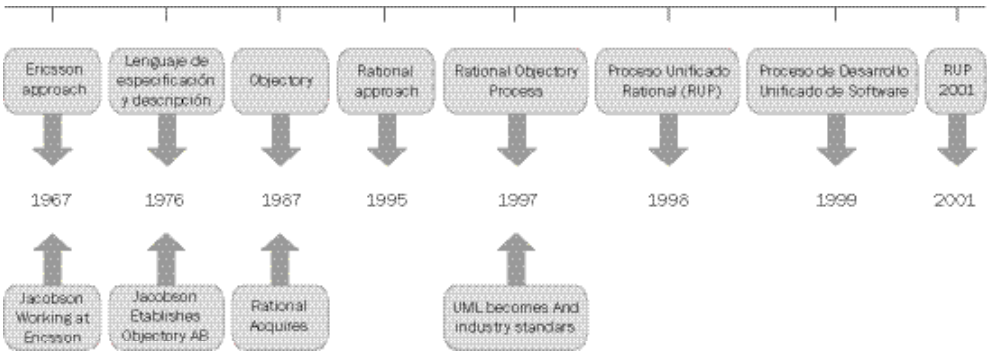
En este capítulo se abordan antecedentes y generalidades del RUP (Rational Unified Process). Se presentan las características más sobresalientes que lo han convertido en un proceso ampliamente difundido, por medio del cual el paradigma de la orientación a objetos se ha involucrado en todo el desarrollo de *software*.

ANTECEDENTES

La evolución de RUP ha estado determinada por aportes claves que han influido en la definición y caracterización de este proceso de producción. Dichas contribuciones se revisan a continuación.

RUP implementa las mejores prácticas de producción de *software* orientadas a objetos, propuestas por autores como Grady Booch, James Rumbaugh e Ivar Jacobson, quienes unificaron conceptos de distintas metodologías. Al haber sido una fusión de diversos fundamentos, es necesario resaltar los antecedentes que dieron paso a su surgimiento.

Figura 6. Evolución de RUP



Fuente: LETELIER, P. Rational Unified Process (RUP)

Como se aprecia en el gráfico de la figura 6, el primer precedente en la historia de RUP aparece en 1967 con la metodología Ericsson (Ericsson Approach), expuesta por Ivar Jacobson, uno de los principales artífices de la consolidación del UP (Unified Process o Proceso Unificado). Se trató de una propuesta para realizar un desarrollo basado en componentes, una aproximación a los casos de uso, los cuales son empleados "para capturar el comportamiento deseado del sistema en desarrollo, sin tener que especificar cómo se implementa ese comportamiento"[16]. La idea de utilizar tales casos para describir los requisitos funcionales de un proceso de producción de *software*, fue introducida en 1986 por el mismo Jacobson, y desde entonces, ha alcanzado gran ascendiente y reconocimiento, en especial por su simplicidad y utilidad[17].

Otro momento significativo fue el surgimiento, en 1994, de UML (Unified Modeling Languaje, Lenguaje Unificado de Modelado), iniciativa de James Rumbaugh y Grady Booch, quienes combinaron los métodos expuestos por este último con la Técnica de Modelado de Objetos (Object Modeling Tehcnique, OMT)[18]. Posteriormente, el

[16] GUERRERO, Luis. Taller de UML. [en línea]. Santiago de Chile: Universidad de Chile, 2002, p. 2. [consultado 19 ago. de 2008]. Disponible en <http://www.dcc.uchile.cl/~luguerre/cc61j/recursos/clase2.ppt>

[17] LARMAN, Craig. UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado. 2 ed. Madrid: Pearson Educación, 2003, p. 44.

[18] Ibid., p. 45.

Rational Objectory Proccess introdujo, en 1996, el concepto de desarrollo iterativo. En este enfoque, el proceso se organiza en una serie de mini-proyectos cortos y de duración fija (por ejemplo, cuatro semanas) llamados iteraciones. El resultado de cada uno de ellos es un sistema que puede ser probado, integrado y ejecutado, pues incluye sus propias actividades de análisis de requisitos, como también de diseño, implementación y pruebas[19]. Finalmente, en 1998 se describió la manera de aplicar los diagramas UML en la construcción de sistemas basados en objetos.

La suma de los antecedentes señalados dio como resultado el nacimiento de RUP en 1998. En las versiones posteriores a la de aquel año, se han ido implementando prácticas de la más alta calidad para la producción de *software*.

FILOSOFÍA

A partir de una base de características generales, en el ámbito de la Ingeniería de *Software* se han originado, definido, implementado, usado y validado procesos orientados hacia la producción de diferentes programas. Al respecto, es procedente considerar algunos puntos y conceptos en aras de conocer mejor y con mayor amplitud esta evolución.

Ante todo, en relación con esas características comunes, las planteadas por Varas resultan ilustrativas: "1) El *software* se desarrolla, 2) No se fabrica en un sentido clásico ni se 'estropea', 3) La mayoría del *software* se desarrolla a medida, en vez de ensamblar componentes existentes"[20].

Por su parte, la Ingeniería de *Software* se define como "la aplicación de un planteamiento sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento de *software*"[21].

---

[19] Ibid., p. 47.

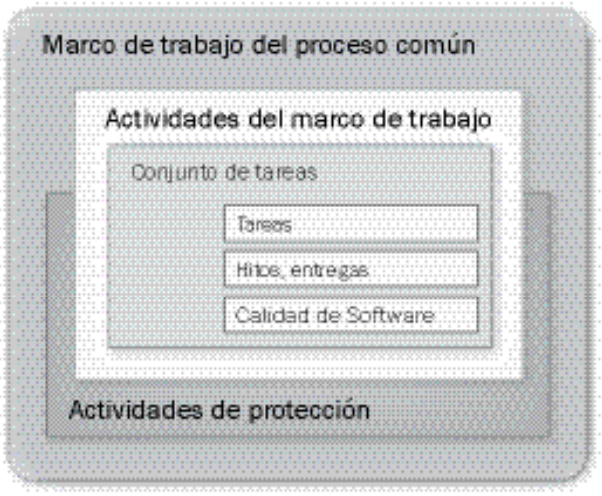
[20] VARAS, Marcela. Gestión de Proyectos de Desarrollo de *Software*. [en línea]. Concepción: Universidad de Concepción, 2000, p. 3. [consultado 19 ago. de 2008]. Disponible en http://www.inf.udec.cl/~mvaras/gpis/apunteGPDS.pdf

[21] PINO, Francisco, et al. Contribución de los estándares internacionales a la gestión de los procesos de *software* [en línea]. (2006). [Consultado 16 jul. 2008]. Disponible en: <http://alarcos.inf-cr.uclm.es/competisoft/publico/downloads/Inf_T%C3%A9cnicos/COMPETISOFT_IT%202_Contribuci%C3%B3n%20de%20los%20est%C3%A1ndares%20internacionales%20a%20la%20gesti%C3%B3n%20de%20procesos%20software.pdf>

A su vez, en este contexto se entiende por proceso "un conjunto de actividades técnicas y administrativas realizadas durante la adquisición, desarrollo, mantenimiento y retiro de *software*"[22].

Así, la evolución de la Ingeniería de *Software* ha producido una transformación del concepto de proceso. En la actualidad, se le concibe como un marco de trabajo que determina una serie de tareas, las cuales arrojan resultados denominados artefactos. Éstos pueden ser "vigilados" por medio de hitos o puntos de control, de acuerdo con ciertas normas de calidad y en condiciones que permiten su adaptación a cualquier tipo de proyecto sin importar su complejidad, tamaño, área de aplicación y equipo de trabajo involucrado.

Figura 7. El proceso de *software*



Fuente: PRESSMAN, Roger. Ingeniería de Software: un enfoque práctico, p. 20.

Tras revisar los puntos anteriores, es posible afirmar que distintos planteamientos teóricos se han implementado de manera práctica para apoyar el desarrollo de *software.* Tal es el caso del recorrido seguido para posibilitar la consolidación de RUP, un proceso reconocible actualmente por sus características y estructura.

[22] Ibid.

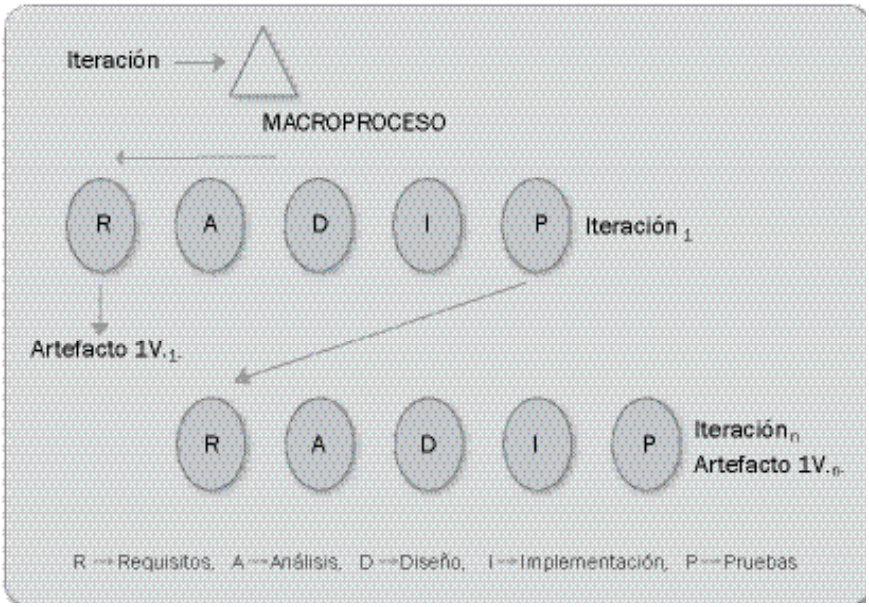
PRINCIPALES CARACTERÍSTICAS

RUP presenta características que lo diferencian de otros procesos de producción de *software*. Entre las principales se encuentran: es iterativo e incremental, se dirige por casos de uso, y está centrado en la arquitectura.

Iterativo e incremental

RUP implementa soluciones escalonadas o por pasos, llamadas iteraciones. Éstas se entienden como la realización de una o más disciplinas o flujos de trabajo, acción que conlleva a mitigar los riesgos y a cumplir algunos requisitos, funcionales o no.

Figura 8. Definición gráfica del término iterativo en RUP

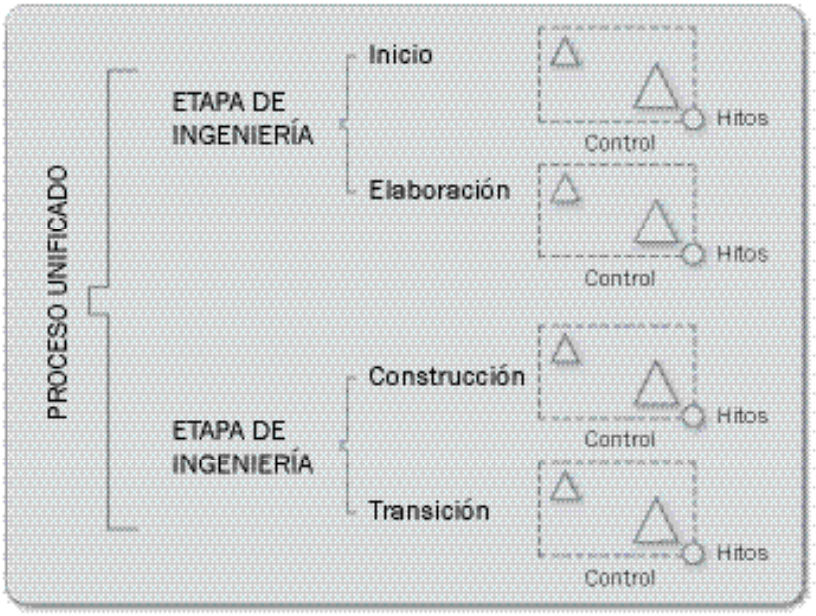


Fuente: las autoras

El carácter mismo de las iteraciones implica un incremento del número de artefactos que deben desarrollarse en las diferentes fases, de acuerdo con los parámetros impuestos por los hitos fijados en cada una de ellas[23].

[23] JACOBSON, Ivar; BOOCH, Grady y RUMBAUGH, James. El Proceso Unificado de Desarrollo de *Software*. Madrid: Pearson Addison-Wesley, 2000, p. 10..

Figura 9. Definición gráfica del término incremental en RUP



Fuente: las autoras

Igualmente, es importante resaltar que una de las ventajas ofrecidas por esta característica es la posibilidad de proponer planes de contingencia escalonados con el fin de mitigar los riesgos. En este contexto, el riesgo se define como "una variable que pone en peligro o impide el éxito del proyecto"[24], pues al ser una fuente de problemas, puede incluso llevar a la cancelación de lo planificado. En algunos casos, es consecuencia de una mala identificación del ámbito del proyecto, y en otras ocasiones, es innato al mismo.

Dirigido por casos de uso

Los casos de uso dirigen las acciones en RUP debido a las siguientes condiciones: 1) Permiten que los requisitos se identifiquen a través de ellos; 2) Hacen parte de la planificación iterativa, pues en la definición y delimitación de una iteración, se tienen en cuenta algunos de sus escenarios o su definición completa; 3) Orientan el diseño, ya que en su realización es preciso diseñar objetos y subsistemas destinados a colaborar con su ejecución; 4) Influyen en la organización de algunos artefactos

[24] Ibid., p. 350.

establecidos en los flujos de trabajo de cada fase[25]; 5) Favorecen la unificación e integración de dichos artefactos.

Los casos de uso constituyen una forma fácil de acercar al desarrollador con sus usuarios. Gracias a su diseño visual es factible incorporarlos en un proyecto y hacerlos partícipes en la definición, construcción, verificación y refinamiento de un sistema de *software*.

Figura 10. Dirigido por casos de uso

Fuente: las autoras

Centrado en la arquitectura

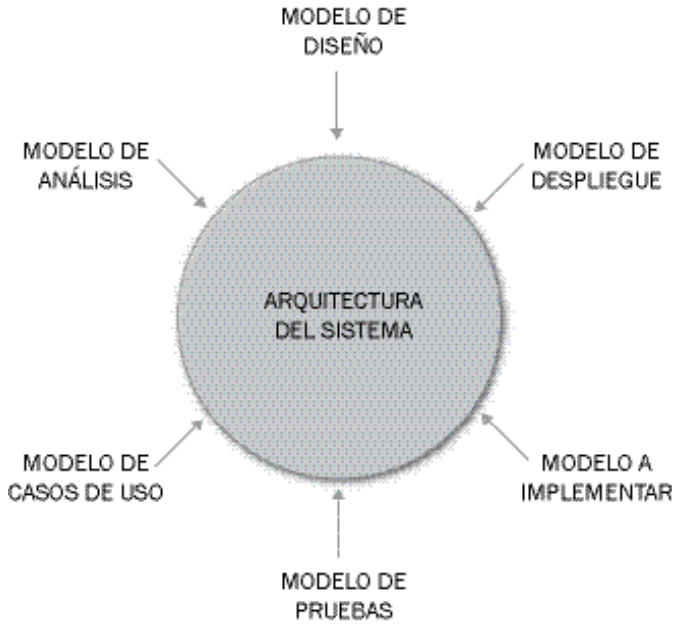
Como base para el desarrollo de un proceso, RUP incorpora el diseño arquitectural desde la fase de factibilidad, en la cual se involucra la organización o estructura de las partes más relevantes del sistema. De este modo, los "stakeholders"[26] adquie-

[25] LARMAN, Op. Cit., p. 155.

[26] Los "stakeholders" son todos aquellos usuarios o desarrolladores que asumen un rol al interior del proyecto y tienen a su cargo la ejecución de distintas actividades.

ren una visión común y una perspectiva completa del trabajo a efectuar[27]. Según Larman[28], la arquitectura es orientada por medio de las siguientes vistas: lógica, proceso, despliegue, datos, casos de uso e implementación.

Figura 11. RUP centrado en la arquitectura



Fuente: las autoras

Otras características

RUP implementa otras reglas o prácticas generales que los grupos de trabajo deben adaptar con el fin de generar un proceso de producción iterativo e incremental más efectivo. Éstas se enuncian a continuación[29].

[27] DÍAZ PÉREZ, María Gabriela. Propuesta de una metodología de desarrollo de *software* educativo bajo un enfoque de calidad sistémica. [en línea]. Caracas: Universidad Simón Bolívar, 2002. [consultado 17 mayo 2006]. Disponible en <http://www.unisimonbolivar.edu.ve>

[28] LARMAN, Op. Cit., p. 178.

[29] LARMAN, Op. Cit., p. 168.

- *Aborda las cuestiones de alto riesgo y valor en las primeras iteraciones*. Tradicionalmente, la gestión en los proyectos en cascada se enfoca en las últimas etapas de los requisitos que requieren mayor esfuerzo, ya sea en ingeniería o en técnica. Esta particularidad puede llevar al fracaso del trabajo, así se haya adelantado un porcentaje considerable del mismo. En consecuencia, el manejo iterativo e incremental RUP contempla a la arquitectura como un artefacto vital, orientado a guiar los proyectos desde la primera fase. En este sentido, considera requisitos funcionales y no funcionales (referentes a inversiones técnicas, económicas y de personal) en aras de garantizar que los temas "complejos" se conviertan en una prioridad conducente a la culminación de lo planeado.
- *Involucra continuamente a los usuarios*. Con el fin de obtener la aprobación necesaria en las distintas fases de desarrollo de un proyecto, RUP debe mostrar los incrementos alcanzados en los artefactos resultantes de las iteraciones. Así corresponde a su carácter iterativo y asegura una relación directa con los futuros usuarios, quienes tienen la oportunidad de moldear los productos según sus requerimientos.
- *Verifica la calidad permanentemente y desde el principio*. A diferencia de los procesos tradicionales, RUP constata en las actividades de planeamiento y gestión de cambios de las distintas fases, la calidad de cada versión de los artefactos. De tal forma, con base en apreciaciones críticas, comprueba el cumplimiento de los hitos de control y certifica que los productos se mantengan con un carácter escalable. La calidad debe verificarse de manera constante en lo concerniente a confiabilidad, funcionamiento y desempeño (tanto de la aplicación como del sistema). Por consiguiente, RUP provee asistencia en la planificación, diseño, implementación, ejecución y evaluación de las pruebas respectivas. De tal suerte que la calidad se evalúa en el proceso mismo, a partir de criterios objetivos, como una actividad presente en todos los momentos y extensible a la integridad de los participantes.
- *Modela visualmente al software*. Un lenguaje visual estándar como UML, ofrece la posibilidad de proyectar los componentes del *software* por medio de herramientas que generan modelos. Éstos se representan en planos y reflejan un ámbito específico del sistema, desde lo general hasta lo particular.

Como lenguaje que es, las características de RUP se orientan a:

1) Modelar el *software*. Sus diferentes modelos se sustentan en un léxico común y siguen reglas que trazan conexiones entre ellos, sin diferenciar un flujo de trabajo

o una fase. Asimismo, aprovecha tales modelos para aplicarlos en las vistas utilizadas en la definición de la arquitectura de un sistema.

2) Visualizar los elementos del *software.* La notación gráfica facilita la rápida comprensión de todos los componentes y niveles. Del mismo modo, la documentación que apoya los modelos sirve de soporte para el mantenimiento y la actualización de éstos.

3) Especificar el *software.* UML determina modelos para todos los flujos de trabajo de un proceso. A su vez, el nivel de profundización y el incremento de las distintas iteraciones, garantiza la congruencia con los requisitos a solucionar.

4) Construir *software.* La correspondencia existente entre UML y los lenguajes de programación orientada a objetos, da cabida a conceptos de ingeniería directa e inversa.

ESTRUCTURA DE RUP

Al representar la estructura de RUP, varios autores[30] lo hacen únicamente con los flujos de trabajo fundamentales, de manera que el modelado de requisitos abarca solo aquellos inmanentes al sistema. Otros estudios[31] analizan dichos flujos por separado e incorporan los que se dedican a la administración del proyecto.

En términos generales, RUP define flujos de trabajo -también llamados disciplinas- inmanentes a la ingeniería (modelado, requerimientos, análisis, diseño, codificación, pruebas e instalación) al igual que otros con funciones de apoyo (administración, configuración y cambios, administración de proyecto y ambiente).

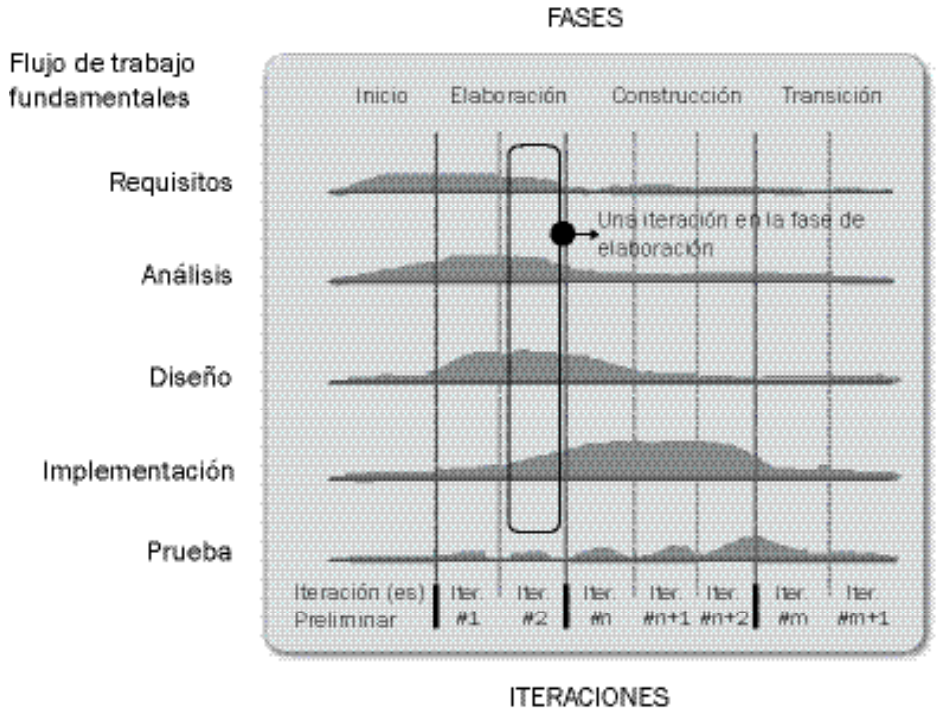
Igualmente, en cada una de las fases de RUP (inicio, elaboración, construcción y transición) se plantea un objetivo principal cuya consecución se logra mediante el cumplimiento de un número de iteraciones que es fijado por la complejidad y el tamaño del proyecto[32].

[30] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 211.

[31] KRUCHTEN, Philippe. The rational unified process: an introduction. 3 ed. Boston: Addison-Wesley, 2005, p. 21.

[32] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 235.

Figura 12. Estructura básica de RUP



Fuente: JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 78.

Como se observa en la figura 12, RUP puede describirse en dos ejes. El horizontal representa el tiempo y muestra el componente dinámico del proceso, expresado en términos de ciclos, fases, iteraciones y objetivos a cumplir[33]. El vertical equivale al aspecto estático y está descrito por actividades, artefactos y roles, dimensiones agrupadas lógicamente por medio de los flujos de trabajo. Cada una presenta mayor desarrollo sobre las fases en que su impacto es más notorio. Por lo tanto, las actividades y sus respectivos artefactos evolucionan con cada iteración.

[33] CONTRERAS, Jaquelina y LACIAR, Verónica. Procesos unificados para modelar sistemas de educación a distancia. En: PÉREZ LÓPEZ, J nombre completo y BRITO DE LA NUEZ, A nombre completo. Memorias del VIII Congreso de educación a distancia CREAD-MERCOSUR/SUL. Córdoba: Instituto Universitario Aeronáutico, 2004, p. 15.

A su vez, los hitos delimitan el alcance tanto de los objetivos contemplados por las actividades como de los artefactos propuestos para cada flujo, de tal suerte que determinan la finalización de una fase.

DISCIPLINAS DEFINIDAS POR RUP

La definición de los trabajadores involucrados, su visión respecto al sistema y los artefactos requeridos, son factores determinantes para establecer el origen de las disciplinas de un proceso. En RUP, la palabra proceso hace referencia[34] al molde que se aplica en diferentes proyectos de desarrollo de *software.* Cabe recordar que este desarrollo comprende las actividades necesarias para convertir los requisitos de usuario en el conjunto consistente de artefactos de un producto.

El proceso también se concibe como el camino seguido en aras de concretar tareas (qué debe hacerse y cómo) y asignar responsabilidades (cuándo es preciso efectuar una actividad y quién la tiene a su cargo). En este orden de ideas, las disciplinas o flujos de trabajo[35] adoptan el carácter de estrategias que son empleadas con el fin de agrupar varias actividades y obtener soluciones mediante los artefactos asociados en los modelos. Sobre estos últimos puede decirse que, al responder a las exigencias de información, conllevan a la culminación de un proyecto. Sus distintas versiones son el resultado de aplicar, en cada fase, las iteraciones procedentes.

Así, los artefactos son componentes de un modelo específico (requisitos, pruebas, instalación), que al ser entendidos por los trabajadores (director, arquitecto de *software*, desarrollador), los clientes y las mismas máquinas, favorecen la comprensión general de los alcances del sistema. Su presencia en todas las fases los hace parte del desarrollo de ingeniería, mientras que por su corta duración, son también considerados como elementos de gestión. En cuanto a su representación, esta se realiza por medio de diagramas UML o argumentos descriptivos.

---

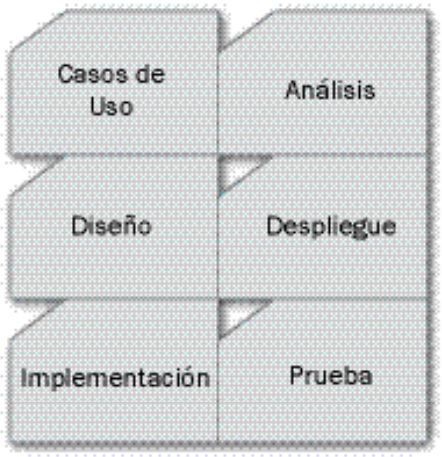
[34] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 87.

[35] En el año 2001, la denominación “flujo de trabajo” (UP) fue sustituida por “disciplina”, un nuevo término formulado con el ánimo de buscar una armonización con el OMG SPEM, un esfuerzo de estandarización internacional. “Flujo de trabajo” ha seguido utilizándose como sinónimo de “disciplina”, aunque esta equivalencia no sea del todo correcta. Su significado es ahora ligeramente distinto en el contexto de un proyecto UP: se trata de la consecuencia de diversas actividades.

El sistema es entonces una colección de modelos relacionados entre sí. Gracias a esa serie de conexiones, todos los participantes de un proyecto pueden comunicarse y negociar en el marco de reglas preestablecidas y con la orientación dirigida hacia objetivos puntuales.

El apoyo de la Ingeniería de *Software* basada en modelos resulta fundamental para que un sistema reciba referentes del mundo real, los cuales han de percibirse en el producto final[36]. De tal modo que el recorrido de RUP permite llegar a un sistema *software* "entero", y no a uno ejecutable[37].

Figura 13. Conjunto fundamental de modelos de RUP



Fuente: JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 20.

A continuación se revisan las disciplinas de RUP con sus características generales[38].

Requisitos

Esta disciplina permite conocer el contexto en que el grupo desarrollador implementará el proyecto. Además, busca proporcionar a los analistas un conocimiento del lenguaje específico de los clientes.

[36] PONS, Claudia. El proceso de desarrollo de *software* basado en modelos. La Plata: Universidad Nacional de La Plata, 2000, p. 1.

[37] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 111.

[38] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 11.

Gracias a este flujo de trabajo, los encargados obtienen artefactos que identifican los procesos y potencian la interacción con los usuarios. De tal forma, se logra formular un sistema óptimo y acorde con los requisitos planteados:

> El propósito fundamental del flujo de trabajo de los requisitos es guiar el desarrollo hacia el sistema correcto. Esto se consigue mediante una descripción de los requisitos del sistema (es decir, las condiciones o capacidades que el sistema debe cumplir) suficientemente buena como para que pueda llegarse a un acuerdo entre el cliente (incluyendo a los usuarios) y los desarrolladores, sobre qué debe y qué no debe hacer el sistema[39].

Para ajustarse a estos parámetros, el analista debe adaptar a la situación puntual las acciones encaminadas a capturar requisitos (casos de uso). Asimismo, ha de tener presente otros puntos: reconocer que cada proyecto es diferente, considerar de qué tipo se trata (implementación, adaptación, o ajuste de arquitectura a servicios), sopesar el grado de conocimiento de los usuarios y examinar la información que servirá de plataforma al sistema. Los pasos dispuestos para el cumplimiento de estas acciones son: enumerar los requisitos candidatos; comprender el contexto del sistema; capturar los requisitos funcionales y los no funcionales[40].

Tabla 1. Artefactos resultantes del Modelado de Negocio (Captura de requisitos)

Trabajo a realizar	**Resumen de los artefactos resultantes**
Enumerar los requisitos candidatos	Lista de características
Comprender el contexto del sistema	Modelo del dominio o del negocio
Capturar los requisitos funcionales	Modelo de casos de uso
Capturar los requisitos no funcionales	Requisitos adicionales o casos de uso concretos (para requisitos específicos)

Fuente: JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 111.

[39] CORPORACIÓN UNIVERSITARIA AUTÓNOMA DEL CAUCA. Flujos de trabajo. [en línea]. Popayán: Corporación Universitaria Autónoma del Cauca, 2007. [consultado 14 nov. 2007]. Disponible en <http://www.uniautonoma.edu.co/_oldweb/docentes/hcordoba/SIGRequerimientos.ppt#430,25,FLUJOS DE TRABAJO>

[40] LARMAN, Op. Cit., p. 45.

1. *Enumerar los requisitos candidatos.* Se trata de relacionar aquellas buenas ideas, surgidas del consenso entre los participantes del proyecto (clientes, desarrolladores) que es factible incorporar en versiones futuras, una vez se hayan suplido los requisitos indispensables. En caso de ser asumidas, especialmente si brindan un valor agregado al sistema y optimizan su funcionalidad y respuesta, se convierten en requisitos que deben ser planificados y evaluados en términos de tamaño, prioridad y riesgos asociados.

2. *Comprender el contexto del sistema.* Es importante que los analistas se encuentren ubicados en dicho contexto. Para ello, resulta primordial manejar la terminología empleada por éste, como también lograr una clara compresión de todo cuanto sucede en el proceso.

El contexto puede plasmarse mediante modelados que al ser aplicados conjuntamente, permiten visualizar las acciones organizacionales (modelado del negocio) y mejorarlas (modelado del dominio). Ahora bien, en la medida que se diseñe el primero, habrá una idea más acertada del sistema.

3. *Capturar los requisitos funcionales*. Gracias a los casos de uso, los analistas determinan lo que esperan y quieren los usuarios. Estos casos representan diversas opciones (una funcionalidad, una actividad ejecutada por una persona con el sistema), por ende, si gracias a su estudio se logran detallar las acciones a efectuar por parte de los clientes, se sabrá qué debe hacer el producto final.

4. *Capturar los requisitos no funcionales.* Estos requisitos -que no corresponden, estrictamente, a peticiones de usuario- involucran términos de la Ingeniería de *Software* cuya presencia ha de evidenciarse en el producto final*,* tales como mantenimiento, rendimiento y fiabilidad,. Con ellos, los analistas identifican las restricciones que pueden repercutir en la funcionalidad del sistema.

Tras aplicar las iteraciones necesarias según el tipo de proyecto, los trabajadores a cargo de esta disciplina (analista de sistemas, especificador de casos de uso, diseñador de interfaz de usuario y arquitecto) han de alcanzar los siguientes artefactos: modelo de casos de uso, identificación de actores, casos de uso, descripción de arquitectura (vista del modelo de los casos), glosario y prototipo de interfaz de usuario. Igualmente, es oportuno recalcar que la mayoría de requisitos se desarrolla en las fases de Inicio y Elaboración, pues es muy bajo el porcentaje de los formulados en la etapa de Construcción.

Análisis

El objetivo final de esta disciplina es analizar los requisitos que fueron descritos en la fase previa. De este modo, es posible comprenderlos plenamente, asegurar su permanencia en todos los momentos, idearlos como un sistema (con la inclusión de su arquitectura inicial) y definirlos de manera conceptual.

Para explorar los casos de uso en consonancia con las características del sistema, el lenguaje del cliente se traduce a otro que es entendido por el desarrollador. Con este propósito se ejecuta la actividad "refinamiento de requisitos", un conducto que ofrece a los trabajadores participantes (arquitecto, ingeniero de casos de uso e ingeniero de componentes) la oportunidad de examinar diferentes detalles.

Son cinco los artefactos resultantes: modelo de análisis, descripción de la arquitectura, realización de casos de uso, clases de análisis, y paquete de análisis.

Tabla 2. Artefactos resultantes de la disciplina de Análisis

Trabajo a realizar	Resumen de los artefactos resultantes
Refinamiento de requisitos	Modelo de análisis
Análisis de la arquitectura	Descripción de la arquitectura
Análisis de casos de uso	Realización de casos de uso
Análisis de clases	Clases de análisis
Análisis de paquetes	Paquete de análisis

Fuente: JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 403.

1. *Modelo de análisis*. Incrementa la comprensión y formalización del sistema descrito en los casos de uso. Este modelo, cuya estructura lo hace flexible a los cambios, es un objeto clave en el diseño, ya que oficia como base para las actividades fijadas en la disciplina.

2. *Descripción de la arquitectura*. Este artefacto propone una vista arquitectónica del modelo de análisis y desglosa el sistema en partes que son modeladas mediante diagramas UML, muy representativos en la concepción de la arquitectura.

3. *Realización de casos de uso*. Permite conocer las vistas funcionales, estructurales y de comportamiento[41]. Así, la parte dinámica se relaciona con la estática, al tiempo que se implementan varios elementos: diagramas de clases (empleados en la representación de esta última); diagramas de secuencia (especifican el comportamiento), y casos de uso (utilizados para describir los requisitos funcionales)[42].

4. Clases de Análisis. Indican niveles de abstracción o subsistemas del diseño[43]. Asimismo, suplen los requisitos funcionales, razón la por cual, los casos de uso se representan en función de ellas. Son de tres tipos: entidad, interfaz o control.

La clase entidad es un modelo de la información perdurable. Para obtenerlo, se parte de los escenarios propuestos en los casos de uso, se sigue luego con un modelado que abarca la descripción de las clases, sus contenidos, atributos e interrelaciones, y se finaliza con la definición de la dinámica, paso consistente en identificar las operaciones efectuadas por cada una de esas clases. Éstas, finalmente, deben ser el resultado de la aplicación de iteraciones e incrementos.

La clase interfaz modela la interacción entre el sistema y sus actores. Su estructuración demanda un manejo de las operaciones y de la información disponible. A partir de esta necesidad, se deduce que una pantalla de interacción con los usuarios, ya sea de entrada o salida de información, requiere una clase de este tipo.

En lo concerniente a las clases de control, es preciso indicar que desde el punto de vista práctico, cualquier cálculo aplicado a un caso de uso (ya sea de operaciones o de la información fluctuante en una base de datos) exige el tratamiento de una entidad de esta naturaleza.

---

[41] NUSEIBEH, B. y EASTERBROOK, S. Requirements Engineering: A roadmap. Conference on the future of *Software* Engineering. En: ACM. Association for Computing Machinery Proceedings of the 22nd Conference on *Software* Engineering, ICSE 2000. Limerick: ACM Press, 2000, p. 37 - 46.

[42] DÍAZ, Isabel; SÁNCHEZ, Juan y PASTOR, Óscar. Metamorfosis: Un marco para el análisis de requisitos funcionales. [en línea]. (2005). [consultado 19 ago. 2008]. Disponible en <http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER05/isabel_diaz.pdf>

[43] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 145.

> Las clases de control permiten manejar los eventos del caso de uso; usualmente, encapsulan el control de un caso de uso (...) además, encapsulan cálculos complejos. Se podría pensar en una clase control como que "ejecuta" o "implementa" el caso de uso. Estas clases se usan durante el "análisis" para representar comportamientos complejos no asociados a un objeto entidad específico[44].

5. Paquete de Análisis. Este artefacto, basado en los requisitos funcionales y el dominio del problema, puede estar conformado por clases de análisis y realización de casos de uso. Permite apreciar los servicios que reúnen las acciones facilitadoras de la funcionalidad del sistema.

La disciplina de análisis, desplegada ampliamente en las iteraciones iniciales de la fase de Elaboración, confiere una arquitectura óptima y consistente con los requisitos.

Diseño

Consolida un sistema que soporte los requisitos determinados en las disciplinas anteriores[45]. Igualmente, ha sido concebida para establecer las relaciones entre clases, interfaces y subsistemas, como también la colaboración que estos componentes deben prestar a la realización de los casos de uso. A diferencia de la disciplina de Análisis, la definición aquí es más física, pues se tiene en cuenta el lenguaje de programación elegido.

Junto a ese lenguaje, en el diseño se consideran otros puntos que guardan relación entre sí: sistemas operativos, componentes del sistema, interfaz de usuario y gestión transaccional. Las actividades de implementación generadas -guiadas por la realización de los casos de uso- representan la totalidad de los requisitos (funcionales y no funcionales)[46].

---

[44] FRANCIA, Joel. Implementando componentes de procesos de usuario. [en línea]. (2007). [consultado 16 nov. 2007]. Disponible en <http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/MTJ_2130.asp>

[45] JIMÉNEZ LÓPEZ, Rafael. Análisis y diseño orientado a objetos de un framework para el modelado estadístico con MGL. Palma de Mallorca, 2003. Trabajo de Grado (Doctorado en Matemáticas). Universitat de lles Ballears. Facultad de Psicología, p. 26.

[46] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 149.

Los artefactos resultantes de esta disciplina son: modelo de diseño, clases de diseño, realización de caso de uso-diseño, subsistemas de diseño, interfaz, descripción de arquitectura (vista modelo de diseño), y modelo de despliegue.

Tabla 3. Artefactos resultantes de la disciplina de Diseño

Trabajo a realizar	**Resumen de los artefactos resultantes**
Diseño de arquitectura	Modelo de diseño
Diseño de casos de uso	Clases de diseño
Diseño de clases	Realización de caso de uso-diseño
	Subsistemas de diseño
	Interfaz
	Descripción de arquitectura (vista modelo de diseño)
	Modelo de despliegue

Fuente: JIMÉNEZ LÓPEZ, Op. Cit., p. 33.

1. *Modelo de Diseño*. Describe la realización física de los casos de uso y proyecta los requisitos en el entorno de implementación del sistema[47]. Se trata de un modelo que incluye diversos elementos: subsistemas, interfaces, clases de diseño, vista arquitectónica y modelo de despliegue. Este último brinda una visión de la forma como se va a instalar y distribuir el producto final, de acuerdo con las especificaciones de red y el sistema operativo.

2. Clases de Diseño. Se obtienen mediante la aplicación de iteraciones sobre las clases definidas en la disciplina de Análisis. Al estar orientadas a modificar o profundizar las operaciones ejecutadas en una clase, pueden originar cambios o especificaciones en razón de un requerimiento de los requisitos, al igual que en la necesidad de realizar algún refinamiento tras haber revisado otros diagramas, como por ejemplo, los de interacción. Precisamente, éstos se diseñan luego de definir las clases en la disciplina anterior y deben ser correspondientes con un bloque de código fuente del programa en desarrollo.

[47] FRANCIA, Op. Cit., p. 23.

3. Realización de caso de uso-diseño. Este artefacto describe cómo se realiza un caso de uso particular en función de los objetos del modelo. Cabe indicar que un caso de uso posee tantas realizaciones (representadas con diagramas de interacción) como escenarios. Éstas se utilizan para conectar los requisitos de usuario con los objetos de diseño que los satisfacen.

4. Subsistemas de diseño. Un sistema puede segmentarse en otros más pequeños, conformados por distintos objetos de diseño (clases, interfaces y realizaciones de casos de uso)[48]. Tal división, aplicada por RUP para posibilitar una mejor organización, facilita que el contenido de esos subsistemas se oriente hacia una o más funcionalidades del conjunto. De este modo, la implementación se verifica de manera escalonada y se obtienen diferentes versiones del producto.

5. Interfaz. El sistema se concibe como la interacción de sus subsistemas. Las interfaces, ligadas a la arquitectura, son el medio que respalda esa relación y favorece la sincronización de los desarrollos simultáneos[49].

6. Descripción de Arquitectura. Comprende los detalles inherentes a la arquitectura de los modelos de diseño y de despliegue. A su vez, identifica elementos asociados a los recursos computacionales, las configuraciones de red, los subsistemas, las interfaces y las clases activas[50].

7. Modelo de Despliegue. Por medio de un modelo de objetos, este artefacto determina la distribución física (localización) de los recursos de cómputo dispuestos para implementar el sistema. Como todos los elementos de RUP, mantiene un diálogo con otras partes gracias a los medios de comunicación dispuestos. Tiene en cuenta las diferentes configuraciones de red, es decir, aquellas destinadas a las actividades de producción, pruebas y simulación.

La disciplina de Diseño -que involucra al Arquitecto y a dos Ingenieros (de casos de uso y de componentes)- es la principal en la fase de elaboración y en las primeras iteraciones de la construcción.

---

[48] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 135.

[49] JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 117.

[50] En una clase activa, los objetos poseen uno o más procesos o hilos de ejecución, por lo tanto, pueden originar actividades de control. Es igual a otra clase, aunque dichos objetos trabajan concurrentemente.

Implementación

En esta disciplina, que representa las actividades relacionadas con la programación, se verifican varias acciones[51]. Por una parte, se planea el proceso de integración, ya que el producto final, como cualquier otro artefacto, resulta segmentado en subsistemas que ameritan la aplicación de, por lo menos, una iteración y un incremento. De otro lado, se realiza la distribución lógica y física del sistema, para lo cual se consideran los recursos computacionales y sus funcionales, identificados en la disciplina anterior mediante el modelo de despliegue. Por último, antes de ser integrados, los subsistemas se prueban de forma individual.

Tabla 4. Artefactos resultantes de la disciplina de Implementación

Trabajo a realizar	Resumen de los artefactos resultantes
Implementar la arquitectura	Modelo de implementación
Integrar el sistema	Componentes
Implementar los subsistemas	Subsistemas de implementación
Implementar las clases	Interfaz
Realizar prueba de unidad	Descripción de arquitectura (vista del modelo de implementación)
	Integración de construcciones

Fuente: JACOBSON; BOOCH y RUMBAUGH, Op. Cit., p. 129.

Los artefactos generados exclusivamente en esta disciplina (no las versiones modificadas de otros que han sido producidos en momentos previos), son:

1. Modelo de Implementación. Traduce el modelo de diseño en componentes de programación. En esta traducción deben usarse subsistemas de implementación para aplicar niveles de jerarquía.

2. Subsistemas de Implementación. Son tres las características de este artefacto: está basado en los subsistemas de diseño, su relación de traza es de uno a uno, y puede estar conformado por componentes, interfaces y otros subsistemas[52]. Se manifiesta por medio de mecanismos de empaquetamiento que en el ámbito de la implementación pueden ser: el proyecto de un lenguaje de programación, un

[51] LARMAN, Op. Cit., p. 44.

[52] LARMAN, Op. Cit., p. 230.

paquete modelado a través de una herramienta asistida por computador, o un paquete de un lenguaje orientado a objetos.

3. Integración de construcciones. Una versión ejecutable de un sistema -generada por una funcionalidad en particular- es una construcción sobre la que se aplican pruebas de integración. Con base en este hecho, al diseñar un plan destinado a reunir las diferentes construcciones, debe tenerse en cuenta la funcionalidad interna de cada una de ellas en el momento de la implementación, así como las partes del modelo a las cuales afectan.

Esta disciplina se presenta en las iteraciones de la fase de construcción con el fin de crear la línea base ejecutable de la arquitectura. Si tras haber sido implementado el sistema permanecen errores, aparecerá en la etapa de transición. Los trabajadores involucrados son: arquitecto, ingeniero de componentes, e integrador de sistemas.

Prueba

Si bien es cierto que esta disciplina se registra en las cuatro fases del proceso, su mayor aplicación se da en la etapa de construcción, cuando las pruebas basadas en los casos de uso definidos al inicio, se desarrollan para validar la integración y funcionalidad de los módulos diseñados.

Tabla 5. Artefactos resultantes de la disciplina de Pruebas

Actividades a realizar	Resumen de los artefactos resultantes
Planificar prueba	Modelo de pruebas
Diseñar prueba	Casos de pruebas
Implementar prueba	Procedimiento de prueba
Realizar pruebas de integración	Componente de prueba
Realizar prueba de sistema	Plan de prueba
Evaluar prueba	Defectos
	Evaluación de pruebas

Fuente: las autoras

A pesar de que al principio son escasas, las pruebas están presentes en cada una de las fases de RUP, y son guiadas por los casos de uso. En esta disciplina partici-

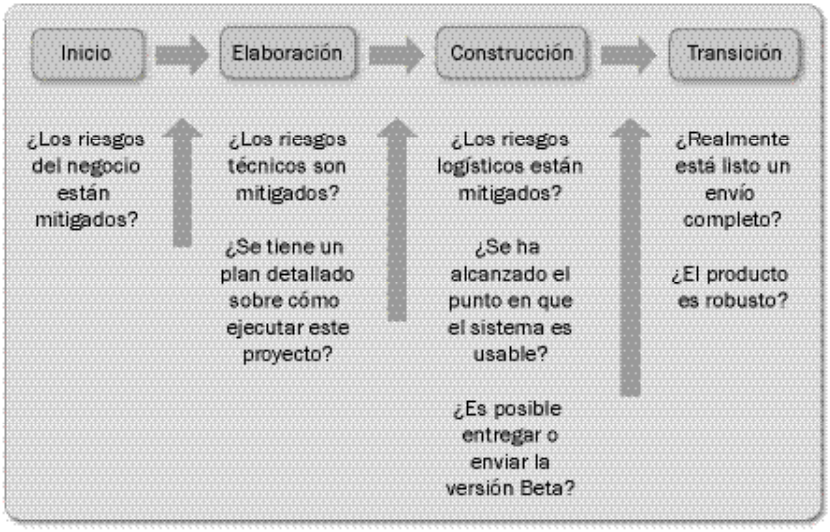
pan cuatro trabajadores: un diseñador de pruebas y tres ingenieros (de componentes, de pruebas de integración y de pruebas del sistema).

FASES DEFINIDAS POR RUP

RUP cuenta con cuatro fases claramente establecidas que determinan el avance del proyecto: Inicio, Elaboración, Construcción y Transición. Cada una de ellas posee objetivos puntuales, y es el cumplimiento de éstos lo que determina si se continúa a la siguiente, o si por el contrario, debe efectuarse una iteración más para lograr su consecución. El desarrollo de una iteración en una etapa específica implica la verificación de todas las actividades definidas (con sus respectivos alcances) en un flujo de trabajo.

Por su parte, el cumplimiento de los requerimientos que han sido identificados en las etapas iniciales y es necesario administrar a lo largo de todo el proyecto, está garantizado por hitos. En RUP, los principales hitos no se expresan en términos de la finalización de artefactos o documentos -como ocurre en otros métodos- sino de mitigación de riesgos y conclusión del producto[53].

Figura 14. Hitos principales en RUP



KROLL, Per y KRUCHTEN, Philippe. The rational Unified Process made easy: A practitioner's guide to the RUP, p. 89.

[53] LARMAN, Op. Cit., p. 135.

La obtención de un hito indica la finalización de una fase y da vía libre para pasar a una nueva.

Tabla 6. Definición de hitos en RUP

	ETAPAS			
	Inicio	**Elaboración**	**Construcción**	**Transición**
Hitos	Objetivos del ciclo de vida	Arquitectura del ciclo de vida	Capacidad operacional inicial	Liberación del producto
Característica principal	Establece la viabilidad	Se centra en la factibilidad	Se remite a la construcción del sistema	Se adentra en el entorno del usuario

Fuente: las autoras

Inicio

En esta fase, la primera del ciclo de desarrollo de RUP, a partir de un diálogo directo entre el cliente y los encargados del trabajo, se definen los propósitos y se planifican los costos y tiempos que regirán al proyecto. Tal interacción debe conducir a un conocimiento global de la empresa objetivo, e igualmente, a una clarificación del impacto, los beneficios y alcances de la solución propuesta. Los objetivos más importantes son[54]:

- Comprender qué se quiere construir.
- Identificar las funcionalidades claves.
- Determinar una posible solución, cuando menos.
- Entender el costo, el calendario y los riesgos asociados.
- Decidir los procesos a seguir y las herramientas a usar.

[54] KROLL, Per y KRUCHTEN, Philippe. The rational Unified Process made easy: A practitioner's guide to the RUP. Boston: Addison-Wesley, 2005, p. 99.

El número de iteraciones a desplegar para atender los objetivos, depende de la complejidad de varios factores: el alcance del sistema, la visión de los "stakeholders", los costos a cubrir y los riesgos técnicos, entre otros. En consecuencia, esta fase puede cumplirse en una sola iteración o requerir más de una.

Los productos obtenidos son:

- Documento de visión general.
- Modelado inicial de los casos de uso (entre el 10 y el 20% de éstos han de quedar listos).
- Glosario.
- Caso de negocio.
- Identificación inicial de riesgos.
- Plan del proyecto.
- Uno o más prototipos.

Elaboración

Al finalizar esta fase, la arquitectura debe encontrarse plenamente delineada y los objetivos encaminados a mitigar los riesgos. Para alcanzar tales estados, se plantean las siguientes metas:[55]

- Entender de manera detallada los requisitos del sistema.
- Diseñar, implementar, validar y elaborar la línea base de la arquitectura.
- Mitigar los riesgos esenciales.
- Refinar los casos de desarrollo.
- Adecuar apropiadamente el ambiente de desarrollo.

El número de iteraciones está supeditado a tres variables, como son: falta de experiencia en el manejo de la aplicación, complejidad del sistema y uso de nueva tecnología. Entonces, podrían precisarse dos o tres iteraciones para mitigar los riesgos claves y lograr una arquitectura acorde con la necesidad que se pretende solucionar.

[55] GUERRERO, Op. Cit., p. 29.

El producto resultante es una arquitectura ejecutable que contemple los riegos y los casos de uso críticos[56].

Construcción

En esta tercera fase se construye el sistema. Guarda cierta semejanza con el proceso de desarrollo, del cual surge el producto *software* definitivo que será entregado a los usuarios finales[57]. Sus objetivos son:[58]

- Minimizar los costos de desarrollo y lograr algún grado de paralelismo.
- Consolidar, de manera iterativa, un producto completo.
- Lograr la suficiencia que permita pasar a la transición con la comunidad de usuarios.

Los productos alcanzados son:

- Un *software* integrado, capaz de correr en la plataforma adecuada.
- Manuales de usuario.
- Descripción del "release" vigente.

Transición

Esta fase busca transferir el *software* a los usuarios del sistema e identificar nuevos ciclos (iteraciones) de acuerdo con el refinamiento necesario y las opiniones del cliente. Tras su realización, se obtiene el producto.

[56] GUERRERO, Op. Cit., p. 35.

[57] GUERRERO, Op. Cit., p. 35.

[58] GUERRERO, Op. Cit., p. 49.

PUNTOS CLAVE DEL CAPÍTULO

- RUP está diseñado para ser manejado a través de sus dos ejes (horizontal y vertical).
- Los artefactos resultantes de cada etapa otorgan una visión del avance y el alcance del proyecto.
- RUP es un marco general de trabajo que debe ser adaptado a cada proyecto.

CAPÍTULO 4

PROCESO DE PRODUCCIÓN DE SOFTWARE RUP – CMMI

Objetivo General: Reconocer la fusión que puede lograrse mediante la integración de las prácticas definidas por CMMI con cada una de las fases establecidas por RUP. Dicha fusión permite cumplir los alcances de las disciplinas inherentes a este marco de trabajo.

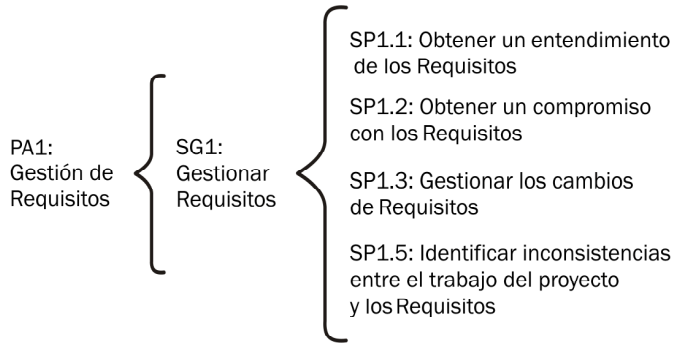
RUP y CMMI poseen características propias, pues el primero es un marco de trabajo, mientras que el segundo, un modelo de calidad. Ahora bien, sus particularidades los hacen complementarios, de manera que es factible plantear una aproximación entre ellos. En este capítulo se examinan las indicaciones propuestas por CMMI en su Nivel 2, las cuales sería recomendable implementar en las disciplinas de RUP.

En aras de facilitar la comprensión de la fusión aquí formulada, se diseñó una matriz de integración RUP -CMMI[59]. A partir de ella, se presentan a continuación las mencionadas recomendaciones de CMMI para ser incorporadas en RUP.

[59] El anexo 1 contiene dicha matriz.

GESTIÓN DE REQUISITOS

Figura 15. Gestión de requisitos



Fuente: las autoras

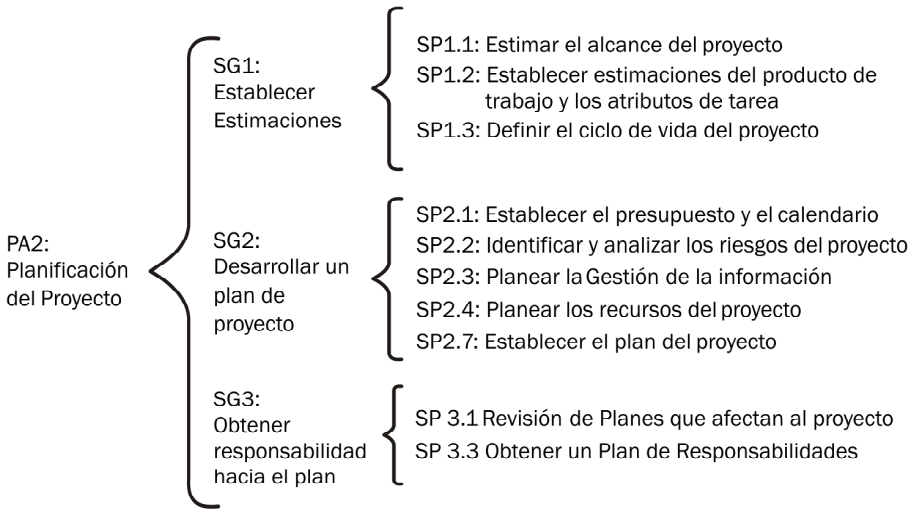
Los requisitos señalan el alcance real del resultado al que se desea llegar con la producción de *software.* En la medida que sean comprendidos de manera óptima, el producto final se acercará a las verdaderas exigencias de los usuarios. Los requisitos (funcionales y no funcionales), deben contemplar:

- Visión general del proyecto. Por medio de esta visión se identifican la misión y los objetivos organizacionales que serán respaldados con el producto. También debe proporcionar absoluta claridad sobre los horizontes a seguir, y puntualizar lo relativo a las dependencias, unidades, prácticas y personas que se beneficiarán.
- Compromiso. Cuando se toma la decisión de persuadir a un usuario con el fin de que use un sistema de información, previamente se han diagnosticado las necesidades de ese beneficiario potencial. Por ende, los requisitos enunciados con base en tal indagación adquieren el carácter de compromisos y su cumplimiento debe hacerse efectivo mediante la elaboración de los distintos artefactos. Asumir dichas obligaciones como alcances y plasmarlas en cada fase, ha de ser un propósito sustancial del proyecto. Para tal fin, a los encargados les corresponde generar procesos de planeación y ejecución que permitan comprobar la implantación de los requisitos en todas las partes de los artefactos.

- Gestión de cambios. Durante el recorrido de un proceso, los requisitos pueden modificarse por múltiples causas. Entre ellas se encuentran: cambios administrativos de notoria incidencia en algún procedimiento; el concepto de los usuarios en torno a la posibilidad de lograr mejorías con la introducción de dichos ajustes; un viraje en las políticas institucionales o en los objetivos misionales de la organización, y fallas en la detección de las necesidades o en la definición de los requisitos. Ante cualquiera de estas circunstancias, resulta imperativo adoptar planes de contingencia que posibiliten la rápida y efectiva adaptación de los artefactos afectados frente a las novedades. Naturalmente, es mucho más práctico implementar tales medidas si se ha llevado un seguimiento directo de las transformaciones y adecuaciones.
- Reconocimiento de las inconsistencias. Cuando la equívoca ejecución de los procedimientos obliga a corregir los artefactos, es preciso constatar qué personas y actividades se han visto perjudicadas. Así se crea el plan de contingencia de errores, cuya aplicación puede hacerse efectiva en el proyecto en curso o en uno futuro que reúna al mismo grupo de analistas y desarrolladores.

PLANIFICACIÓN DEL PROYECTO

Figura 16. Planificación del proyecto



Fuente: las autoras

Numerosos proyectos concebidos para producir un *software,* no llegan a feliz término debido a su falta de planeación y gestión organizacional. La imprecisión al calcular el tiempo de trabajo, los presupuestos pocos reales y la inclusión de recursos desactualizados, constituyen obstáculos para el desenvolvimiento de las actividades y llegan a incidir en la calidad de los productos.

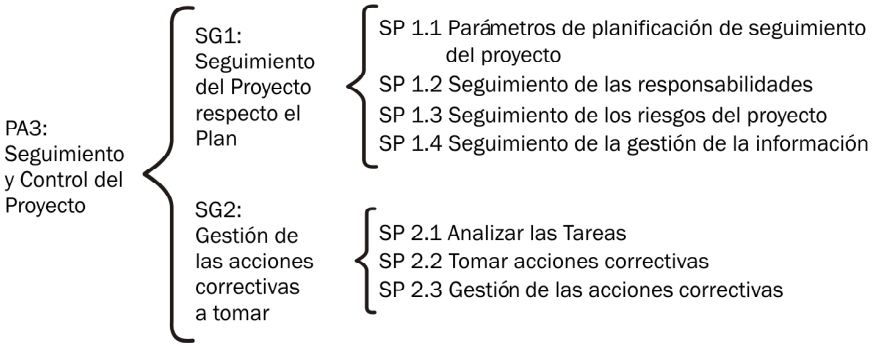
Por ello, las estimaciones juegan un papel fundamental. En éstas, los alcances del proceso son medidos y concretados con exactitud, en medio de un lenguaje común en términos de tiempo, calidad y funcionalidad de los productos. Tales mediciones confieren una visión certera de los recursos requeridos para desplegar las actividades, es decir, para elaborar el *software*.

Asimismo, algunas de las iteraciones características de RUP, al estar orientadas hacia el cumplimiento de requisitos y la disminución de riesgos, favorecen la detección de las causas internas y externas que pueden ocasionar la terminación del proyecto.

La complejidad de todas estas valoraciones sugiere la importancia de especificar el papel de las personas involucradas en un proyecto. Las actividades planteadas implican que los participantes asuman "roles" particulares. Por medio de éstos, cada quien reconoce los objetivos a alcanzar, las responsabilidades que le competen, los alcances esperados, las exigencias a suplir y los parámetros de calidad trazados.

SEGUIMIENTO Y CONTROL

Figura 17. Seguimiento y Control



Fuente: las autoras

Los proyectos de Ingeniería de *Software* sólo funcionan si existen actividades de seguimiento que evalúen el avance de los procedimientos acordados para alcanzar los objetivos. Dichas actividades deben proporcionar herramientas que examinen el plan estructurado en lo atinente a tiempo, responsabilidades y alcances.

En este orden de ideas, el primer paso a seguir consiste en enunciar los parámetros que permitan soportar un seguimiento de las actividades por realizar. La selección de dichos lineamientos es clave para el control del proceso, pues de ellos depende que, en diferentes momentos, se establezca el cumplimiento del plan y se corroboren los progresos reales. Asimismo, posibilitan el análisis de la manera como cada trabajador, desde su propio rol, ha cumplido sus objetivos y responsabilidades en aras de contribuir a la terminación de las fases o a la generación de versiones previas o definitivas de los artefactos.

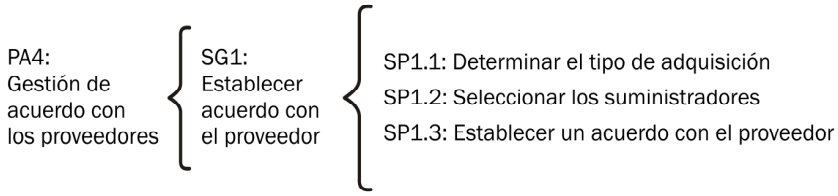
Otro frente fundamental para RUP y CMMI es el control de riesgos. El conocimiento de las posibles problemas constituye la base para la creación de iteraciones, y por ende, para el diseño de las actividades y los artefactos orientados a mitigarlos. En el desarrollo de *software*, los riesgos están revestidos de una doble condición: llegan a convertirse en oportunidades para mejorar los resultados, o se traducen en amenazas que inciden en la terminación del proceso. Por consiguiente, se les debe tener presentes con la misma intensidad, ya sea a largo, mediano o corto plazo.

A su vez, la forma como se produzca, administre y vigile la información, influye en la acertada ejecución del plan de trabajo. Un manejo adecuado de los datos obtenidos permite examinar los procesos y las actividades, al tiempo que facilita la conservación de copias y registros de los pasos ejecutados.

Claro está que el control no se limita al seguimiento de los procedimientos, pues también abarca acciones de corrección. Se corrige cuando algún requisito no ha sido identificado de manera acertada, o simplemente, en vista de las transformaciones de los requerimientos. Para tal efecto, se elabora un plan que contiene las tareas encaminadas a paliar los errores de ejecución.

GESTIÓN DE ACUERDO CON LOS PROVEEDORES

Figura 18. Gestión de acuerdo con los proveedores



Fuente: las autoras

De la adquisición de insumos -en especial de tecnología en el caso de los proyectos de producción de *software*- se derivan en buena parte los éxitos en la implantación y el desarrollo de los procesos. Este punto exige especial atención, ya que una compra inconveniente u obsoleta, pone en peligro todo lo realizado y puede ocasionar resultados negativos. Por ello, al decidir las contrataciones de un proyecto, es preciso considerar:

Por parte del cliente:

- La tecnología con que cuenta.
- Los recursos económicos disponibles.
- El nivel tecnológico de los futuros usuarios.
- La infraestructura física y locativa existente.
- La capacidad de expansión, tanto física como administrativa.
- Por parte del proceso:
- Los requerimientos básicos (de desarrollo, de implantación y de funcionamiento).
- La infraestructura locativa necesaria.

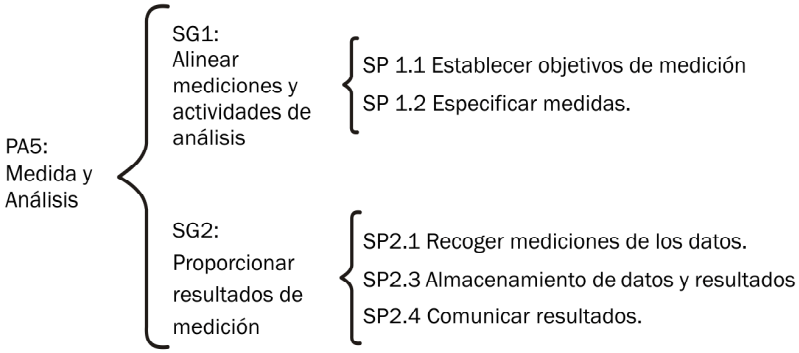
Una revisión juiciosa de los anteriores puntos conlleva a concretar los insumos tecnológicos requeridos. Tales exigencias, a su vez, operan como un marco referencial que, conjugado con el examen de las especificidades de cada proveedor, determina la selección definitiva de éste. Así, las características a sopesar en un oferente son:

- Oferta económica.
- Garantía y capacidad de expansión de la propuesta.
- Transmisión tecnológica a los usuarios.
- Aspectos legales y compromisos.
- Tiempo para la entrega e implementación.

Si estos elementos satisfacen las expectativas y se construye una relación comercial sólida y segura con el proponente escogido, las contrataciones de tecnología resultarán favorables para la organización y se ajustarán a sus demandas.

MEDIDA Y ANÁLISIS

Figura 19. Medida y Análisis



Fuente: las autoras

Los mecanismos destinados a medir el proceso y sus resultados deben ser concertados por los gestores del proyecto y el cliente. Por su parte, la gerencia de este último tiene a su cargo el estudio general del conjunto de tareas planeadas, al igual que de cada una en particular. La información obtenida ha de arrojar parámetros utilizables en la estimación de los adelantos del plan de trabajo y del alcance de las actividades, de acuerdo con los requisitos gerenciales de la organización.

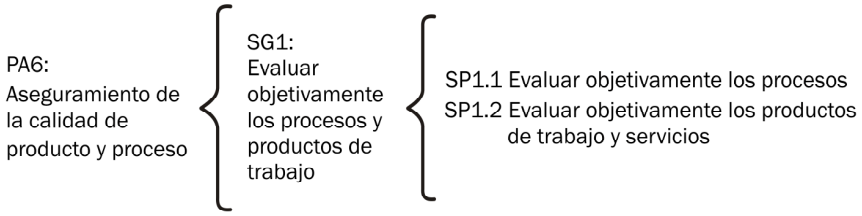
Para las directivas no es necesario conocer los detalles del proyecto, sino su incidencia positiva en el cumplimiento de los objetivos misionales. Por lo tanto, es prio-

ritario visualizar sus puntos claves, por medio de los cuales se mide su capacidad para apoyar los procedimientos internos. Dicho análisis ha de partir de una revisión histórica de sus antecedentes y referentes, como también de los logros alcanzados en otros momentos.

El reconocimiento del ciclo de vida de la organización, sumado a una valoración de cuanto se ha realizado hasta determinado punto en el proceso de producción de *software,* otorga información relevante para explorar las debilidades, fortalezas, amenazas y oportunidades del proyecto.

ASEGURAMIENTO DE LA CALIDAD DE PRODUCTO Y PROCESO

Figura 20. Aseguramiento de la calidad de producto y proceso



Fuente: las autoras

Tras establecer la información pertinente para la gerencia y puntualizar lo que ésta busca, se pasa a evaluar en qué medida los productos y servicios ofrecidos por el proceso de creación de *software* benefician a la organización. De ellos es importante cualificar sus actividades competitivas, es decir, aquellas que conllevan a la consecución de los objetivos misionales. En este sentido, es procedente constatar el grado de relevancia de cada una, como también su perfil en términos de competitividad.

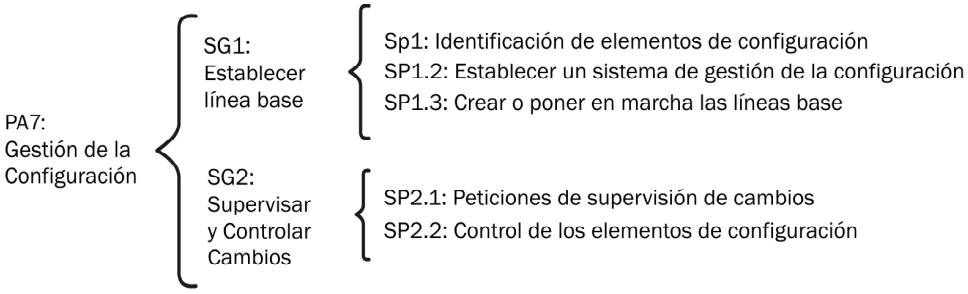
Por otro lado, varios tópicos del proyecto son objeto de análisis: los recursos, su trascendencia, el aporte a la organización y el respaldo que brinda a otras actividades al interior de ésta.

Finalmente, los resultados finales deben ser considerados como:

- Producto. De acuerdo con sus funcionalidades.
- Estrategia de servicio. Según sea su incidencia en frentes concernientes a los productos y servicios: la creación de éstos, su administración, venta y postventa.
- Punto de relación con el cliente y el usuario interno. Se observa el impacto, el uso y la resistencia de los resultados.
- Estrategia de mercado. Se centra en contrastar los resultados alcanzados con los conseguidos por los competidores. También se calculan las facilidades de salida al mercado.

GESTIÓN DE LA CONFIGURACIÓN

Figura 21. Gestión de la configuración



Fuente: las autoras

El éxito en el empleo del producto final debe comenzar desde su instalación en el equipo del cliente. Este procedimiento se inicia con una dinámica de "concientización y descongelamiento", cuya finalidad es desligar a los usuarios de las normas utilizadas en el manejo de los recursos existentes en la organización, para hacerles comprender la pertinencia de implementar el nuevo elemento. Se pasa luego a una etapa de "tránsito", consistente en concientizar al usuario en cuanto se refiere a las pautas que regirán el funcionamiento general. Por último, aparece el "recongelamiento", un paso destinado a transmitir y socializar dichas pautas, para lo cual se cuenta con los productos resultantes. En este momento se efectúan capacitaciones dirigidas a los destinatarios finales, directivos y demás personas directamente vinculadas con el proceso.

También es necesario diseñar el plan de configuración e instalación de la infraestructura tecnológica de respaldo. En la dispoción de tal plataforma han de considerarse los recursos tecnológicos, físicos y humanos disponibles.

Tanto de la etapa de recongelamiento como de la implementación del plan de configuración e instalación, pueden surgir requisitos que, posteriormente, se convierten en cambios a incorporar en los productos *software.* Para adoptar esas modificaciones, es preciso recurrir a nuevas iteraciones.

Finalmente, como punto de partida para la elaboración y adecuación de los artefactos, se sugiere seguir la guía proporcionada por la matriz de integración RUP-CMMI incluida en el anexo 1.

PUNTOS CLAVE DEL CAPÍTULO

- La gestión de requisitos es una etapa clave de un proceso de producción de *software*, pues en ella se identifican su objetivo y su alcance.
- Una buena planificación hace posible medir y visualizar factores de importancia, tales como tiempo, recursos y presupuestos reales. De este modo, se garantiza el desarrollo del proyecto.
- Es fundamental definir actividades que permitan realizar el seguimiento y el control al avance. De tal forma, se identifica el grado de cumplimiento del plan trazado inicialmente.
- Todo proyecto involucra un manejo de proveedores de tecnología. Por tal motivo, es prioritario considerar este frente en las acciones de gestión, ya que su omisión puede convertirse en un problema para el progreso esperado.
- Es necesario definir actividades que apoyen la instalación del producto. Entre éstas debe involucrarse una etapa de "concientización y descongelamiento de los usuarios".

CAPÍTULO 5

PROCESO DE PRODUCCIÓN DE SOFTWARE RUP – CMMI: ¿CÓMO IMPLEMENTARLO?

Objetivo General: Implementar el proceso de producción de *software* mediante una integración de RUP y CMMI que se exprese en la definición de artefactos aplicables al desarrollo de un proyecto.

Los artefactos definidos es este capítulo hacen parte de las prácticas del nivel 2 de maduración del modelo CMMI. Todos pueden experimentar cambios o ampliaciones, de acuerdo con el tipo de proyecto que vaya a implementarse.

PA1: GESTIÓN DE REQUISITOS

Esta práctica busca definir los requisitos del proyecto. Tiene en cuenta sus posibles transformaciones, sus aciertos e incoherencias.

En la fase de Inicio, RUP se remite a la modelación del negocio y la definición de los requerimientos. Para tal fin, ha establecido los siguientes artefactos:

Nombre del artefacto definido por RUP	Tipo de artefacto	Descripción de la información detallada	Flujo de trabajo/Fase	Metas requeridas/ Práctica específica	Artefactos sugeridos (adiciones a los definidos)
	General	Identificación de los cambios realizados en todos los artefactos específicos de la meta trabajada.	Modelo del Negocio-Requerimientos /Inicio	Gestión de Requisitos/ SP1.1	Catálogo de cambios
	General	Definición de los instrumentos destinados a la obtención de la información. Éstos serán utilizados en la meta respectiva.			Registro de los instrumentos de obtención de la información
	Específico	Identificación de la misión, la visión y los objetivos institucionales de la organización en que se desarrolla el proyecto.	Modelo del Negocio-Requerimientos /Inicio	Gestión de Requisitos/ SP1.1	Noción empresarial
	Específico	Identificación del perfil del negocio de la organización en que se desarrolla el proyecto. Se tiene en cuenta el ámbito del mercado (competencia, clientes, ubicación empresarial).	Modelo del Negocio-Requerimientos /Inicio	Gestión de requisitos/ SP1.1	Perfil del negocio
Glosario Empresarial	Específico	Identificación de la terminología manejada por las personas del entorno de la organización en que se desarrolla el proyecto.	Modelo del Negocio-Requerimientos /Inicio	Gestión de Requisitos/ SP1.1	

	Específico	Identificación de las personas claves al interior de la institución en que se desarrolla el proyecto. Con ellas se tendrá contacto directo.	Modelo del Negocio-Requerimientos /Inicio	Gestión de Requisitos/ SP1.1	Visión y personas
					Plantilla de descripción general de requerimientos
Modelo de casos de uso	Específico	Identificación de los actores y de la funcionalidad del *software* con ellos (requisitos funcionales).	Modelo del Negocio-Requerimientos /Inicio		
	Específico	Identificación de los casos de uso en funcionalidad del *software.*	Modelo del Negocio-Requerimientos /Inicio/Elaboración/ Construcción/ Transición		Definición de la implementación de requisitos
	Específico	Proyección de casos de uso/funcionalidad/ Tiempo.	Modelo del Negocio-Requerimientos /Inicio/Elaboración/ Construcción/ Transición	Gestión de Requisitos/ SP1.2	Proyección de casos de uso
	Específico	Proyección de casos de uso/funcionalidad/ Implementación.	Modelo del Negocio-Requerimientos /Construcción/ Transición	Gestión de Requisitos/ SP1.2	Proyección de casos de uso
	Específico	Casos de uso/"stakeholders"/ Proyección.	Modelo del Negocio-Requerimientos /Construcción/ Transición	Gestión de Requisitos/ SP1.2	Proyección e implementación de casos de uso

	General	Definición de riesgos y casos de uso. Se analiza su avance y las variables de control.	Modelado del Negocio-Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/Construcción/ Transición	Gestión de Requisitos/ SP1.2	Definición de riesgos/casos de uso
Estudio inicial de riesgos	General	Listado general de riesgos que pueden provocar la terminación del proyecto.	Modelado de Negocio-Requerimientos-Análisis y Diseño-Implementación-Prueba/ Inicio/ Elaboración/Construcción/ Transición	Gestión de requisitos/ SP1.2	

SG1: Gestionar requerimientos.
SP1.1: Obtener una comprensión de los requerimientos.
SP1.2: Obtener un compromiso con los requerimientos.
SP1.3: Gestionar los cambios de requerimientos.
SP1.5: Identificar inconsistencias entre el trabajo y los requerimientos.

El objetivo principal de esta práctica es gestionar los requisitos técnicos (funcionales) y no técnicos (no funcionales) con el fin de detectar inconsistencias y anormalidades del proceso (relacionadas con información, procedimientos o personas). De tal modo, es posible diseñar el plan a seguir y determinar los elementos necesarios. Cabe recordar que la identificación de los requisitos se basa en el trabajo conjunto entre los clientes y quienes están a cargo del proyecto.

PA2: PLANIFICACIÓN DEL PROYECTO

Una vez identificada la base del proyecto (requisitos funcionales y no funcionales en RUP), la planificación debe orientarse a diseñar, mantener y controlar las actividades, para lo cual recurre a planes de trabajo.

Nombre del artefacto definido por RUP	Tipo de artefacto	Descripción de la información detallada	Flujo de trabajo/Fase	Metas requeridas/ práctica específica	Artefactos sugeridos (adiciones a los definidos)
	General	Con la identificación de los requisitos funcionales y no funcionales, se busca definir la línea base arquitectónica del proyecto. Asimismo, se pretende determinar los alcances de los artefactos.	Modelo del Negocio-Requerimientos /Inicio	Planificación del proyecto/SP1.1	Alcance del proyecto
	General	Identificación de los objetivos específicos de cada actor que se relacionan con el producto final.	Modelo del Negocio-Requerimientos /Inicio	Planificación del proyecto/SP1.1	Objetivos /Actores del proyecto
	General	Identificación de variables de medición, como son el tamaño y las funciones del proyecto.	Modelo del Negocio-Requerimientos /Inicio	Planificación del proyecto/ SP1.2	Medición del proyecto
	Específico	Identificación de parámetros de tamaño, tales como: número de líneas por artefacto (sistema ejecutable), errores detectados, defectos y personas involucradas.	Modelo del Negocio-Requerimientos /Inicio	Planificación del proyecto/ SP1.2	Tamaño del proyecto
	Específico	Determinación de las funciones del artefacto y del nivel de complejidad de cada una de ellas.	Modelo del Negocio-Requerimientos /Inicio	Planificación del proyecto/ SP1.2	Medición

	General	Definición de flujos de trabajo y fases. Se consideran los alcances del proyecto.	Modelo del Negocio-Requerimientos /Inicio	Planificación del proyecto/ SP1.2	Definición del ciclo de vida del proyecto
	Específico	Definición de artefactos de ingeniería de *software*. Los flujos de trabajo se cruzan con las fases del proyecto.	Modelo del Negocio-Requerimientos /Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP1.2	Definición de artefactos de ingeniería
	General	Definición de los riesgos del proyecto en relación con los siguientes factores: cliente, entorno, *software*, proceso de ingeniería, componentes tecnológicos y técnicos.	Modelo del Negocio-Requerimientos /Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP2.2	Riesgos del proyecto
	Específico	Definición del plan de gestión y mantenimiento del riesgo. Identificación de las variables que ocasionan riesgos. Formulación del plan para la mitigación del riesgo.	Modelo de Negocio-Requerimientos /Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP2.2	Plan de gestión y mantenimiento de riesgos
	General	Definición de los recursos humanos del proyecto: identificación de los roles de los “stakeholders”, reconocimiento de las responsabilidades de cada rol.	Modelado de Negocio-Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP2.4	Recursos humanos del proyecto
	General	Definición de los recursos tecnológicos del proyecto de acuerdo con: requerimientos, propuestas, clientes y proveedores.	Modelado de Negocio-Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP2.4	Recursos Tecnológicos

	General	Definición de los procesos de ingeniería relacionados con entradas, actividades, tareas, y salidas.	Modelado de Negocio-Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP2.7	Planeación del proceso de ingeniería
	General	Definición de las responsabilidades del proyecto. Identificación de los equipos de trabajo, los cuales se visualizan conjuntamente, con sus responsabilidades y alcances.	Modelado de Negocio-Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Planificación del proyecto/ SP2.7	Responsabilidades del proyecto

SG1: Establecer estimaciones.
SP1.1: Estimar el alcance del proyecto.
SP1.2: Establecer estimaciones relacionadas con el producto de trabajo y los atributos de tarea.
SP1.3: Definir el ciclo de vida del proyecto.
SG2: Desarrollar un plan del proyecto.
SP2.1: Establecer el presupuesto y el calendario.
SP2.2: Identificar y analizar los riesgos.
SP2.3: Planear la gestión de la información.
SP2.4: Planear los recursos.
SP2.7: Construir el plan del proyecto.
SG3: Obtener responsabilidad hacia el plan.
SP3.1: Revisar los planes que afectan al proyecto.
SP3.3: Obtener un plan de responsabilidades.

Esta práctica comprende tres frentes indispensables:

- Planear las actividades del proyecto.
- Involucrar a personas competentes y con un compromiso óptimo en cada una de esas actividades.
- Mantener el plan durante todo el desarrollo.

Para que dicho plan pueda realizarse y alcance una incidencia positiva en la ejecución del proyecto, es necesario que permanezca constantemente activo, reciba mantenimiento en cada avance y sea actualizado con base en los requerimientos nuevos o en los eventuales riesgos.

PA3: SEGUIMIENTO Y CONTROL DEL PROYECTO

Por medio de dos procedimientos característicos de esta práctica -como son el control y el monitoreo de determinadas actividades- es posible tomar decisiones que permitan adoptar los correctivos necesarios y orientar las tareas del plan del proyecto hacia la consecución de los objetivos planteados.

Nombre del artefacto definido por RUP	Tipo de artefacto	Descripción de la información detallada	Flujo de trabajo/Fase	Metas requeridas/ práctica específica	Artefactos sugeridos (adiciones a los definidos)
	General	Definición del plan de gestión del proyecto (tareas, tiempos y responsabilidades).	Modelo del Negocio-Requerimientos /Inicio	Seguimiento y control del proyecto/ SP1.1	Plan de gestión del proyecto
	General	Prevención y corrección de los eventos y riesgos que limitan el proyecto. Es preciso identificar: tareas, actividades, acciones, preventivas y correctivas de cada uno de ellos.	Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Seguimiento y control del proyecto/ SP1.4	Plan de prevención y corrección de tareas

	Específico	Definición de los cambios autorizados en el proyecto. El documento resultante contendrá información sobre las modificaciones realizadas en cuanto se refiere a los datos de entrada y salida del proceso.	Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/ Construcción/ Transición	Seguimiento y control del proyecto/ SP1.4	Definición de cambios de información
	Específico	Definición de los cambios autorizados en el proyecto. El documento producido contendrá información sobre las modificaciones realizadas en cuanto se refiere a recurso humano (responsabilidades, funciones y desarrollo de actividades y tareas).	Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Seguimiento y control del proyecto/ SP1.4	Definición de cambios de recurso humano
	Específico	Definición de los cambios autorizados en el proyecto. El documento resultante contendrá información sobre las modificaciones realizadas en cuanto se refiere a *software* (funcionalidad entrada-salida y procedimientos).	Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Seguimiento y control del proyecto/ SP1.4	Definición de cambios de funcionalidad del *software*
	General	Definición de la metodología de dirección (dimensión técnica del proyecto, dimensión humana, variables de gestión). Percepción técnica del proyecto. Montaje y pruebas relacionadas.	Requerimientos-Análisis y Diseño/Inicio/ Elaboración/ Construcción/ Transición	Seguimiento y control del proyecto/ SP1.4	Metodología de dirección del proyecto

SG1: Seguimiento del proyecto respecto al plan.
SP1.1: Parámetros de planificación del seguimiento del proyecto.
SP1.2: Seguimiento de las responsabilidades.
SP1.3: Seguimiento de los riesgos del proyecto.

SP1.4: Seguimiento de la gestión de la información.
SG2: Gestión de las acciones correctivas a tomar.
SP2.1: analizar las tareas.
SP2.2: tomar acciones correctivas.
SP2.3: gestionar las acciones correctivas.

Todas las acciones de seguimiento y control del plan del proyecto se basan en elementos claves: definición de la tarea, medición de sus alcances y de lo logrado en diferentes momentos, identificación de los actores y de sus responsabilidades, análisis de la consistencia entre los tiempos ejecutados y los planeados. Si estos parámetros son atendidos debidamente, es posible detectar los riesgos latentes y adoptar las medidas correctivas necesarias.

PA4: GESTIÓN DE ACUERDO CON LOS PROVEEDORES

Para implementar el plan y posibilitar la obtención de cuanto se requiere en la ejecución de las distintas tareas, es necesario tener una idea clara de los recursos disponibles y de sus fuentes de origen. En este sentido, los proveedores juegan un papel fundamental, pues la calidad de los productos finales depende de los insumos y de la tecnología con que se dota al proyecto. De ahí la importancia de conocer rigurosamente la información organizacional de los oferentes y asegurar que sus suministros posean las garantías suficientes.

Nombre del artefacto definido por RUP	Tipo de artefacto	Descripción de la información detallada	Flujo de trabajo/ Fase	Metas requeridas/ práctica específica	Artefactos sugeridos (adiciones a los definidos)
	General	Plan de gestión de adquisiciones: identificación de los riesgos del proyecto que inciden en los activos tecnológicos de la organización; definición del cronograma de adquisiciones y de los documentos requeridos para la realización de éstas.	Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/ Construcción/ Transición	Gestión de acuerdo con los proveedores/ SP1.1	Planificación de Contratación

	General	Documento en el cual se resume: propuesta de cada uno de los proveedores que ofrecen venta de activos, calificación de estos oferentes y diseño de la metodología adecuada para la selección.	Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/ Construcción/ Transición	Gestión de acuerdo con los proveedores/ SP1.1	Documento de relación de proveedores
	General	Documento con los siguientes contenidos: identificación de los activos tecnológicos de la organización (inventario de tecnología que incluya las configuraciones básicas y las limitaciones encontradas al momento del levantamiento).	Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/ Construcción/ Transición	Gestión de acuerdo con los proveedores/ SP1.1	Plan de compras y adquisiciones
	General	Enunciado sobre el alcance del proyecto. Los requerimientos técnicos, en lo concerniente a *hardware*, *software* y comunicaciones, deben aparecer definidos.	Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/ Construcción/ Transición	Gestión de acuerdo con los proveedores/ SP1.1	Plan de compras y adquisiciones
	General	Análisis de activos de tecnología por parte de un experto. Plan de gestión de adquisiciones y plan de cambios de activos.	Requerimientos-Análisis y Diseño/ Inicio/ Elaboración/ Construcción/ Transición	Gestión de acuerdo con los proveedores/ SP1.1	Plan de compras y adquisiciones

SG1: Establecer un acuerdo con el proveedor.
SP1.1: Determinar el tipo de adquisición.
SP1.2: Seleccionar los suministradores.
SP1.3: Establecer un acuerdo con el proveedor.

Es primordial que en los acuerdos pactados con los proveedores se consideren todas las variables inherentes al proyecto (ya sea en materia de responsabilidades, tiempos o alcances), en aras de lograr productos y servicios de óptima calidad.

PA5: MEDIDA Y ANÁLISIS

Esta práctica relaciona al proyecto con su entorno organizacional, pues si bien es cierto que desde el comienzo existen unos objetivos y alcances definidos, éstos deben alinearse con las metas misionales de la organización.

Nombre del artefacto definido por RUP	Tipo de artefacto	Descripción de la información detallada	Flujo de trabajo/ Fase	Metas requeridas/ práctica específica	Artefactos sugeridos (adiciones a los definidos)
	General	Definir los objetivos que deben cumplirse en cada etapa.	Todas las etapas	Medida y análisis / SP 1.1	Definición de objetivos
	General	Definir las métricas para la medición de objetivos.	Todas las etapas	Medida y análisis / SP 1.1	Definición de métricas
	General	Definir estrategias para la evaluación del desempeño de carga, el almacenamiento y la recuperación de datos.	Pruebas/ Transición	Medida y análisis / SP 2.1	Formatos de verificación para el manejo de datos

	General	Definir estrategias para validar el correcto almacenamiento y la recuperación de datos.	Pruebas / Transición	Medida y análisis /SP 2.3	Formatos de validación de datos. Proceso sobre los datos y recuperación
	General	Generar reportes de las pruebas de carga y del almacenamiento de datos.	Pruebas/ Transición	Medida y análisis /SP 2.4	Reporte de resultados

SG1: Alinear las mediciones y las actividades de análisis.
SP1.1: Establecer los objetivos de medición.
SP1.2: Especificar las medidas.
SG2: Proporcionar los resultados de medición.
SP2.1: Recoger las mediciones de los datos.
SP2.3: Almacenar los datos y resultados.
P2.4: Comunicar los resultados.

Las mediciones indicadas permiten identificar aquellos datos que pueden servir como fuente de información y análisis para la gerencia o el área administrativa de la organización.

PA6: ASEGURAMIENTO DE LA CALIDAD DE PRODUCTO Y PROCESO

Los encargados de un proyecto deben garantizar que, a lo largo de todo el desarrollo, se cumplan de manera idónea los procesos diseñados. De esta forma, se busca asegurar la calidad en los distintos frentes.

Nombre del artefacto definido por RUP	Tipo de artefacto	Descripción de la información detallada	Flujo de trabajo /Fase	Metas requeridas/ práctica específica	Artefactos sugeridos (adiciones a los definidos)
	General	Evaluación de los procesos de acuerdo con los estándares definidos.	Análisis y Diseño /Inicio Análisis y Diseño-Implementación/ Construcción. Implementación / Transición	Evaluación objetiva de los procesos y productos de trabajo/ SP1.1	Formato de evaluación de procesos
	General	Evaluación, de acuerdo con los estándares definidos, de los productos y servicios generados.	Todas las etapas	Evaluación objetiva de los procesos y productos de trabajo/ SP1.2	Formatos para la evaluación de productos y servicios

G1: Evaluar objetivamente los procesos y productos de trabajo.
SP1.1: Evaluar objetivamente los procesos.
SP1.2: Evaluar objetivamente los productos de trabajo y los servicios.

En el aseguramiento de la calidad, es pertinente tener en cuenta los siguientes puntos:

- Los procesos planeados, diseñados y desarrollados, deben ser evaluados según los estándares y procedimientos que soportan al proyecto.
- Los encargados del proyecto deben tener la capacidad de reconocer y documentar las falencias, pues la información así registrada resulta de gran valía en la estructuración de estrategias que permitan mejorar o reorientar algunas actividades. Adicionalmente, estos datos propician una retroalimentación permanente entre distintos ámbitos encargados de la ejecución de las acciones.
- Los elementos no aptos deben ser corregidos.

PA7: GESTIÓN DE LA CONFIGURACIÓN

Todo proyecto posee una línea base de elementos que direccionan las diversas tareas, y por ende, el desempeño de las personas participantes. Identificar esa línea desde el momento mismo del planteamiento del trabajo, constituye una condición imprescindible. Más aun: si llegan a presentarse cambios trascendentales durante el desarrollo, es primordial controlarlos y gestionarlos.

Mediante el control de la configuración se construyen esos elementos, en tanto que los sistemas de gestión de la configuración -los cuales siempre deben ser conocidos por los "stakeholders"- garantizan su integración.

SG1: Establecer la línea base.
SP1: Identificar los elementos de configuración.
SP1.2: Establecer un sistema de gestión de la configuración.
SP1.3: Crear o poner en marcha las líneas base.
G2: Supervisar y controlar los cambios.
SP2.1: Efectuar peticiones de supervisión de cambios.
SP2.2: Efectuar el control de los elementos de configuración.

CONTRERAS, Jaquelina y LACIAR, Verónica. Procesos unificados para modelar sistemas de educación a distancia. En: PÉREZ LÓPEZ, Julio y BRITO DE LA NUEZ, Alfredo. Memorias del VIII Congreso de educación a distancia CREAD-MERCOSUR/SUL. Córdoba: Instituto Universitario Aeronáutico, 2004. p. 1-45.

CORPORACIÓN UNIVERSITARIA AUTÓNOMA DEL CAUCA. Flujos de trabajo. [en línea]. Popayán: Corporación Universitaria Autónoma del Cauca, 2007. [consultado 14 nov. 2007]. Disponible en <http://www.uniautonoma.edu.co/_oldweb/docentes/hcordoba/SIGRequerimientos.ppt#430,25,FLUJOS DE TRABAJO>

DÍAZ, Isabel; SÁNCHEZ, Juan y PASTOR, Óscar. Metamorfosis: Un marco para el análisis de requisitos funcionales. [en línea]. (2005). [consultado 19 ago. 2008]. Disponible en <http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER05/isabel_diaz.pdf>

DÍAZ PÉREZ, María Gabriela. Propuesta de una metodología de desarrollo de *software* educativo bajo un enfoque de calidad sistémica. [en línea]. Caracas: Universidad Simón Bolívar, 2002. [consultado 17 mayo 2006]. Disponible en <http://www.unisimonbolivar.edu.ve>

FRANCIA, Joel. Implementando componentes de procesos de usuario. [en línea]. (2007). [consultado 16 nov. 2007]. Disponible en <http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/MTJ_2130.asp>

GUERRERO, Luis. Taller de UML. [en línea]. Santiago de Chile: Universidad de Chile, 2002. [consultado 19 ago. 2008]. Disponible en <http://www.dcc.uchile.cl/~luguerre/cc61j/recursos/clase2.ppt>

HUAROTO, Concha. Propuesta para implantar CMMI en una empresa con múltiples unidades desarrolladoras de *software*. [en línea]. Lima: Universidad Mayor de San Marcos, 2005. 127 p. [consultado 27 mar. 2008]. Disponible en <http://sisbib.unmsm.edu.pe/BibVirtualData/Tesis/Basic/concha_hn/cap2.pdf>

JACOBSON, Ivar; BOOCH, Grady y RUMBAUGH, James. El Proceso Unificado de Desarrollo de *Software*. Madrid: Pearson Addison-Wesley, 2000. 464 p.

JIMÉNEZ LÓPEZ, Rafael. Análisis y diseño orientado a objetos de un *framework* para el modelado estadístico con MGL. Palma de Mallorca, 2003, 241 p. Trabajo de Grado (Doctorado en Matemáticas). Universitat de Iles Ballears. Facultad de Psicología.

KROLL, Per y KRUCHTEN, Philippe. The rational Unified Process made easy: A practitioner's guide to the RUP. Boston: Addison-Wesley, 2005. 416 p.

KRUCHTEN, Philippe. The rational unified process: an introduction. 3 ed. Boston: Addison-Wesley, 2005. 310 p.

LARMAN, Craig. UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado. 2 ed. Madrid: Pearson Educación, 2003. 590 p.

LETELIER, P. Rational Unified Process (RUP). [en línea]. Valencia: Universidad Politécnica de Valencia, 2005. 420 p. [consultado 17 jul. de 2006]. Disponible en <http://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducción%20a%20RUP.doc>

NUSEIBEH, B. y EASTERBROOK, S. Requirements Engineering: A roadmap. Conference on the future of Software Engineering. En: ACM. Association for Computing Machinery Proceedings of the 22nd Conference on *Software* Engineering, ICSE 2000. Limerick: ACM Press, 2000. P. 37 - 46.

PÁEZ, Otoniel. Métricas, estimación y planificación en proyectos de *software.* [en línea]. [consultado 2 oct. 2006]. Disponible en http://www.willydev.net/descargas/WillyDEV_Planea*Software*.Pdf

PALACIO BAÑERES, Juan. Sinopsis de los modelos CMM y CMMI. [en línea]. [consultado 2 oct. 2006]. Disponible en <http://www.navegapolis.net/files/articulos/sinopsis_cmm.pdf>

PHILLIPS, Mike y KONRAD, Mike. Beyond CMMI. [en línea]. [consultado 2 oct. 2006]. Disponible en <http:// www.sei.cmu.edu>

PINO, Francisco, et al. Contribución de los estándares internacionales a la gestión de los procesos de *software* [en línea]. (2006). [Consultado 16 jul. 2008]. Disponible en: <http://alarcos.inf-cr.uclm.es/competisoft/publico/downloads/Inf_T%C3%A9cnicos/COMPETISOFT_IT%202_Contribuci%C3%B3n%20de%20los%20est%C3%A1ndares%20internacionales%20a%20la%20gesti%C3%B3n%20de%20procesos%20*software*.pdf>

PONS, Claudia. El proceso de desarrollo de *software* basado en modelos. La Plata: Universidad Nacional de la Plata, 2000. 20 p.

PRESSMAN, Roger. Ingeniería de *software*: un enfoque práctico. 5 ed. México: Graw-Hill, 2003. 581 p.

REYES, Percy. Construyendo *software* de alta calidad. [en línea]. (2005). [consultado 2 oct. 2006]. Disponible en <http://www.elguille.info/colabora/NET2005/Percy-net_ConstruyendoSoftCalidad.htm>

SOFTWARE ENGINEERING INSTITUTE (SEI). Beyond CMMI v1.2. [en línea]. [consultado 2 oct. 2006]. Disponible en <http// www.sei.cmu.edu>

VARAS, Marcela. Gestión de Proyectos de Desarrollo de *Software*. [en línea]. Concepción: Universidad de Concepción, 2000. [consultado 19 ago. de 2008]. Disponible en: http://www.inf.udec.cl/~mvaras/gpis/apunteGPDS.pdf

3WSTUDIO. Bloques de construcción de UML. [en línea]. (2007). [consultado 7 mar. 2007]. Disponible en <http://3wstudio.com.ar/index.php?dCat=32>

BASTARRICA, Cecilia y HURTADO, Julio. Proyecto SIMEP/SW: Hacia una línea de procesos ágiles, Versión 1.0. Cali: Universidad del Cauca, 2005. 139 p.

BOOCH, Grady; RUMBAUGH, James y JACOBSON, Ivar. El Lenguaje Unificado de Modelado. Madrid: Addison Wesley Iberoamericana, 1999. 432 p.

CANALES MORA, Roberto. Conocimiento aplicado a nuevas tecnologías y mejora de empleo. [en línea]. (2004). [consultado 14 sep. 2008]. Disponible en <http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=cmmi>

CUEVA, Juan Manuel. Calidad del *software.* [en línea]. (1999). [consultado 2 oct. 2006]. Disponible en <http://gidis.ing.unlpam.edu.ar/downloads/pdfs/Calidad_software.PDF>

DEMARCO, Tom. Structured Analysis and System Specification. Englewood Cliffs: Prentice Hall, 1979. 352 p.

GACITÚA BUSTOS, Ricardo. Métodos de Desarrollo de *Software*: El desafío pendiente de la estandarización. Chilán: Universidad del Bío-Bío de Chile, 2003. 39 p.

GRUPO CONSULTORÍA CMMI. [en línea]. (2007). [consultado 18 mar. 2008]. Disponible en <http://www.grupoconsultoria.com.co/cmmi.htm>

HERRERA GONZÁLEZ, Patricia. CMMI-SW (por etapas). [en línea]. [consultado 13 mar. 2008]. Disponible en <http://alarcos.inf-cr.uclm.es/doc/calidad/Trabajos/CMMI%20por%20etapas%202.pdf>

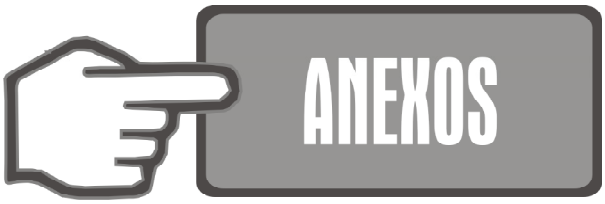
IEEE COMPUTER SOCIETY. Guide to the *Software* Engineering Body of Knowledge "SWEBOK", 2004 version. [en línea]. (2004). [consultado 16 sep. 2008]. Disponible en <http://www.swebok.org>

LÓPEZ PÉREZ, Carmelo. Modelo de madurez de la capacidad del *software*. En: Revista de Ingeniería Informática del CIIRM. [en línea]. No. 3 (2005). [consultado 15 mar. 2008]. Disponible en <www.cii-murcia.es/informas/ene05/articulos/Modelo_de_Madurez_de_la_Capacidad_del_Software.html>

SALAZAR CARABALLO, Luis Antonio. Prolegómenos sobre el Lenguaje de Modelado Unificado (UML) - Segunda Parte. [en línea]. (2004). [consultado 1 feb. 2007]. Disponible en <http://www.topgroup.com.ar/es/home.do?action=listhome>

SCHACH, Stephen. Análisis y diseño orientado a objetos con UML y Proceso Unificado. Madrid: McGrawHill, 2005. 458 p.

TOPGROUP TECHNOLOGY & SOLUTIONS. Metodología de Desarrollo. [en línea]. (2006). [consultado 1 feb. 2007]. Disponible en: <http://www.topgroup.com.ar/es/home.do?action=listhome>



MATRIZ DE INTEGRACIÓN RUP CMMI

(SE PRIORIZA SU CALIDAD)

INTEGRACIÓN RUP_CMMI		METAS REQUERIDAS CMMI	FASES RUP																									
			INICIO							ELABORACIÓN							CONSTRUCCIÓN							TRANSICIÓN					
FLUJOS DE TRABAJO RUP	MODELADO DEL NEGOCIO		PA1	PA2	PA3		PA5	PA6	PA7	PA1	PA2	PA3	PA6	PA5	PA6		PA1	PA2	PA3		PA5	PA6		PA1	PA2	PA3	PA5	PA6	
	REQUERIMIENTOS				PA3		PA5	PA6	PA7			PA3	PA6	PA5	PA6				PA3		PA5	PA6				PA3	PA5	PA6	
	ANÁLISIS Y DISEÑO				PA3		PA5	PA6	PA7			PA3	PA6	PA5	PA6				PA3		PA5	PA6				PA3	PA5	PA6	
	IMPLEMENTACIÓN				PA3		PA5	PA6				PA3	PA6	PA5	PA6				PA3		PA5	PA6				PA3	PA5	PA6	
	PRUEBA				PA3		PA5	PA6				PA3	PA6	PA5	PA6				PA3		PA5	PA6				PA3	PA5	PA6	
	DESARROLLO				PA3							PA3							PA3							PA3			
	FLUJOS DE TRABAJO DE SOPORTE																											
	ADM. CONFIG. Y CAMBIO		PA2		PA3	PA4	PA5	PA6	PA7	PA2	PA4	PA3	PA6	PA5	PA6	PA7	PA2		PA3	PA4	PA5	PA6	PA7	PA2		PA3	PA5	PA6	PA7
	ADMINISTRACIÓN DEL PROYECTO		PA2		PA3	PA4	PA5	PA6	PA7	PA2		PA3	PA6	PA5	PA6	PA7	PA2		PA3		PA5	PA6	PA7	PA2		PA3	PA5	PA6	PA7
	AMBIENTE		PA2		PA3	PA4	PA5	PA6			PA4	PA3	PA6	PA5	PA6			PA4	PA3	PA4	PA5	PA6				PA3	PA5	PA6	

PA1: Gestión de requisitos
PA2: Planificación del Proyecto
PA3 Seguimiento y Control del Proyecto
PA 4 Gestión de acuerdo con los proveedores
PA 5 Medida y Análisis
PA6 Aseguramiento de la calidad de producto y proceso
PA7: Gestión de la Configuración.

Fuente: Las Autoras

Este libro se terminó de imprimir
en el mes de abril de 2015 en
PANAMERICANA Formas e Impresos S.A.